Set Domande

FONDAMENTI DI INFORMATICA

INGEGNERIA INFORMATICA E DELL'AUTOMAZIONE (D.M. 270/04)

Docente: *Tradigo Giuseppe*

Generato il	17/06/2022 14:13:25
N° Domande Aperte	115
N° Domande Chiuse	586

Indice

Indice Lezioni ..... p. 2
Lezione 002 ..... p. 5
Lezione 003 ..... p. 7
Lezione 004 ..... p. 8
Lezione 005 ..... p. 10
Lezione 006 ..... p. 12
Lezione 007 ..... p. 14
Lezione 008 ..... p. 16
Lezione 009 ..... p. 18
Lezione 010 ..... p. 20
Lezione 011 ..... p. 22
Lezione 012 ..... p. 24
Lezione 013 ..... p. 26
Lezione 014 ..... p. 28
Lezione 019 ..... p. 30
Lezione 020 ..... p. 32
Lezione 021 ..... p. 34
Lezione 022 ..... p. 36
Lezione 023 ..... p. 38
Lezione 024 ..... p. 40
Lezione 025 ..... p. 42
Lezione 026 ..... p. 44
Lezione 027 ..... p. 46
Lezione 028 ..... p. 48
Lezione 029 ..... p. 50
Lezione 032 ..... p. 52
Lezione 033 ..... p. 54
Lezione 034 ..... p. 56
Lezione 035 ..... p. 57
Lezione 036 ..... p. 59
Lezione 037 ..... p. 61
Lezione 038 ..... p. 63
Lezione 039 ..... p. 65
Lezione 040 ..... p. 67
Lezione 041 ..... p. 69
Lezione 042 ..... p. 71
Lezione 043 ..... p. 73

Lezione 044 ..... p. 75
Lezione 045 ..... p. 77
Lezione 046 ..... p. 78
Lezione 047 ..... p. 79
Lezione 048 ..... p. 80
Lezione 049 ..... p. 82
Lezione 050 ..... p. 84
Lezione 051 ..... p. 85
Lezione 052 ..... p. 87
Lezione 053 ..... p. 89
Lezione 055 ..... p. 91
Lezione 056 ..... p. 92
Lezione 057 ..... p. 94
Lezione 058 ..... p. 96
Lezione 059 ..... p. 98
Lezione 060 ..... p. 99
Lezione 061 ..... p. 101
Lezione 062 ..... p. 103
Lezione 063 ..... p. 105
Lezione 064 ..... p. 107
Lezione 065 ..... p. 109
Lezione 066 ..... p. 111
Lezione 067 ..... p. 113
Lezione 068 ..... p. 115
Lezione 070 ..... p. 117
Lezione 071 ..... p. 118
Lezione 072 ..... p. 120
Lezione 073 ..... p. 122
Lezione 074 ..... p. 124
Lezione 075 ..... p. 126
Lezione 076 ..... p. 128
Lezione 077 ..... p. 130
Lezione 080 ..... p. 132
Lezione 081 ..... p. 134
Lezione 082 ..... p. 135
Lezione 083 ..... p. 137
Lezione 084 ..... p. 139
Lezione 085 ..... p. 141
Lezione 086 ..... p. 143
Lezione 088 ..... p. 145

Lezione 089 ..... p. 147
Lezione 090 ..... p. 149
Lezione 091 ..... p. 151
Lezione 092 ..... p. 153
Lezione 093 ..... p. 154
Lezione 094 ..... p. 156
Lezione 095 ..... p. 158

Lezione 002

01. Qual è la definizione corretta di un computer?

- [ ] è una macchina intelligente cui porre domande
- [ ] è una macchina limitata che va istruita per fare calcoli
- [ ] è una macchina che esegue codice predeterminato
- [ ] è un sistema che comprende il linguaggio umano

02. Che cos'è la programmazione?

- [ ] La programmazione è una pianificazione numerica
- [ ] La programmazione permette di eseguire una sequenza di istruzioni
- [ ] La programmazione permette di guidare l'utente finale verso la soluzione da lui cercata
- [ ] La programmazione permette di preparare una sequenza di istruzioni

03. Qual'è la definizione migliore di Algoritmo?

- [ ] è un procedimento i cui passi sono chiari ed ambigui
- [ ] è un procedimento composto da una sequenza illimitata di passi
- [ ] è un procedimento composto da una sequenza limitata di passi
- [ ] è un procedimento in grado di prevedere un problema

04. Quale fra queste è una definizione falsa di Algoritmo?

- [ ] un problema è calcolabile se e solo se esiste un algoritmo che lo risolve
- [ ] un algoritmo si dice atomico quando i passi che lo costituiscono sono elementari
- [ ] un algoritmo è valido se termina in un numero finito di passi o non termina mai
- [ ] un algoritmo è valido se è definito da una sequenza finita di passi

05. Quale fra le seguenti proprietà è falsa se applicata agli Algoritmi?

- [ ] un algoritmo è valido quando i passi di cui si compone sono finiti
- [ ] un algoritmo è atomico perchè i passi che lo costituiscono devono essere elementari
- [ ] un algoritmo è non allocabile se la memoria che deve contenerlo è piena
- [ ] un algoritmo è non ambiguo quando i passi di cui si compone sono univoci

06. Quale fra le seguenti definizioni è corretta se riferita agli Algoritmi?

- [ ] Un problema si dice calcolabile se non esiste un algoritmo in grado di risolverlo
- [ ] Passando all'algoritmo due dati in ingresso uguali fra loro si possono ottenere risultati diversi
- [ ] Un algoritmo può terminare in tempo finito o non terminare mai
- [ ] Passando all'algoritmo due dati in ingresso uguali fra loro si devono ottenere risultati identici

07. Quale fra le seguenti definizioni circa le interfacce utente è falsa?

- [ ] l'interfaccia grafica usa la tecnologia WYSIWYG
- [ ] una interfaccia può essere a caratteri o grafica
- [ ] una interfaccia grafica non può contenere caratteri
- [ ] l'interfaccia a caratteri si usa solitamente nei sistemi server

08. Cosa è falso se riferito alle applicazioni che vengono eseguite sui calcolatori?

- [ ] le applicazioni prevedono quello di cui l'utente ha bisogno
- [ ] le applicazioni girano sulle macchine
- [ ] le applicazioni manipolano e gestiscono le informazioni
- [ ] le applicazioni vengono eseguite dalle macchine

09. Scrivere la definizione di Algoritmo.

10. Descrivere le proprietà che devono essere verificate per un Algoritmo.

Lezione 003

01. Descrivere cosa contiene il grafico tipico della Legge di Moore.

02. Si descriva la Legge di Moore e le sue implicazioni.

Lezione 004

01. A cosa serve il BUS?

- [ ] Garantisce l'espandibilità del computer
- [ ] Collega la scheda madre ai registri
- [ ] Garantisce l'espandibilità della CPU
- [ ] Collega la CPU alla memoria

02. Che cos'è il software applicativo?

- [ ] è il sistema operativo e l'insieme dei driver
- [ ] è l'insieme dei componenti software che viene personalizzato in base alle esigenze dell'utente
- [ ] è l'insieme dei programmi che coordinano e organizzano le funzionalità dell'hardware
- [ ] è l'insieme dei componenti software che permettono il funzionamento del terminale

03. Che cos'è falso se riferito alla Macchina di Von Neumann?

- [ ] I suoi componenti principali sono: la cpu, la memoria e le periferiche di I/O
- [ ] I suoi componenti principali sono: la cpu, il bus e il sistema operativo
- [ ] I suoi componenti principali sono: la cpu, il bus e le interfacce delle periferiche
- [ ] I suoi componenti principali sono: la cpu, la memoria e il bus

04. Che cos'è il software di base?

- [ ] un insieme di programmi che coordinano le attività di lavoro dell'utente
- [ ] un insieme di programmi che coordinano le attività e il funzionamento dell'hardware
- [ ] un insieme di programmi che l'utente scrive per risolvere problemi
- [ ] un insieme di programmi che eseguono calcoli per l'utente

05. Che cos'è la Macchina di Von Neumann?

- [ ] Una architettura software per lo scambio di dati
- [ ] Un modello semplificato di calcolatore
- [ ] Una architettura software per il coordinamento dei componenti
- [ ] Una macchina generale in cui dati e programmi risiedono in memorie separate

06. Qual è la frase falsa se riferita alla ALU?

- [ ] La ALU esegue per la CPU solo calcoli aritmetici
- [ ] La ALU è l'unità che esegue calcoli numerici e logici
- [ ] La ALU è l'unità logico-aritmetica
- [ ] La ALU esegue calcoli numerici per la CPU

07. Quale fra i seguenti, non è un elenco di elementi della CPU valido?

- [ ] PC, IR, ALU
- [ ] PC, IR
- [ ] PC, MAB, MBR
- [ ] PC, MAR, MDR

08. Qual è la frase errata riferita al concetto di dato di un calcolatore?

- [ ] Un dato è rappresentato come sequenza di bit
- [ ] Un dato è una informazione elementare
- [ ] Un dato è un valore numerico
- [ ] Un dato non può essere un valore vero/falso

09. Qual è la frase corretta se riferita alla ALU?

- [ ] La ALU esegue calcoli numerico-simbolici per la CPU
- [ ] La ALU è l'unità numerico-aritmetica
- [ ] La ALU è l'unità di esecuzione di codice aritmetico
- [ ] La ALU è l'unità logico-aritmetica

10. Quali fra le seguenti operazioni effettuate durante il ciclo fetch-decode-execute è sicuramente errata?

- [ ] Elaborare operazioni aritmetiche il più velocemente possibile
- [ ] Incrementare l'Instruction Register e copiare le istruzioni nel Program Counter
- [ ] Incrementare il Program Counter e copiare le istruzioni nell'Instruction Register
- [ ] Copiare valori dalla memoria ai registri e viceversa

11. Quale proprietà fra le seguenti è errata se riferita alla memoria?

- [ ] Un valore memorizzato in memoria deve occupare una sola cella elementare
- [ ] Ogni cella di memoria è identificata da un numero intero
- [ ] La memoria è organizzata in celle, in un numero finito
- [ ] Un valore memorizzato in memoria può occupare più celle elementari

12. Scrivere la definizione dell'Architettura di Von Neumann.

L'Architettura di Von Neumann è un modello teorico di calcolatore in cui:

Programmi e dati risiedono nella stessa memoria, rappresentati come sequenze di bit.

La CPU è composta da due unità principali:

Unità di Controllo (CU) → coordina le operazioni del calcolatore.

Unità Logico-Aritmetica (ALU) → esegue operazioni aritmetiche e logiche.

Il Program Counter (PC) tiene traccia dell'indirizzo della prossima istruzione.

Il calcolatore esegue le istruzioni seguendo il ciclo fetch-decode-execute, prelevando un'istruzione dalla memoria, decodificandola ed eseguendola.

Dispositivi di input/output permettono la comunicazione con l'esterno.

In sintesi, è un modello di calcolatore generale, sequenziale e programmabile, su cui sono basati la maggior parte dei computer moderni.

13. Descrivere il funzionamento della ALU.

La ALU è l'unità logico-aritmetica della CPU che esegue tutte le operazioni di calcolo necessarie al processore. È una c componenti fondamentali della CPU insieme all'unità di controllo (CU) e ai registri.

Funzioni principali

Operazioni aritmetiche
Addizioni, sottrazioni, moltiplicazioni, divisioni..

Operazioni logiche
Esempio: verificare se un numero è maggiore di un altro.

Come funziona

La CU (unità di controllo) invia i comandi alla ALU indicando quale operazione eseguire.

La ALU prende i dati di input dai registri o dalla memoria.

Esegue l'operazione richiesta.

Restituisce il risultato, che viene memorizzato in un registro o inviato alla memoria.

Aggiorna eventualmente i flag di stato (es. zero, overflow, segno) per influenzare istruzioni future.

14. Descrivere le fasi del ciclo fetch-decode-execute.

È il processo fondamentale con cui la CPU esegue le istruzioni di un programma. Ogni istruzione passa attraverso tre fasi principali:

1. Fetch (prelievo dell'istruzione)

La CPU preleva l'istruzione successiva dalla memoria principale.

Il Program Counter (PC) contiene l'indirizzo della prossima istruzione.

L'istruzione viene copiata nella Instruction Register (IR).

Il PC viene incrementato per puntare all'istruzione successiva.

Obiettivo: ottenere l'istruzione da eseguire.

2. Decode (decodifica dell'istruzione)

L'unità di controllo (CU) legge l'istruzione dall'IR.

Determina quale operazione deve eseguire la CPU.

Identifica eventuali operand (dati su cui lavorare) e la loro posizione (registri, memoria).

Obiettivo: capire cosa deve fare la CPU e con quali dati.

3. Execute (esecuzione)

La ALU o altre unità della CPU eseguono l'operazione richiesta:

Operazioni aritmetiche e logiche

Spostamento dati tra registri e memoria

Controllo di flusso (salti, branch)

Il risultato può essere scritto in un registro, in memoria o nei flag di stato.

Obiettivo: completare l'istruzione e aggiornare lo stato della CPU.

Lezione 005

01. Qual è la sequenza delle operazioni più corretta che porti svuotare una Pila da 3 elementi?

- [ ] pop(), push(), pop(), push(), pop(), push()
- [ ] pop(), push(), pop(), pop(), pop()
- [ ] push(), pop(), pop(), pop()
- [ ] push(), push(), push()

02. A cosa servono le memorie gerarchiche?

- [ ] A salvare spazio su disco
- [ ] A far sopravvivere i dati della RAM al reset della macchina
- [ ] A migliorare le performance di accesso ai dati della RAM
- [ ] A misurare le performance di accesso ai dati della RAM

03. Quale politica implementa una Coda?

- [ ] First out, last in
- [ ] First in, last in
- [ ] First in, last out
- [ ] First in, first out

04. Cosa è corretto dire delle Unità a Disco?

- [ ] L'accesso ai dati avviene in modalità casuale e non dipende dalla posizione del dato
- [ ] L'accesso ai dati avviene in modalità mista e dipende dalla posizione del dato
- [ ] L'accesso ai dati avviene in modalità sequenziale e bisogna leggere prima tutti i dati precedenti
- [ ] L'accesso ai dati avviene con velocità indipendente dalla posizione del dato

05. Qual è la sequenza corretta più corretta per descrivere il funzionamento delle memorie gerarchiche?

- [ ] Cerco il dato, se esiste lo restituisco, se non esiste non ritorno nulla
- [ ] Cerco il dato, se esiste lo restituisco, se non esiste non ritorno zero
- [ ] Cerco il dato, leggo il dato, restituisco il dato
- [ ] Cerco il dato, se esiste lo restituisco, se non esiste lo salvo e lo restituisco

06. Qual'è la struttura della memoria RAM?

- [ ] Suddivisa in tracce e settori
- [ ] Ogni cella ha un indirizzo univoco
- [ ] Per leggere una cella bisogna leggere tutte le tutte le celle che la precedono
- [ ] Accesso alle celle tramite il bus dati

07. Cos'è la RAM?

- [ ] Memoria tipicamente suddivista in tracce e settori
- [ ] Memoria ad accesso diretto
- [ ] Memoria che mantiene i dati in assenza di tensione
- [ ] Memoria ad accesso sequenziale

08. Indicare la frase false riferita alla memoria RAM:

- [ ] è una memoria ad accesso casuale
- [ ] è una memoria ad accesso sequenziale
- [ ] è una memoria ad accesso diretto
- [ ] è una memoria che mantiene il suo stato al riavvio della macchina

09. Cosa è errato dire delle Unità a Disco?

- [ ] L'indirizzo del dato non consente di determinarne la posizione in modo preciso
- [ ] La lettura di un dato implica dover effettuare diversi accessi
- [ ] La velocità di accesso ai dati è indipendente dalla posizione
- [ ] La velocità di accesso ai dati è variabile in base alla posizione

10. Quale politica implementa una Pila?

- [ ] Last in, first in
- [ ] Last out, first in
- [ ] Last in, first out
- [ ] Last in, last out

11. Descrivere il funzionamento di una Pila, le sue funzioni e un esempio.

12. Descrivere a cosa servono e come si utilizzano le memorie gerarchiche.

13. Descrivere le differenze fra una Pila e una Coda.

Una pila è una struttura dati lineare che segue la politica LIFO (Last In, First Out):
L'ultimo elemento inserito è il primo a essere rimosso.
Funziona come una pila di piatti: aggiungi un piatto in cima (push) e lo togli dalla cim

Funzionamento

Inserimento (push)
Aggiunge un elemento in cima alla pila.

Rimozione (pop)
Rimuove l'elemento dalla cima della pila.

Visualizzazione del top (peek o top)
Mostra il valore dell'elemento in cima senza rimuoverlo.

Le memorie gerarchiche sono un insieme di livelli di memoria organizzati in ordine di velocità, costo e c

Più vicine alla CPU → più veloci ma costose e di capacità ridotta.

Più lontane dalla CPU → più lente, economiche e capienti.

Tipici livelli della gerarchia:

Registri CPU → velocissimi, capacità minima, direttamente nella CPU.

Cache → veloce, vicino alla CPU, contiene dati frequentemente usati.

RAM → memoria principale, più lenta della cache, volatile.

Dischi (HDD/SSD) → molto capienti, più lenti della RAM.

Memoria esterna o cloud → lenta, usata per backup o archiviazione massiva.

Pila (Stack) – LIFO

Gestione chiamate funzione: mantiene l'ordine di ritorno delle funzioni in esecuzione (stack delle chiamate).

Undo/Redo: salva operazioni precedenti in applicazioni come editor di testo.

Valutazione espressioni: es. conversione da notazione infissa a postfissa.

Navigazione web: memorizza cronologia di pagine visitate.

Coda (Queue) – FIFO

Gestione processi in CPU: i processi vengono eseguiti in ordine di arrivo.

Code di stampa: documenti vengono stampati in ordine di invio.

Gestione richieste in rete: pacchetti o messaggi elaborati in ordine di arrivo.

Buffer di dati: streaming audio/video

Lezione 006

01. Cosa è sbagliato dire riferendosi alla struttura di un programma Assembler x86?

- [ ] ha sicuramente una intestazione
- [ ] può contenere un segmento stack
- [ ] possiamo trovarci i segmenti intestazione, dati, stack e programma
- [ ] possiamo trovarci i segmenti dati, stack e codice

02. Quale fra le seguenti frasi riferite al linguaggio Assembler è corretta?

- [ ] Gli indirizzi di memoria possono essere soltanto numeri decimali
- [ ] Gli indirizzi di memoria possono essere espressi solo in formato binario o esadecimale
- [ ] Gli indirizzi di memoria possono essere espressi in formato binario, ottale, decimale ed esadecimale
- [ ] Gli indirizzi di memoria binari non si possono utilizzare

03. Cosa è sbagliato dire riferendosi al linguaggio Assembler?

- [ ] I dati vengono caricati nei registri prima di essere elaborati
- [ ] Dati e programmi risiedono in zone diverse della memoria
- [ ] I dati non si possono spostare dalla CPU alla memoria
- [ ] Dati e programmi vengono elaborati dalla CPU

04. Cosa è corretto dire riferendosi ai programmi Assembler?

- [ ] Dati e programmi risiedono entrambi in memoria
- [ ] Dati e programmi risiedono nella CPU
- [ ] Dati e programmi non risiedono entrambi in memoria
- [ ] Dati e programmi sono usati in modo efficiente

05. Cosa è corretto dire riferendosi al linguaggio Assembler?

- [ ] usandolo non si possono implementare tutti i tipi di algoritmi
- [ ] usandolo si possono ottenere programmi molto efficienti nell'uso dell'hardware
- [ ] usandolo si usa poca memoria
- [ ] usandolo non si usano molte risorse di calcolo della CPU

06. Cosa è corretto dire riferendosi al linguaggio Assembler?

- [ ] è un linguaggio di alto livello
- [ ] è un linguaggio di basso livello
- [ ] è un linguaggio molto spesso usato da solo per progettare un sistema
- [ ] è uno dei linguaggi più diffusi

07. Selezionare una proprietà sicuramente falsa del linguaggio Assembler:

- [ ] Assembler è raramente usato da solo in un progetto
- [ ] Assembler è un linguaggio facilmente portabile
- [ ] Assembler è vicino al linguaggio macchina
- [ ] Assembler è ottimizzato per l'hardware

08. Quale fra le seguenti affermazioni è errata se ci riferiamo ai registri Intel x86?

- [ ] i registri sono contenitori di bit
- [ ] solo il registro AX si compone di due parti da 8 bit indirizzabili separatamente
- [ ] il registro AX è composto da due parti: AH ed AL
- [ ] i registri da 16 bit hanno porzioni da 8 bit che si possono usare separatamente

Lezione 007

01. A che serve l'Interprete dei comandi?

- [ ] verifica che i comandi eseguiti non abbiano generato errori
- [ ] verifica che i dati e i comandi immessi dall'utente siano sicuri
- [x] verifica che i comandi invocati dall'utente operino sui dati corretti
- [ ] verifica che i comandi invocati dall'utente siano sicuri

02. In merito allo Stato dei Processi, quale affermazione è corretta?

- [ ] un processo in stato di "Pronto" sarà eseguito appena la memoria diventerà disponibile
- [x] un processo in stato di "Pronto" sarà eseguito appena il processore diventerà disponibile
- [ ] un processo in stato di "In attesa" sarà eseguito appena il processore diventerà disponibile
- [ ] un processo in stato di "In attesa" sarà eseguito appena la memoria diventerà disponibile

03. Che cos'è il kernel?

- [ ] è un gestore della memoria RAM
- [ ] serve ad eseguire il ciclo fetch-decode-execute
- [ ] è una parte interna della memoria
- [x] gestisce l'accesso alle risorse

04. Qual è l'ordine corretto dei livelli di astrazione di un Sistema Operativo, dal più alto al più basso?

- [x] Programmi Applicativi, Interprete dei Comandi, Hardware
- [ ] Programmi Applicativi, Interprete dei Comandi, Memoria
- [ ] Programmi Applicativi, Nucleo, Interprete dei Comandi, Hardware
- [ ] Interprete dei Comandi, Programmi Applicativi, Memoria, Nucleo

05. Cosa è corretto dire del Sistema Operativo?

- [ ] esegue il codice binario
- [ ] gestisce il reset della macchina
- [x] gestisce l'accesso contemporaneo alle risorse
- [ ] gestisce l'accesso della CPU alla memoria

06. A che serve il Gestore della memoria?

- [ ] risponde alle richieste della CPU su eventuali spazi liberi su disco
- [ ] aumenta o diminuisce la memoria in base alle richieste dei processi
- [ ] risponde alle richieste dei processi su eventuali spazi liberi in memoria
- [ ] permette ai processi di salvare dati su disco

07. Quale fra le seguenti affermazioni è falsa?

- [ ] un processo può girare sia in modalità supervisore che in modalità utente
- [ ] i processi del sistema operativo girano in modalità supervisore
- [ ] un processo in modalità supervisore è detto anche privilegiato
- [x] un processo in modalità utente può accedere a tutta la memoria

08. In merito allo Stato dei Processi, quale affermazione è corretta?

- [ ] i possibili stati di un processo sono: in attesa, pronto o errore
- [ ] un processo cambia stato in risposta un evento
- [ ] un processo può trovarsi in uno fra i due possibili stati: modalità supervisore o modalità utente
- [ ] i possibili stati di un processo sono: in attesa o terminato

Lezione 008

01. Quale delle seguenti frasi, relative al Simultaneous Multithreading, è falsa?

- [ ] la tecnologia Simultaneous Multithreading permette di eseguire calcoli vettoriali
- [ ] in un sistema multicore, di solito uno dei core viene assegnato al sistema operativo
- [ ] tutte le istruzioni, per essere parallelizzabili da una architettura superscalare, devono appartenere allo stesso processo
- [ ] la tecnologia Simultaneous Multithreading permette di eseguire più processi in parallelo

02. A cosa serve una architettura Superscalare?

- [ ] ad eseguire più istruzioni in parallelo
- [ ] a scalare il sistema in base ai processi in esecuzione
- [ ] a scalare il sistema rispetto al numero di utenti
- [ ] ad gestire più macchine connesse in rete

03. Cosa è falso se riferito all'evoluzione delle CPU?

- [ ] il numero di transistor è rimasto pressochè invariato negli anni, mentre la frequenza è aumentata
- [ ] le cpu sono di solito caratterizzate da parametri quali: frequenza di clock, numero di registri e numero transistor
- [ ] il numero di transistor, il numero di bit dei registri e la frequenza sono aumentati negli anni
- [ ] la dimensione in bit dei registri è aumentata nel tempo

04. Cosa è falso se ci riferiamo ad una architettura Superscalare?

- [ ] implementa l'Instruction-level parallelism
- [ ] serve ad eseguire più istruzioni in contemporaneamente
- [ ] è sempre possibile velocizzare il codice
- [ ] ogni ciclo di clock possono essere eseguite più istruzioni

05. Gli scenari in cui due istruzioni NON possono essere eseguite in parallelo sono:

- [ ] quando le due istruzioni sono operazioni logico-aritmetiche
- [ ] quando le due istruzioni sono dipendenti fra di loro
- [ ] quando le due istruzioni sono operazioni aritmetiche
- [ ] quando le due istruzioni sono identiche

06. In una architettura superscalare capace di eseguire due istruzioni in parallelo:

- [ ] nessuna delle altre risposte è corretta
- [ ] c'è una e una sola ALU
- [ ] ci sono esattamente due ALU
- [ ] possono esserci molte ALU

07. Quale fra questi NON è un parametro tipico della CPU?

- [ ] numero di registri
- [ ] frequenza di clock
- [ ] numero di celle
- [ ] profondità di bit

08. A cosa serve una Superpipeline?

- [ ] ad eseguire più istruzioni in parallelo
- [ ] ad eseguire calcoli vettoriali
- [ ] ad eseguire più istruzioni in contemporaneamente
- [ ] ad implementare l'Instruction-level parallelism

09. Descrivere in quali casi non è possibile eseguire due istruzioni in parallelo in presenza di Architetture Superscalari.

10. Considerando un sistema Superscalare a due core, a cui arrivano da eseguire le due istruzioni seguenti:

mov ax, 0 (salva il valore zero nel registro ax)
sub ax, 1 (sottrai 1 dal valore registro ax e salvalo in ax)

descrivere tutti gli scenari che si possono verificare nel caso di esecuzione parallela e, per ognuno, il valore finale del registro ax.

Lezione 009

01. Qual è la definizione migliore per le architetture Massive Parallel Computing?

- [ ] permettono di paralllelizzare i processi in un sistema operativo
- [ ] tentano di ovviare alle limitazioni della Legge di Moore
- [ ] permettono di effettuare calcoli in parallelo sulla propria macchina
- [ ] permettono di elaborare problemi usando i principi della fisica dei quanti

02. Quanti stati si possono rappresentare su una macchina dotata di 3 qubit?

- [ ] 8
- [ ] nessuna delle risposte è corretta
- [ ] 3 o multipli di 3
- [ ] esattamente 3

03. Che cos'è un qubit?

- [ ] una coppia di bit
- [ ] un insieme di 4 bit
- [ ] un elemento che vale 0 ed 1
- [ ] un elemento che vale 0 oppure 1

04. Cosa dice la Legge di Moore?

- [ ] che il numero dei computer raddoppia ogni anno
- [ ] che il numero dei transistor cresce in modo lineare
- [ ] che il numero dei transistor raddoppia ogni anno
- [ ] che la quantità di ram raddoppia ogni anno

05. Quale fra le seguenti frasi è falsa se riferita alla rete Internet?

- [ ] Internet è un sistema non centralizzato
- [ ] Il World wide web è uno dei numerosi servizi di Internet
- [ ] Internet serve solo a navigare sul World Wide Web
- [ ] Il servizio di posta elettronica nasce prima del World Wide Web

06. Come viene identificata una macchina sulla rete Internet?

- [ ] Da un indirizzo IP
- [ ] Dal percorso di instradamento dei pacchetti
- [ ] Dall'indirizzo del destinatario
- [ ] Dal contenuto del pacchetto

07. Cos'è il WWW?

- [ ] Un servizio per la sincornizzazione di calcolatori in rete
- [ ] Un servizio per far funzionare un browser web
- [ ] Un servizio per mandare e ricevere messaggi di posta elettronica
- [ ] Un servizio per la condivisione di pagine multimediali

08. Cosa si fa per ovviare ai limiti della Legge di Moore?

- [ ] Si costruiscono computer con integrazione subatomica
- [ ] Si raffreddano i processori per migliorare le prestazioni
- [ ] Si costruiscono computer probabilistici
- [ ] Si utilizzano architetture massicciamente parallele

09. Descrivere le principali differenze fra un computer classico e un computer quantistico.

10. Dopo aver descritto brevemente la Legge di Moore, illustrare quali sono le sue limitazioni principali e come si possono superare.

Lezione 010

01. Quali, fra le seguenti, NON è una caratteristica di Posix?

- [ ] Modalità di connessione
- [ ] Specifica di Sistema Operativo
- [ ] Portabilità del codice
- [ ] Interfaccia fra sistemi

02. Quale filosofia generale accomuna Posix, Unix e Linux?

- [ ] Il file system
- [ ] Tutte le periferiche sono a riga di comando
- [ ] La gestione della memoria
- [ ] Tutto è un file

03. Quale fra i seguenti NON è un gruppo di permessi POSIX valido?

- [ ] rw-
- [ ] www
- [ ] r-x
- [ ] --x

04. Che cos'è Posix?

- [ ] Posix, Unix e Linux sono lo stesso Sistema Operativo
- [ ] Posix è Unix
- [ ] è una specifica per un Sistema Operativo
- [ ] è l'implementazione di un Sistema Operativo

05. Cosa hanno in comune Posix, Unix e Linux?

- [ ] Il file system
- [ ] L'albero delle directory
- [ ] La condivisione delle risorse
- [ ] La gestione delle periferiche di I/O

06. Cosa è errato dire se riferito a Posix?

- [ ] è una specifica per creare nuovi Sistemi Operativi
- [ ] contiene dettami su come implementare un Sistema Operativo
- [ ] permette ai sistemi di essere compatibiliti fra loro
- [ ] Unix è basato su Posix

07. Cosa hanno in comune Unix e Posix?

- [ ] Sono entrambi Sistemi Operativi
- [ ] Da un certo momento in poi sono diventati obsoleti
- [ ] Sono identici in termini di gestione delle risorse
- [ ] Da un certo momento in poi sono diventati entrambi degli standard

08. Qual è il modo più corretto per leggere il gruppo di permessi "rw-"?

- [ ] Il file è leggibile e modificabile, ma non eseguibile
- [ ] Il file è leggibile e modificabile
- [ ] Il file è modificabile ma non eseguibile
- [ ] Il file è leggibile, modificabile ed eseguibile

09. Illustrare brevemente i motivi per i quali è stato creato lo standard Posix e le sue caratteristiche principali.

10. Descrivere i gruppi di permessi di un sistema Unix/Posix.

Lezione 011

01. A cosa serve una shell di Unix?

- [ ] ad avviare il sistema operativo
- [ ] a gestire le finestre
- [ ] a lanciare comandi
- [ ] a coordinare i processori del sistema operativo

02. Quale fra le seguenti NON è una shell di Unix?

- [ ] Korn shell
- [ ] C-Shell
- [ ] Batch shell
- [ ] Bourne shell

03. Qual è il formato corretto per l'esecuzione dei comandi Unix?

- [ ] comando -opzioni argomenti
- [ ] comando -opzioni
- [ ] -opzioni ? cmd1 : cmd2
- [ ] argomenti > comando (-opzioni)

04. Qual è il ruolo del comando man nel sistema operativo Unix?

- [ ] è un comando che descrive altri comandi
- [ ] è un software applicativo generico per l'esecuzione di comandi
- [ ] serve ad aiutare nella digitazione dei comandi
- [ ] è il Manual Assisted Name di sistema

05. Cosa viene riportato nella pagina del manuale di un comando invocata con man?

- [ ] il nome, la sinossi e una descrizione del comando
- [ ] lo spazio su disco occupato dal comando
- [ ] la lista di file contenuti della directory del comando
- [ ] il manuale del sistema operativo Unix

06. A cosa serve una pipe di Unix?

- [ ] ad ordinare alfabeticamente linee di testo in unput
- [ ] a fare ricerche nei file di testo
- [ ] a concatenare comandi
- [ ] a contare il numero di righe, di caratteri e di byte contenuti in un file

07. Qual è la funzione più usata del comando awk di Unix?

- [ ] invertire l'ordine delle righe di un file
- [ ] selezionare le righe di un file
- [ ] contare il numero di righe, di caratteri e di byte di un file
- [ ] selezionare e manipolare le colonne di un file

08. A che cosa NON serve la redirezione di input e output di Unix?

- [ ] a salvare ciò che viene scritto da un comando su un file
- [ ] ad usare il contenuto di un file come input di un comando
- [ ] a migliorare la velocità di esecuzione di un comando
- [ ] a salvare su un file il flusso dati proveniente da un dispositivo

09. Descrivere che cos'è e a cosa serve la shell di Unix.

10. Dare una descrizione e qualche esempio del comando man di Unix.

11. Descrivere che cos'è e a cosa serve la pipe di Unix.

Lezione 012

01. Per esprimere numeri in base 8, di quali cifre disponiamo?

- [ ] dalla cifra 1 alla cifra 8
- [ ] dalla cifra 0 alla cifra 9
- [ ] dalla cifra 0 alla cifra 8
- [x] dalla cifra 0 alla cifra 7

02. Se al numero 111 espresso in base 2, si aggiunge 1, quale sarà il risultato?

- [x] un numero con quattro cifre in base 2
- [ ] il numero 111 e riporto di 1 in base 2
- [ ] il numero 112 in base 2
- [ ] il numero 112 in base 10

03. Perchè il calcolatore usa il sistema binario che ha solo due cifre?

- [ ] perchè è il sistema più veloce eseguire i calcoli
- [x] perchè è molto efficiente a rilevare la presenza o l'assenza di tensione, cui le due cifre si riferiscono
- [ ] perchè con meno cifre la rappresentazione dei numeri è più efficiente
- [ ] perchè è necessario meno spazio per rappresentare numeri binari

04. Quale fra le seguenti frasi è vera?

- [ ] il numero 100 non esiste in base 2
- [ ] il numero 100 ha lo stesso valore sia in base 10 che in base 2
- [ ] il numero 100 esiste in base 2 e in base 10, ma non in base 8
- [x] il numero 100 espresso in base 2, vale 4 in base 10

05. In quali fra le seguenti basi di numerazione il numero 256, espresso in base 10, rimane valido?

- [ ] in base 2
- [ ] nelle basi 2, 8 e 10
- [ ] solo in base 10
- [x] nelle basi 8, 10 e 16

06. Il numero 111 in base 2 a quale numero in base 10 corrisponde?

- [ ] 8
- [x] 7
- [ ] 13
- [ ] 111

07. Il numero 27143A in quali basi di numerazioni potrebbe essere espresso?

- [x] in base 16 o in base 36
- [ ] solo in base 16
- [ ] in base 8 e in base 10
- [ ] in base 10 e in base 16

08. Quanti oggetti posso etichettare avendo a disposizione 4 bit?

- [ ] 32 oggetti
- [ ] 16 oggetti
- [ ] 4 oggetti
- [ ] 8 oggetti

09. Di quanti bit ho bisogno per etichettare 3 oggetti?

- [ ] mi basta 1 bit
- [ ] mi servono almeno 3 bit
- [ ] mi servono più di 3 bit
- [ ] mi servono almeno 2 bit

10. A cosa serve il complemento a due?

- [ ] a trasformare i numeri positivi in numeri negativi
- [ ] a rendere leggibili i numeri binari con segno
- [ ] a rendere più efficienti i calcoli con i numeri negativi
- [ ] a rappresentare i numeri con segno in formato binario

11. Per quale motivo si usa il complemento a due?

- [ ] per velocizzare i calcoli
- [ ] per ottimizzare la circuiteria della CPU
- [ ] per evitare di usare i numeri negativi al calcolatore
- [ ] per poter rendere complementari i numeri positivi e negativi

12. Descrivere la procedura per trasformare un numero positivo nel suo corrispondente numero negativo quando esso è rappresentato in formato binario in complemento a due. Proporre un esempio di tale trasformazione.

13. Si calcoli la somma fra numeri binari 1001+0011 discutendo i passaggi. Si converta poi il risultato in base 10.

14. Dato il numero 35, rappresentato in base 10, illustrare i passaggi per la conversione in base 2. Una volta convertito nella sua rappresentazione binaria, trasformarlo nella sua controparte negativa (ovvero -35) usando la rappresentazione in complemento a due.

15. Calcolare, illustrando tutti i passaggi, quanto vale in base 10 il numero binario 10110011.

Lezione 013

01. Qual è la caratteristica errata riferita alla programmazione dichiarativa?

- [ ] descrive un problema definendo in modo dettagliato i passi per risolverlo
- [ ] descrive un problema in termini di ciò che si vuole ottenere
- [ ] di solito non permette l'acceso diretto all'hardware
- [ ] permette di programmare tramite vincoli e funzioni

02. Qual è la caratteristica errata riferita alla programmazione imperativa?

- [ ] permette di decidere cosa fare nel codice in dipendenza di certe condizioni
- [ ] permette di progettare la soluzione a un problema in termini di ciò che si vuole ottenere
- [ ] permette la definizione di algoritmi tramite passi ben definiti
- [ ] permette una gestione diretta delle risorse hardware della macchina

03. Quali tra le seguenti frasi è corretta relativamente ai Linguaggi Compilati?

- [ ] Risulta necessario avere un compilatore per ogni hardware
- [ ] Basta avere un interprete per eseguire il programma
- [ ] Sono lontani dall'architettura
- [ ] Sono poco efficienti

04. Quali tra le seguenti frasi è corretta relativamente ai Linguaggi Interpretati?

- [ ] Sono vicini all'architettura hardware e al suo linguaggio macchina
- [ ] Lo stesso codice funziona su più architetture hardware
- [ ] Sono molto efficienti
- [ ] Prima di poter essere eseguito il codice deve essere compilato

05. Quali sono le fasi del processo di compilazione?

- [ ] login, interpretazione dei comandi, esecuzione dei comandi, pulizia, uscita
- [ ] analisi lessicale, sintattica, semantica, ottimizzazione, generazione file eseguibile
- [ ] scrittura del codice, debug, correzione, salvataggio
- [ ] importazione librerie, codice nativo, traduzione, esecuzione

06. Quali tipi di variabili esistono in Java?

- [ ] i tipi possono essere definiti dal programmatore
- [ ] esistono solo i tipi primitivi
- [ ] esistono solo i tipi definiti dal programmatore
- [ ] i tipi sono stati definiti dai progettisti del linguaggio

07. Cosa è corretto affermare in merito al tipo associato alle variabili in Java?

- [ ] il tipo viene scelto quando si dichiara la variabile e può essere cambiato
- [ ] il tipo è obbligatorio e immutabile
- [ ] il tipo della variabile cambia durante il programma a seconda del contenuto
- [ ] il tipo associato a una variabile è opzionale

08. Quali fra le seguenti caratteristiche non fa parte di quelle di Java?

- [ ] linguaggio a oggetti
- [ ] compilatore just-in-time
- [ ] gestione diretta delle risorse hardware
- [ ] portabilità del codice

09. Come devono essere scelti gli identificatori delle variabili in Java?

- [ ] devono iniziare con una cifra o con una lettera
- [ ] devono avere valore numerico
- [ ] devono essere numeri interi
- [ ] devono iniziare con "$", "_" o con una lettera

10. Quali fra le seguenti caratteristiche è sicuramente riferita a Java?

- [ ] non è un linguaggio a oggetti
- [ ] è un linguaggio non portabile
- [ ] è un linguaggio case-sensitive
- [ ] è un linguaggio case-insensitive

11. Descrivere il processo di compilazione di un programma Java.

12. Date le regole di produzione:
A->B|topolino
B->CB|pippo
C->pluto
Mostrare almeno tre esempi di frasi ben formate.

Lezione 014

01. Quale fra i seguenti non è un tipo base di Java?

- [ ] char
- [ ] int
- [ ] double
- [ ] string

02. Quale fra le seguenti affermazioni è falsa se riferita al linguaggio Java?

- [ ] Il tipo string non è un tipo base
- [ ] Il dominio del tipo int contiene il dominio del tipo byte
- [ ] Il tipo String è un tipo base
- [ ] Il tipo di una variabile non è immutabile

03. Quale fra le seguenti relazioni è errata se riferita ai domini dei tipi interi?

- [ ] dominio(long) > dominio(int) > dominio(short)
- [ ] dominio(long) > dominio(byte) > dominio(short)
- [ ] dominio(int) > dominio(short) > dominio(byte)
- [ ] dominio(byte) <dominio(short) <dominio(long)

04. Quale fra le seguenti affermazioni è falsa se riferita al linguaggio Java?

- [ ] Gli unici tipi ammessi per le variabili in Java sono soltanto i tipi base
- [ ] Dichiarare una variabile indica al compilatore di riservare un'area di memoria
- [ ] Il metodo main è obbligatorio in ogni programma Java eseguibile
- [ ] Le variabili in Java hanno associato un identificatore ed un tipo

05. Quale fra le seguenti affermazioni è falsa se riferita alla dichiarazione delle variabili?

- [ ] In Java non è possibile utilizzare variabili senza dichiararle
- [ ] Le variabili in Java hanno associato un solo identificatore e un solo tipo
- [ ] Dichiarare una variabile indica al compilatore di riservare un'area di memoria
- [ ] In Java è possibile utilizzare variabili senza dichiararle

06. Quale fra le seguenti affermazioni è corretta se alla dichiarazione delle variabili?

- [ ] In Java è possibile utilizzare variabili senza dichiararle
- [ ] Il tipo di una variabile Java va scelto esclusivamente fra i tipi base
- [ ] Il tipo di una variabile Java è immutabile
- [ ] In Java il tipo di una variabile è opzionale

07. Quale fra le seguenti affermazioni è falsa se riferita ai tipi base di Java?

- [ ] una variabile di tipo float memorizza numeri in virgola mobile
- [ ] una variabile di tipo byte non può memorizzare valori più grandi di 127
- [ ] una variabile di tipo boolean può memorizzase soltanto due valori
- [ ] una variabile di tipo float memorizza numeri in virgola fissa

08. Quale fra le seguenti affermazioni è falsa se riferita alle variabili Java?

- [ ] Le variabili, dopo essere state dichiarate, devono essere inizializzate
- [ ] Le variabili dichiarate ma non inizializzate generano un errore in fase di compilazione
- [ ] E' obbligatorio inizializzare tutte le variabili prima di usarle
- [ ] Le variabili, dopo essere state dichiarate, possono essere inizializzate

09. Descrivere i motivi per i quali è necessario dichiarare le variabili.

10. Elencare i tipi primitivi di Java e il loro dominio.

11. Mettere in relazione fra di loro i domini dei tipi base interi di Java.

Lezione 019

01. Il campo MAGIC di un file bytecode quale valore contiene?

- [ ] 0xCAFEBABE
- [ ] 0xCOKEBABE
- [ ] 0xJAVABABE
- [ ] 0xMILKBABE

02. Un file .class:

- [ ] esclude le classi di un programma in codice Java
- [ ] include le classi di un programma in codice Java
- [ ] è un file binario il risultato della compilazione
- [ ] è il risultato della scrittura del codice di una classe in Java

03. Il formato binario adottato da Java per la rappresentazione dei valori in memoria è:

- [ ] Big-endian
- [ ] Medium-endian
- [ ] Little-endian
- [ ] Short-endian

04. A cosa serve il Just-in-Time compiler?

- [ ] A compilare variabili molto grandi
- [ ] A liberare la memoria
- [ ] A compilare "al volo" i cicli in codice binario nativo
- [ ] Ad eseguire classi di grandi dimensioni

05. In quale struttura dati della Java Virtual Machine vengono salvati i valori stringa?

- [ ] Nella tabella stack_pool
- [ ] Nella tabella heap_poll
- [ ] Nella tabella constant_pool
- [ ] Nella memoria RAM

06. Il tipo B nel file .class a quale tipo base corrisponde?

- [ ] int
- [ ] byte
- [ ] double
- [ ] char

07. Cosa significa il termine JVM?

- [ ] Java Very Machine
- [ ] Java Virtual Mobile
- [ ] Java View Machine
- [ ] Java Virtual Machine

08. In quale fase interviene il Just-in-Time compiler?

- [ ] A tempo di esecuzione
- [ ] Durante la scrittura del programma
- [ ] A tempo di compilazione
- [ ] Prima della compilazione del codice

Lezione 020

01. Cosa fa System.out.println()?

- [ ] trasforma una variabile in stringa
- [ ] scrive del testo e resta sulla stessa linea
- [ ] legge una variabile dalla tastiera
- [ ] scrive del testo e va a capo

02. Cosa fa System.out.print()?

- [ ] scrive del testo e va a capo
- [ ] trasforma una variabile in stringa
- [ ] scrive del testo e resta sulla stessa linea
- [ ] legge una variabile dalla tastiera

03. A cosa serve System.in?

- [ ] A leggere dalla memoria
- [ ] A scrivere sulla memoria
- [ ] A leggere da un nastro
- [ ] A leggere un flusso di byte dallo standard input

04. Come si fa a leggere in modo semplice dalla tastiera in Java?

- [ ] Usando il metodo System.in.println()
- [ ] Usando direttamente System.in
- [ ] Usando la classe java.util.Scanner
- [ ] Usando il metodo System.in.readln()

05. A cosa servono le classi BufferedReader e InputStreamReader?

- [ ] A leggere stringhe dal video
- [ ] A scrivere stringhe sulla tastiera
- [ ] A scrivere stringhe sul video
- [ ] A leggere stringhe dalla tastiera

06. Quale stringa di formato possiamo usare per troncare un numero double alla seconda cifra dopo la virgola?

- [ ] "%2f"
- [ ] "%.2"
- [ ] "%2.f"
- [ ] "%.2f"

07. Cosa fa l'istruzione Java String.format("%.3f", 3.1416)?

- [ ] Restituisce la stringa "3.141"
- [ ] Restituisce la stringa "%"
- [ ] Restituisce 3.1416 % 3
- [ ] Restituisce la stringa "%.3f"

08. Come si legge una stringa dalla tastiera utilizzando java.util.Scanner?

- [ ] Con il metodo printLine()
- [ ] Con il metodo println()
- [ ] Con il metodo readln()
- [ ] Con il metodo nextLine()

Lezione 021

01. Quale frase è vera se riferita all'istruzione switch-case?

- [ ] l'espressione dello switch può essere un numero o una stringa
- [ ] lo switch contiene sempre un blocco di codice
- [ ] tutti i rami case terminano con una istruzione break
- [ ] i tipi ammessi per l'espressione dello switch sono solo int e byte

02. Quale fra le seguenti frasi è falsa se riferita alle istruzioni if-else e switch-case?

- [ ] switch-case ed if-else sono costrutti sintattici alternativi
- [ ] si preferisce usare switch-case al posto di if-else quando i casi da controllare sono tanti
- [ ] l'uso di switch-case permette di rendere più evidenti le scelte del programmatore
- [ ] if-else è più efficiente di switch-case ma quest'ultima è più compatta

03. Quale frase è vera se riferita all'istruzione switch-case?

- [ ] al termine di un ramo case si esce sempre dallo switch
- [ ] in assenza di break, al termine di un ramo case viene eseguito il successivo case
- [ ] se un ramo case non termina con break, il compilatore genera un errore
- [ ] se il ramo default non è presente, il compilatore genera un errore

04. Quale frase è vera se riferita all'istruzione switch-case?

- [ ] l'espressione dello switch può essere un numero o una stringa
- [ ] i tipi ammessi per l'espressione dello switch sono solo int e byte
- [ ] il ramo default è opzionale
- [ ] tutti i rami case terminano con una istruzione break

05. Quale frase è falsa se riferita all'istruzione switch-case?

- [ ] il ramo default è opzionale
- [ ] se il ramo default non è presente, il compilatore genera un errore
- [ ] in assenza di break, al termine di un ramo case viene eseguito il successivo case
- [ ] lo switch contiene sempre un blocco di codice

06. Quale frase è vera se riferita all'istruzione if-else?

- [ ] una istruzione if è sempre seguita da un ramo else
- [ ] se il ramo else è semplice (ovvero senza condizione) viene sempre eseguito
- [ ] il ramo else viene eseguito se la condizione dell'if è vera
- [ ] un blocco di codice è sempre neccessario per eseguire più di una istruzione

07. Quale frase è vera se riferita all'istruzione if-else?

- [ ] un else si riferisce sempre all'if più vicino che lo segue
- [ ] un else si riferisce sempre all'if più vicino che lo precede
- [ ] il ramo else viene eseguito se e solo se la condizione dell'if è vera
- [ ] un else si riferisce sempre all'if più vicino, tranne in presenza di blocchi di codice

08. Dopo aver eseguito "if (5<5) i=0; else i=5;", quanto vale i?

- [ ] i ha un valore indeterminato
- [ ] i vale 5
- [ ] i vale 0
- [ ] viene generato un errore dal compilatore

09. Descrivere le principali differenze tra le istruzioni if-else e switch-case ed illustrarle con degli esempi.

10. Scrivere il codice di una istruzione switch-case che converta ogni cifra della base 16 nel corrispondente valore in base 10 stampandolo a video. Si gestiscano anche, con un opportuno messaggio di errore, eventuali cifre non appartenenti alla base 16.

Lezione 022

01. Che tipo di operatore è un operatore unario?

- [ ] Opera su una singola variabile
- [ ] Somma valori unitari
- [ ] Somma valori uno alla volta
- [ ] Restituisce uno

02. Che tipo di operatore è l'operatore autoincremento?

- [ ] Operatore unario
- [ ] Operatore bit a bit
- [ ] Operatore booleanto
- [ ] Operatore di differenza

03. Cosa è falso se riferito un operatore binario?

- [ ] La somma è un operatore binario
- [ ] Restituisce due
- [ ] E' sempre commutativo
- [ ] Opera su coppie di operandi

04. Cosa è falso se riferito un operatore unario?

- [ ] L'operatore not è un operatore unario
- [ ] L'operatore di autoincremento è un operatore unario
- [ ] Non opera su coppie di operandi
- [ ] Restituisce uno

05. Qual'è la definizione corretta di AND logico?

- [ ] Verifica se due operandi hanno lo stesso valore intero
- [ ] Vero se entrambi gli operandi sono veri
- [ ] Congiunzione tra caratteri
- [ ] Vero solo se due numeri hanno la virgola

06. Qual'è la definizione corretta di OR logico?

- [ ] Verifica solo se gli operandi hanno lo stesso segno
- [ ] Vero se almeno uno degli operandi è vero
- [ ] Disgiunzione solo tra caratteri
- [ ] Vero solo se uno dei due operandi sono vuoti

07. Cosa è vero se riferito all'operatore modulo?

- [ ] Verifica che il denominatore della divisione non sia zero
- [ ] Calcola il quoziente e il resto della divisione intera fra due numeri
- [ ] Viene utilizzato per testare se un numero è dispari
- [ ] Calcola il quoziente della divisione intera fra due numeri

08. Quale fra questi operatori logici calcola la disgiunzione?

- [ ] OR
- [ ] XOR
- [ ] AND short circuit
- [ ] NOT

09. Descrivere gli operatori relazionali.

10. Descrivere gli operatori booleani e le loro precedenze.

Lezione 023

01. Che fa l'operazione "a=b"?

- [ ] Ritorna vero se e solo se a e b hanno lo stesso valore
- [ ] Scrive il valore di b in a
- [ ] Nulla, a meno che a e b abbiano lo stesso valore
- [ ] Scrive il valore di a in b

02. Cosa fa l'istruzione "int a = 0;"?

- [ ] Verifica se a è un intero, nel qual caso lo inizializza a zero
- [ ] Definisce una variabile di zero bit
- [ ] Definisce un vettore di zero elementi
- [ ] Dice al compilatore di allocare una variabile intera contenente il valore zero

03. A cosa serve l'istruzione "a==b"?

- [ ] Assegna b ad a
- [ ] Assegna a b il doppio di a
- [ ] Verifica se a è uguale a b
- [ ] Assegna ad a il doppio di b

04. Cosa è falso se riferito alle variabili Java?

- [ ] Il nome e il dominio della variabile sono immutabili
- [ ] Il tipo di una variabile, una volta definito, non si può più cambiare
- [ ] Durante la vita di una variabile è possibile assegnarle un nuovo valore e un nuovo tipo
- [ ] Due variabili dello stesso tipo occupano lo stesso spazio di memoria

05. Quale fra le seguenti frasi è vera se riferita alle variabili in Java?

- [ ] La dichiarazione delle variabili è facoltativa perchè il compilatore è capace di verificare automaticamente il codice
- [ ] La dichiarazione delle variabili avviene sempre all'inizio di un blocco di codice
- [ ] La dichiarazione delle variabili è obbligatoria e ciò permette al compilatore di effettuare la validità del codice
- [ ] La dichiarazione delle variabili è quasi sempre obbligatoria tranne per le conversioni automatiche di tipo

06. Quale fra le seguenti frasi è vera se riferita alle variabili in Java?

- [ ] Il nome della variabile deve essere unico nel blocco di codice considerato
- [ ] Il tipo di una variabile deve essere scelto fra i tipi base
- [ ] Il valore iniziale della variabile deve essere unico nel blocco di codice considerato
- [ ] Il valore della variabile deve essere unico nel blocco di codice considerato

07. Quale fra le seguenti frasi è vera se riferita alle variabili in Java?

- [ ] Dichiarare una variabile garantisce l'assegnazione di un valore di default
- [ ] Dichiarare una variabile è obbligatorio quando si usano i tipi base
- [ ] Dichiarare una variabile non è obbligatorio in Java
- [ ] Dichiarare una variabile permette al compilatore di allocare un'area di memroia sufficiente

08. Quale fra le seguenti frasi è falsa se riferita all'operatore assegnamento?

- [ ] E' un operatore binario
- [ ] A sinistra dell'assegnamento può esserci soltanto una variabile
- [ ] "boolean b = 4 <3" da errore di "incompatible types" in fase di compilazione
- [ ] La parte destra dell'assegnamento deve essere dello stesso tipo della parte sinistra

09. Descrivere l'operazione di assegnamento di una variabile.

10. Descrivere l'operazione di inizializzazione di una variabile.

Lezione 024

01. Quali tra le seguenti non è una classe di operatori?

- [ ] boolean
- [ ] logici
- [ ] aritmetici
- [ ] relazionali

02. La negazione è un operatore:

- [ ] quaternario
- [ ] unario
- [ ] binario
- [ ] ternario

03. L'assegnazione in Java si rappresenta con:

- [ ] =:
- [ ] :=
- [ ] =
- [ ] ==

04. L'operatore autoincremento è:

- [ ] +=
- [ ] ++
- [ ] +
- [ ] +1

05. Se x vale 14, l'espressione "3 * -(x - 12)" quanto vale?

- [ ] -6
- [ ] -14
- [ ] 14
- [ ] 6

06. L'espressione "a ^ b", che usa l'operatore XOR, restituisce true se:

- [ ] se a e b sono false
- [ ] se entrambi sono false
- [ ] se a oppure b vale true
- [ ] se a e b sono true

07. L'espressione "10 <100 == 5 > 3" quanto vale?

- [ ] 0
- [ ] -1
- [ ] false
- [ ] true

08. L'espressione "10 % 5" quanto vale?

- ☐ 5
- ☐ 0
- ☐ 1
- ☐ 2

09. Descrivere che cosa sono e come funzionano gli operatori con associatività a sinistra e fare almeno 3 esempi di espressoni che li utilizzano.

10. Scrivere almeno 3 esempi di espressioni logiche e descrivere come possono essere utilizzate nei programmi Java.

11. Utilizzando il codice Java scrivere un esempio di uso dell'operatore modulo per testare se una variabile intera è pari o dispari.

Lezione 025

01. "int[] a=new int[5]" cosa vuol dire?

- [ ] crea un array di 5 elementi di tipo intero
- [ ] rinnova un array di 5 elementi e li azzera
- [ ] salva il valore 0 in una variabile di 5 elementi
- [ ] crea un array di 5 elementi interi o a virgola mobile

02. Il primo elemento "dell'array int[] a" si indica come:

- [ ] a
- [ ] a[1]
- [ ] a[0]
- [ ] a[a.length]

03. Quale di queste affermazioni è vera per gli array?

- [ ] Tutti gli elementi dell'array sono dello stesso tipo
- [ ] Un array contiene uno e un solo valore
- [ ] La lunghezza dell'array è variabile
- [ ] Gli elementi dell'array possono essere di tipo diverso

04. Un indice di un array:

- [ ] Serve ad indicare un array
- [ ] Serve a scorrere gli elementi dell'array
- [ ] Serve a trovare la lunghezza dell'array
- [ ] Identifica un array

05. Come variano gli indici di un array?

- [ ] Da 1 a length-1
- [ ] Da 0 a length-1
- [ ] Da 1 a length
- [ ] Da 0 a length

06. Cosa fa a[10]=100?

- [ ] assegna il valore 10 al centesimo elemento dell'array
- [ ] assegna il valore 100 all'undicesimo elemento dell'array
- [ ] assegna il valore 100 al decimo elemento dell'array
- [ ] verifica se a[10] è uguale a 100

07. "double[] d;" cosa indica?

- [ ] un vettore che possa contenere caratteri
- [ ] un vettore che possa contenere valori doppi
- [ ] un vettore che possa contenere valori in virgola mobile
- [ ] un vettore che possa contenere stringhe

08. L'istruzione "byte[] i = {2, 3, 6, 100, 230, 340};" cosa provoca se compilata?

- [ ] Crea un array di byte di 6 elementi con i valori specificati tra graffe
- [ ] Crea un array di byte di lunghezza fissa
- [ ] Genera un errore in fase di compilazione
- [ ] Crea un array vuoto di byte di 6 elementi

Lezione 026

01. Un ciclo for esegue un numero di cicli pari a:

- [ ] un numero finito noto prima della prima istruzione
- [ ] un numero massimo pari a 100
- [ ] un numero non prevedibile
- [ ] un numero infinito

02. Quale fra i seguenti cicli for provoca un errore di compilazione?

- [ ] for (;)
- [ ] for (;;)
- [ ] for (;i<5;i++)
- [ ] for (;i<5;)

03. Dato l'array "char[] a={'1','2','3'}" come possiamo stamparlo invertito con un ciclo for?

- [ ] for (int i=a.length-1;i>=0;i--) System.out.print(a[i]);
- [ ] for (int i=0;i
- [ ] for (int i=0;i
- [ ] for (int i=a.length;i>=0;i--) System.out.print(a[i]);

04. Nell'istruzione "for (int a=0; a<100; a++)" che valori assume la a?

- [ ] Da 1 a 99
- [ ] Da 1 a 100
- [ ] Da 0 a 99
- [ ] Da 0 a 100

05. Nell'istruzione "for (int a=0; a<100; a++)" la variabile a a quale dominio di definizione appartiene?

- [ ] virgola mobile
- [ ] booleano
- [ ] infinito
- [ ] intero

06. A cosa serve il ciclo for?

- [ ] Ad eseguire una o più istruzioni almeno una volta
- [ ] Ad eseguire una o più istruzioni in un certo arco di tempo
- [ ] Ad eseguire una o più istruzioni più di una volta
- [ ] Ad eseguire una sola istruzione più volte

07. Quale fra le seguenti sintassi del for è la più corretta?

- [ ] for (inizializzazione; valore-booleano) istruzioni;
- [ ] for (inizializzazione; valore-booleano; incremento) istruzioni;
- [ ] for (inizializzazione; valore-booleano);
- [ ] for (inizializzazione; valore-booleano; incremento);

08. Quale fra i seguenti cicli for è equivalente al ciclo "while (true)"?

- [ ] for (i=0;i
- [ ] for (i=10;i<1;i++)
- [ ] for (;i
- [ ] for (;;)

09. Scrivere un programma Java che inverta con un ciclo for l'array di caratteri "char[] a = {'p', 'i', 'p', 'p', 'o', '.'}" e che lo stampi a video.

10. Utilizzando il ciclo for scrivere un programma Java che stampi il doppio di ogni numero naturale compreso fra 1 e 100. Ad esempio "2 4 6 8 10 12 ... ecc".

Lezione 027

01. Se una istruzione di un ciclo può essere eseguita 0 volte:

- [ ] E' possibile usare il do-while solo con condizioni booleane
- [ ] E' possibile usare il do-while
- [ ] Si può usare il while o il for
- [ ] Non è mai possibile usare il do-while

02. E' sempre possibile che do-while sia equivalente ad un ciclo for?

- [ ] No, perchè il for non è mai equivalente ad un do-while
- [ ] No, perchè il do-while non è un ciclo
- [ ] No, perchè il do-while viene eseguito almeno una volta
- [ ] Si, perchè il do-while viene eseguito almeno una volta

03. Quando si utilizza l'istruzione do-while?

- [ ] Quando ci serve eseguire le istruzioni del ciclo almeno una volta
- [ ] Quando ci serve eseguire le istruzioni del ciclo una e una sola volta
- [ ] Quando ci serve eseguire le istruzioni del ciclo una sola volta
- [ ] Quando ci serve eseguire le istruzioni del ciclo in ordine inverso

04. La sintassi corretta dell'istruzione do-while è:

- [ ] do while (condizione) istruzione/blocco-di-istruzioni;
- [ ] do istruzione/blocco-di-istruzioni while (condizione);
- [ ] do (condizione) while istruzione/blocco-di-istruzioni;
- [ ] while (condizione) do istruzione/blocco-di-istruzioni;

05. Se i=1, l'istruzione "do {fat = fat * i; i = i + 1;} while (i <= n)" esegue:

- [ ] La moltiplicazione di fat*i esattamente n volte
- [ ] La moltiplicazione di fat*i almeno n-1 volte
- [ ] La moltiplicazione di fat*i esattamente n-i volte
- [ ] La moltiplicazione di fat*i 0 volte

06. Se i=1, l'istruzione "do {fat = fat * i; i = i + 1;} while (i <n)" esegue:

- [ ] La moltiplicazione di fat*i almeno n volte
- [ ] La moltiplicazione di fat*i 0 volte
- [ ] La moltiplicazione di fat*i esattamente n-i volte
- [ ] La moltiplicazione di fat*i esattamente n-1 volte

07. Se i=n, l'istruzione "do {fat = fat * i; i = i + 1;} while (i <= n)" esegue:

- [ ] La moltiplicazione di fat*i almeno n-1 volte
- [ ] La moltiplicazione di fat*i esattamente 1 volta
- [ ] La moltiplicazione di fat*i esattamente n-i volte
- [ ] La moltiplicazione di fat*i 0 volte

08. Se una istruzione di un ciclo può essere eseguita 0 volte:

- [ ] Si può sempre usare il do-while
- [ ] E' possibile usare il do-while
- [ ] Non è mai possibile usare il do-while
- [ ] E' possibile usare il do-while solo con condizioni booleane

Lezione 028

01. Cosa fa una istruzione break?

- [ ] Salta una iterazione del ciclo senza uscire
- [ ] Riavvia il sistema operativo
- [ ] Blocca ed esce dal programma
- [ ] Interrompe il ciclo e riporta alla prima istruzione dopo il blocco

02. Cosa fa una istruzione continue?

- [ ] Interrompe il ciclo e riporta alla prima istruzione dopo il blocco
- [ ] Riavvia il sistema operativo
- [ ] Blocca ed esce dal programma
- [ ] Salta una iterazione del ciclo senza uscire

03. Dove si usa il comando break?

- [ ] Solo nell'if
- [ ] In un ciclo o in uno switch/case
- [ ] Solo nei cicli
- [ ] Solo nello switch/case

04. Dove si usa il comando default?

- [ ] Solo nello switch/case
- [ ] Solo nei cicli
- [ ] In un ciclo o in uno switch/case
- [ ] Solo nell'if

05. Quando è possibile usare un comando di break?

- [ ] Solo in caso di assegnazione di variabile
- [ ] Anche all'interno di un do-while
- [ ] Solo nel blocco else
- [ ] Solo all'interno di un do-while

06. Quando è possibile usare un comando di continue?

- [ ] Anche all'interno di un for
- [ ] Solo in caso di assegnazione di variabile
- [ ] Solo all'interno di un for
- [ ] Solo nel blocco else

07. Quante volte viene eseguito il seguente ciclo : "for (i=0; i<100; i++) { break; }"?

- [ ] Viene eseguito 1 volta
- [ ] Viene eseguito 99 volte
- [ ] Viene eseguito 100 volte
- [ ] Viene eseguito 0 volte

08. Quante volte viene eseguito il seguente ciclo : "for (i=0; i<100; i++) { continue; }"?

- [ ] Viene eseguito 0 volte
- [ ] Viene eseguito 100 volte
- [ ] Viene eseguito 1 volta
- [ ] Viene eseguito 99 volte

Lezione 029

01. Usando quale proprietà degli array si ottiene il conteggio dei suoi elementi?

- [ ] non si può ottenere
- [ ] size
- [ ] length
- [ ] dimensions

02. Quale fra le seguenti frasi è falsa se riferita alla proprietà length degli array?

- [ ] serve a sapere il numero totale dei suoi elementi
- [ ] serve a sapere quanti elementi sono stati allocati in memoria
- [ ] serve a sapere quanti elementi sono occupati
- [ ] serve a sapere quanti elementi compongono il vettore

03. Per copiare nel modo più efficiente possibile un array in un altro usiamo:

- [ ] la funzione arraycopy della classe System
- [ ] un doppio ciclo for
- [ ] la funzione copyarray della classe System
- [ ] un ciclo for

04. Quale fra le seguenti frasi è falsa se riferita allo scambio di due elementi di un array?

- [ ] per effettuare lo scambio abbiamo bisogno di un vettore di appoggio
- [ ] per effettuare lo scambio abbiamo bisogno di una variabile temporanea
- [ ] per effettuare lo scambio basta assegnare un elemento dell'array all'altro e viceversa
- [ ] per effettuaro lo scambio basta usare una funzione predefinita di Java

05. Quale fra le seguenti frasi è falsa se riferita all'inversione degli elementi di un array?

- [ ] non abbiamo bisogno di memoria aggiuntiva, oltre quella già allocata per l'array
- [ ] possiamo utilizzare una variabile temporanea
- [ ] possiamo utilizzare un array di appoggio
- [ ] possiamo utilizzare due indici

06. Quale fra le seguenti frasi è falsa se riferita al metodo System.arraycopy()?

- [ ] Il metodo ha cinque parametri
- [ ] Il metodo permette la copia di elementi fra array di lunghezze diverse
- [ ] Il metodo implementa una copia efficiente fra array
- [ ] Il metodo ha tre parametri

07. Quale fra le seguenti frasi è corretta se riferita al metodo System.arraycopy()?

- [ ] Il metodo trova gli elementi in comune fra due array
- [ ] Il metodo copia gli elementi di un array in un altro
- [ ] Il metodo accetta tre parametri
- [ ] Il metodo copia due array con la stessa efficienza di un ciclo for

08. Supponendo di aver dichiarato un array "int[] a = new int[5]", quale fra le seguenti frasi è falsa?

- [ ] nell'array a possiamo inserire elementi con valore fino a 5
- [ ] nell'array a possiamo inserire elementi fino a un massimo di 5
- [ ] gli elementi di a hanno indici che vanno da 0 a 4
- [ ] l'ultimo elemento dell'array ha indice pari a 4

Lezione 032

01. Cosa indica "nomi[0][1]"?

- [ ] l'elemento dell'array nomi uguale a 01
- [ ] l'elemento dell'array nomi in prima riga
- [ ] l'elemento dell'array non è valido perchè la riga 0 non esiste
- [ ] l'elemento dell'array nomi in prima riga e seconda colonna

02. La seguente struttura "String[][] nomi" cosa rappresenta?

- [ ] una struttura dati per contenere un numero pari di nomi
- [ ] una struttura dati per contenere un nome
- [ ] un insieme di nomi
- [ ] un array di 10 stringhe

03. Quale fra le seguenti frasi è falsa se riferita agli array in Java?

- [ ] In Java gli indici degli array partono sempre da 0
- [ ] In Java si possono creare solo array monodimensionali
- [ ] In Java gli array multidimensionali aggregano solo elementi dello stesso tipo base
- [ ] In Java si possono creare array di qualsiasi dimensione

04. Quale fra le seguenti frasi è falsa se riferita alla matrice "int[][] matrix = {{1,2},{3,4}}"?

- [ ] la matrice è 3x3
- [ ] il valore 2 è in posizione matrix[0][1]
- [ ] il valore 3 è in posizione matrix[1][0]
- [ ] la matrice è bidimensionale

05. Data l'istruzione "float[][][] m = new float[5][5][5]" quante variabili vengono allocate in memoria?

- [ ] 125
- [ ] 5
- [ ] 25
- [ ] 15

06. L'elemento di un array indicato con "a[1][2]" cosa indica?

- [ ] indica l'elemento di valore 12
- [ ] indica l'elemento in riga 2, colonna 1
- [ ] indica l'elemento in prima riga, seconda colonna
- [ ] indica l'elemento in seconda riga, terza colonna

07. Quale fra le seguenti frasi è falsa se riferita agli array in Java?

- [ ] un array contiene sempre righe della stessa lunghezza
- [ ] un array può contenere righe di lunghezze diverse
- [ ] i componenti di un array multidimensionale sono a loro volta array
- [ ] la lunghezza di un array si ottiene con la proprietà length

08. Una matrice 5x2 in Java si definisce come:

- ☐ int[][] = new int[5][2]
- ☐ int[][] = new int[5,2]
- ☐ int[][][] = new int[2][5]
- ☐ int[][] = new int[2][5]

Lezione 033

01. Data una matrice r[][] di interi, l'operazione "System.out.print(r[i][j])":

- [ ] Stampa il valore contenuto nella cella in colonna i e riga j di r
- [ ] Stampa la dimensione delle matrice (ovvero il valore di "i" e il valore di "j")
- [ ] Stampa tutti i valori della matrice per ogni posizione i,j di r
- [ ] Stampa il valore contenuto nella cella in riga i e colonna j di r

02. La seguente istruzione: "float[][] g = new float[4][7]" esegue il seguente comando:

- [ ] Crea e istanzia una matrice 4x7 per numeri a virgola mobile
- [ ] Crea e istanzia una matrice di 7 righe e 4 colonne
- [ ] Crea e istanzia un array di 28 elementi
- [ ] Crea e istanzia una matrice 4x7 di valori verità

03. La seguente istruzione: "v[0] *= 121" ha il seguente significato:

- [ ] Prende il valore contenuto nella cella 0 dell'array v, lo moltiplica per 121 e il risultato lo rimette in v
- [ ] Prende il valore contenuto nella cella 0 e ci copia 121
- [ ] Prende il valore della cella 0 dell'array v e ci mette un asterisco
- [ ] Prende 121 lo moltiplica per 0 e lo inserisce nella posizione v[0]

04. Quale di queste affermazioni per un array "double[] a" è falsa?

- [ ] La variabile "a" indica un array e può contenere anche valori interi
- [ ] La variabile "a" indica un array di valori numerici in virgola mobile
- [ ] La variabile "a" è una matrice
- [ ] La variabile "a" indica un array

05. Quale di queste affermazioni per un array "double[] a" è vera?

- [ ] La variabile "a" indica un array doppio
- [ ] La variabile "a" indica un array e può contenere valori di verità
- [ ] La variabile "a" indica un array non ancora allocato in memoria
- [ ] La variabile "a" indica un array di lunghezza arbitraria

06. Per moltiplicare una matrice m per un array v:

- [ ] La dimensione della matrice in posizione 0 (m[0].length) deve esser maggiore di v.length
- [ ] La dimensione dell'array deve essere uguale alla dimensione dell'array in posizione 0 ovvero m[0].length deve esser uguale a v.lenght
- [ ] La dimensione della matrice in posizione 1 (m[1].length) deve esser uguale a v.length
- [ ] La dimensione dell'array deve essere maggiore di 1 della dimensione dell'array in posizione 0 ovvero m[0].length deve esser uguale a v.length

07. Date due matrici "double[][] a = {{1,2}, {4,3}, {7,4}}" e "double[][] b = {{1,2,3}, {4,3,6}}":

- [ ] Non è possibile fare il prodotto a*b
- [ ] Si può fare solo il prodotto b*a
- [ ] Si può fare la somma a+b
- [ ] E' possibile fare il prodotto a*b

08. Dati due array "double[] a = {1,2,3}" e "double[] b = {4,5,6,7}", quale fra le seguenti affermazioni e' falsa:

- [ ] Non si può fare il prodotto scalare a*b
- [ ] E' possibile fare il prodotto a*b
- [ ] Si può stampare sia a che b
- [ ] Non è possibile fare il prodotto a*b

09. Scrivere il codice Java del prodotto di una matrice per uno scalare.

Lezione 034

01. Descrivere i tipi di passaggio di parametri nelle funzioni del linguaggio Java.

Lezione 035

01. La tecnica di programmazione ricorsiva:

- [ ] Si basa sulla tecnica di induzione
- [ ] Si basa sulla tecnica di programmazione dinamica
- [ ] Si basa sulla programmazione iterativa
- [ ] Si basa su metodi euristici

02. Dato un algoritmo matematicamente ricorsivo si dice che:

- [ ] Non è possibile stabilire se la sua implementazione iterativa sia sempre migliore di quella ricorsiva
- [ ] Non si può mai definire una implementazione iterativa
- [ ] E' sempre possibile indicare una implementazione iterativa ma mai ricorsiva
- [ ] Si può definire sempre e solo una implementazione ricorsiva

03. Dovendo implementare in Java l'algoritmo di calcolo del fattoriale di un numero contenuto in una variabile "n", quale fra queste linee di codice posso utilizzare?

- [ ] La linea di codice "return n * fact(n-1)"
- [ ] La linea di codice "return (n+1)*fact(n+1)"
- [ ] La linea di codice "return fact(n)*fact(n)"
- [ ] La linea di codice "return fact(n+1)"

04. Quale di queste affermazioni è falsa:

- [ ] La tecnica di ricorsione si usa ad esempio per il calcolo del numero di fibonacci
- [ ] La tecnica di ricorsione si può solo usare se contiene cicli (for o while)
- [ ] La tecnica di ricorsione è una funzione che contiene il nome della funzione al suo interno
- [ ] La tecnica di ricorsione si usa ad esempio per il calcolo del fattoriale di un numero

05. Data una funzione recursive(), quale delle seguenti affermazioni è certamente corretta?

- [ ] La funzione recursive() è ricorsiva perché fa parte delle librerie Java
- [ ] La funzione deve avere nel suo blocco di codice l'istruzione "recursive()" per definirsi ricorsiva
- [ ] La funzione deve includere sempre un ciclo di while per esser ricorsiva
- [ ] La funzione deve contenere una invocazione ricorsiva in un ciclo

06. Il calcolo del fattoriale di un numero intero si può:

- [ ] Calcolare solo con metodo ricorsivo
- [ ] Calcolare solo con l'ausilio del calcolo parallelo
- [ ] Calcolare solo con un ciclo di for
- [ ] Calcolare sia con la tecnica ricorsiva che tramite l'uso di cicli

07. La tecnica ricorsiva consente di invocare una stessa funzione dall'interno del codice in quanto:

- [ ] Usa lo Stack di attivazione
- [ ] Usa il calcolo parallelo
- [ ] Usa il disco rigido
- [ ] Usa la memoria a nastro

08. La Ricorsione e' una tecniche di programmazione che:

- [ ] Si utilizza quando è necessario ripetere delle azioni in modo ottimale
- [ ] Si basa sulla rincorsa del codice
- [ ] Si utilizza solo per analizzare numeri interi
- [ ] Si basa sulla invocazione di una stessa funzione dall'interno della funzione stessa

09. Scrivere il codice Java di una funzione ricorsiva che calcoli il fattoriale di un numero intero.

Lezione 036

01. Per aggiungere una stringa ad un'altra stringa in una variabile si usa:

- [ ] Il metodo "string.appendMeOnline()"
- [ ] Il metodo "StringBuffer.append(char[] c)"
- [ ] Il metodo "Char.AggangiaStringa()"
- [ ] Il metodo "String.Append(s1, s2)"

02. Data la funzione "StringBuffer insert(int offset, long l)" consente di:

- [ ] Inserire un nuovo valore di offset su una stringa s
- [ ] Cancellare una stringa di lunghezza l da un una posizione "offset"
- [ ] Inserire la rappresentazione stringa dell'argomento long nella sequenza di caratteri
- [ ] Stampare a video una parola inserita in una stringa

03. Data una variabile sb di tipo StringBuffer, con valore: sb = "Prova "; la funzione sb.append("test"):

- [ ] Appende uno spazio vuoto alla fine di sb (risultato sb = "Prova ")
- [ ] Appende un carattere alla variabile sb (risultato sb = "Prova t")
- [ ] Appende un testo alla variabile sb (risultato sb = "Prova test")
- [ ] Appende la parola append alla variabile sb (risultato sb = "Prova append")

04. Data l'istruzione: String s = new String("test"); la variabile s consente di:

- [ ] Definire un insieme di caratteri stampabili singolarmente
- [ ] Memorizzare solo bytecode
- [ ] Memorizzare in s valori numerici
- [ ] Stampare il contenuto della variabile s

05. Se serve per una stringa manipolare i singoli caratteri è necessario:

- [ ] Si usa un array di caratteri della forma char[]
- [ ] Si usa una tecnica di mappaggio delle stringhe in interi
- [ ] Non si può implementare mai in Java
- [ ] Si definisce una funzione dedicata che usi i double

06. Data l'istruzione: String saluto = "Ciao a tutti"; quale di queste affermazioni è vera:

- [ ] Non si possono modificare i singoli caratteri della variabile "saluto"
- [ ] Si possono invertire o saltare i caratteri dalle stringa "saluto"
- [ ] Non si può stampare la stringa "saluto"
- [ ] Si possono cambiare i caratteri della stringa "saluto"

07. In Java le stringhe si definiscono utilizzando:

- [ ] La classe CharacaterString()
- [ ] Le funzioni char()
- [ ] Le matrici definite dagli utenti
- [ ] La classe statica definita nella libreria di Java di nome String

08. Quale di queste affermazioni è sicuramente falsa?

- [ ] Una istanza di una variabile char è stampabile a video
- [ ] Una istanza di stringa String() è stampabile a video
- [ ] Una istanza della classe String e un array di char sono esattamente uguali
- [ ] I caratteri in una istanza di String sono immutabili

09. Descrivere la differenza tra la classe String e un array di char.

10. Descrivere la differenza tra la classe String e la classe StringBuffer.

Lezione 037

01. Le due operazioni: 1) String s = new String(); e 2) String s = "";

- [ ] Sono differenti e costruiscono una stringa s con valori nulli e una stringa s senza caratteri
- [ ] Sono equivalenti e costruisono una stringa con 10 caratteri vuoti
- [ ] Sono equivalenti e costruiscono una stringa vuota
- [ ] Sono differenti e costruiscono una stringa s senza caratteri e una stringa s con valori nulli

02. Date le due istruzioni Java in sequenza: "char[] c = {‘t’, ‘e’, ‘s’, ‘t’};" e "String s = new String(c);"

- [ ] Crea una stringa s che contiene il primo carattere "t" della variabile c
- [ ] Crea una stringa s che contiene la parola "'t"e"s"t'"
- [ ] Crea una nuova stringa s che non contiene nulla
- [ ] Crea una stringa s che contiene la parola "test"

03. Data una stringa s = new String("prova");, per stampare la lettera "o" si procede con:

- [ ] System.out.println([2]);
- [ ] System.out.println([II]);
- [ ] System.out.println(s.charAt(2));
- [ ] System.out.println(s.2);

04. Per confrontare due stringhe s1 ed s2 e verificare se hanno lo stesso valore si procede con:

- [ ] L'istruzione Java "if (s1=s2)"
- [ ] L'istruzione Java "if (Equals(s1,s2))"
- [ ] L'istruzione Java "if (s1==s2)"
- [ ] L'istruzione Java "if (s1.equals(s2))"

05. Data una stringa s generata in Java come: s = new String("prova");, quale delle seguenti affermazioni è falsa?

- [ ] s2=s.substring(1,3) inserisce in s2 la stringa "rov"
- [ ] s2=s.substring(1,3) inserisce in s2 la stringa intera
- [ ] s2=s.substring(1,3) serve a copiare una parte di s in s2
- [ ] s2=s.substring(1,3) si usa per generare una sottostringa di s

06. La funzione "String concat(String s)":

- [ ] E' una funzione della classe String che serve a concatenare una stringa data con la stringa parametro s
- [ ] E' una funzione che serve a concatenare sue stringhe date in input e mette in risultato nel parametro s
- [ ] E' una procedura della classe String che serve a stampare la stringa s concatenata con se stessa
- [ ] E' una funzione che serve a comunicare una stringa s a una funzione di stampa

07. Data il seguente codice: String s1 = new String("stringa"); String s2 = new String(" di prova"); quale delle seguenti affermazioni è vera?

- [ ] System.out.println(s1.concat(s2)) stampa "stringa di prova"
- [ ] System.out.println(s1.concat(s2)) stampa " prova di stringa"
- [ ] System.out.println(s1.concat(s2)) stampa " prova"
- [ ] System.out.println(s1.concat(s2)) stampa " stringa"

08. A cosa serve la classe StringTokenizer?

- [ ] Serve ad estrarre token da una stringa
- [ ] Serve a gestire una sequenza di stringhe come interi
- [ ] Serve a fare dei cicli sulle stringhe
- [ ] Serve a generare token casuali usando le stringhe

09. Descrivere il funzionamento dei metodi toUpperCase() e toLowerCase() della classe String e fare degli esempi.

Lezione 038

01. La parola chiave "import" indica:

- [ ] Un modo per dire che una istruzione è importante
- [ ] Un modo per escludere alcune classi
- [ ] Un modo per includere le funzionalità di una classe
- [ ] Un modo per indicare l'importazione di variabili

02. La parola chiave protected si può usare:

- [ ] Nella definizione dei package per cui chiedo protezione
- [ ] Nella definizione dei metodi di una classe
- [ ] Per le funzioni invocabili solo con sistema operativo
- [ ] Nella definizione di codice antihacker

03. Quali di queste affermazioni è sicuramente falsa?

- [ ] La parola chiave "long" è un intero con segno
- [ ] La parola chiave "long" indica un tipo
- [ ] La parola chiave "long" si usa per i valori numerici
- [ ] La parola chiave "long" serve a definire stringhe più lunghe

04. La parola chiave package:

- [ ] Indica la versione dei pacchetti generati dall'utente
- [ ] Indica la versione della CPU
- [ ] Indica la versione del compilatore di Java
- [ ] Indica un pacchetto o libreria di funzioni

05. La parola chiave instanceof si utilizza per:

- [ ] Testare se un programma è istanziato
- [ ] Testare se un programma è in esecuzione
- [ ] Testare se una variabile e istanziabile
- [ ] Testare se una variabile è una istanza di un certo tipo

06. Data l'istruzione: String final = "fine"; che succede se la compiliamo?

- [ ] Genera un errore perchè final è una parola chiave
- [ ] Genera un errore perchè una stringa non si definisce così
- [ ] Non genera alcun errore
- [ ] Genera un errore perchè dopo l'uguale non ci vogliono mai le virgolette

07. Quali delle seguenti parole non è una parola chiave di Java?

- [ ] Class
- [ ] abstract
- [ ] else
- [ ] boolean

08. Per invocare un metodo di una classe padre nella gerarchia di classi Java si usa la parola chiave:

- ☐ super
- ☐ ancestor
- ☐ ancien
- ☐ overthetop

09. Descrivere che cos'è una keyword di Java e fare degli esempi di keyword.

Lezione 039

01. Data la classe "Persona", la seguente istruzione "Persona p1 = new Persona();" serve a:

- [ ] Creare un oggetto p1 che sia di tipo Persona
- [ ] Inizializzare la classe Persona
- [ ] Creare una nuova classe p1 di tipo Persona
- [ ] Creare una variabile p1 che contenga un array di persone

02. L'istruzione new serve a:

- [ ] Creare un nuovo programma Java
- [ ] Istanziare un nuovo sistema operativo
- [ ] Istanziare una nuova funzione
- [ ] Istanziare un nuovo oggetto di una classe

03. Data la classe "public class Persona { String nome; }" e una sua istanza p1, quali delle seguenti istruzioni è corretta:

- [ ] System.out.println(nome);
- [ ] System.out.println(p1->nome);
- [ ] System.out.println(p1.nome);
- [ ] System.out.println(p1);

04. Per ottenere una nuova istanza di una classe non statica è necessario:

- [ ] Allocare la memoria con una istruzione di new
- [ ] Invocare e allocare la classe
- [ ] Allocare la classe
- [ ] Definire la classe

05. Data una classe quante istanze di oggetti si possono creare?

- [ ] Fino a un massimo di 10
- [ ] Soltanto due
- [ ] Un numero illimitato
- [ ] Un numero limitato

06. Il codice sorgente Java che definisce una classe di nome "Prova" in quale file deve essere salvato?

- [ ] Prova.exe
- [ ] Prova.class
- [ ] Un file di testo qualsiasi
- [ ] Prova.java

07. Dato un insieme di classi java è conveniente:

- [ ] Salvare tutto in un file.doc
- [ ] Salvare ogni classe in un file nomeclasse.java
- [ ] Salvare ogni classe in un file estensione ".txt"
- [ ] Salvare ogni classe in una cartella senza estensione

08. Per accedere alla variabile di istanza "n" della classe "NomeClasse" si usa:

- [ ] n.NomeClasse.GetClassVariable
- [ ] NomeClasse.n
- [ ] n.NomeClasse
- [ ] NomeClasse.GetName("n")

09. Descrivere che cos'è un Abstract Data Type e quale relazione ha con il concetto di classe Java.

10. Descrivere la relazione tra classe ed oggetto Java e fare degli esempi.

Lezione 040

01. E' possibile definire dei metodi statici di classe:

- [ ] L'utente non può mai farlo
- [ ] Inserendo il punto prima del metodo
- [ ] Inserendo la parola NoDynamic prima del metodo
- [ ] Inserendo il nome static prima del metodo

02. Data la classe Math che fa parte del package java.lang, la funzione sqrt(double a):

- [ ] Esegue il quadrato del valore nella variabile "a"
- [ ] Esegue la radice cubica del valore nella variabile "a"
- [ ] Arrotonda il valore del numero a virgola mobile di "a"
- [ ] Esegue la radice quadrata del valore nella variabile "a"

03. Un metodo statico può essere:

- [ ] Invocato una sola volta
- [ ] Non può mai esser invocato
- [ ] Invocato senza definire un oggetto
- [ ] Invocato senza programmazione dinamica

04. La funzione "static double cos(double a)" della classe Math restituisce:

- [ ] Il valore doppio della variabile a
- [ ] Il coseno della variabile a
- [ ] True se la variabile a contiene un valore costante
- [ ] L'arcoseno della variabile a

05. Data una classe ed un oggetto, l'operatore instanceof consente di:

- [ ] Sapere se la classe è statica
- [ ] Sapere se l'oggetto è statico
- [ ] Sapere se sia l'oggetto che la classe sono istanze statiche
- [ ] Sapere se l'oggetto è istanza della classe

06. Dato il seguente codice Java: String s = new String("test"); if (s instanceof String), il risultato del test dell'if è:

- [ ] Non si può eseguire
- [ ] false
- [ ] true
- [ ] "test"

07. La funzione "static double abs(double a)" della classe Math restituisce:

- [ ] Il valore zero se a è pari
- [ ] Il valore assoluto della variabile a
- [ ] Il valore zero se a è dispari
- [ ] La parte decimale del valore di a

08. Data la funzione Java: "public void stampa() {System.out.println("Messaggio di prova");}", il metodo main "public static void main(String[] args) {stampa();}" è:

- [ ] Errato in quanto la funzione richiede che venga creata una istanza della classe
- [ ] Corretto in quanto il main viene sempre compilato
- [ ] Errato perchè il main non può chiamare una funzione pubblica
- [ ] Corretto in quanto la funzione non richiede che venga creata una istanza della classe

09. Scrivere la definizione di metodo statico e le implicazioni di avere dei metodi statici per una classe.

10. Descrivere che cos'è l'istanza di una classe e come si crea una nuova istanza, riportando degli esempi di codice.

Lezione 041

01. Data la seguente porzione di codice: "Punto p = new Punto();", dove Punto è il nome di una classe, quale delle seguenti affermazioni è falsa?

- [ ] p è un metodo
- [ ] p è una istanza di classe
- [ ] p è una istanza e si può accedere alle sue variabili
- [ ] p non è un metodo

02. In Java un attributo e un metodo:

- [ ] Sono sinonimi
- [ ] Sono due cose diverse
- [ ] Sono intercambiabili
- [ ] Sono la stessa cosa

03. Dato il seguente spezzone di codice Java "Punto p = new Punto();" se la classe Punto contiene due variabili accessibili x e y, allora:

- [ ] E' possibile accedere ad x e ad y usando p^x e p^y
- [ ] E' possibile accedere ad x e ad y usando p->x e p->y
- [ ] E' impossibile accedere ad x e y
- [ ] E' possibile accedere ad x e ad y usando p.x e p.y

04. Quale delle seguenti affermazioni è corretta?

- [ ] Gli attributi di una classe sono metodi che si invocano dall'esterno
- [ ] Gli attributi di una classe sono delle variabili che ne rappresentano lo stato
- [ ] Gli attributi di una classe sono metodi che si invocano dall'interno
- [ ] Gli attributi di una classe sono metodi ricorsivi

05. L'istruzione Java "return Math.sqrt(x*x + y*y);" ritorna:

- [ ] La radice quadrata di x sommato ad y
- [ ] Il quadrato di x
- [ ] Il quadrato di x sommato ad y
- [ ] La radice quadrata della somma dei quadrati di x e y

06. Data una classe di nome "Punto" ed un metodo dichiarato come "double getDistanza()":

- [ ] Il metodo non è invocabile
- [ ] Il metodo si invoca attraverso il nome di una istanza di classe
- [ ] Il metodo si invoca senza istanza di classe
- [ ] Il metodo si invoca solo dall'interno della classe

07. In Java un attributo si definisce usando:

- [ ] Solo tipi definiti dagli utenti
- [ ] Tipi interessanti
- [ ] Tipi speciali
- [ ] Tipi di base

08. In Java, dovendo dichiarare due variabili x e y, le due seguenti modalità: 1) "double x, y;" e 2) "double x; double y;":

- [ ] La modalità 1 è errata mentre la 2 è corretta
- [ ] Sono entrambe errate
- [ ] Sono equivalenti ma la prima modalità è sconsigliata
- [ ] Sono due modalità diverse che hanno effetti differenti in memoria

Lezione 042

01. In Java un costruttore di una classe:

- [ ] Ha lo stesso nome della classe
- [ ] E' un oggetto che è obbligatorio definire
- [ ] E' un sistema di costruzione degli attributi
- [ ] Ha un metodo standard build

02. Se ci riferiamo ai costruttori, quale fra le seguenti affermazioni è falsa?

- [ ] In Java un costruttore di classe ha il nome della classe
- [ ] In Java un costruttore di classe si invoca nella istruzione new
- [ ] In Java un costruttore di classe è sempre presente
- [ ] In Java un costruttore di classe può avere parametri

03. Data una classe Java è possibile:

- [ ] Avere un costruttore che ritorna un int
- [ ] Avere più costruttori
- [ ] Avere due costruttori con gli stessi parametri
- [ ] Avere un costruttore che ritorna un valore booleano

04. In una classe Java due costruttori:

- [ ] Si distinguono per nome
- [ ] Si distinguono per il tipo di ritorno
- [ ] Si distinguono per i valori che ritornano
- [ ] Si distinguono per numero di parametri passati alla funzione di costruzione

05. In un costruttore è possibile utilizzare il comando "this" per:

- [ ] Riferirsi alle variabili esterne
- [ ] Riferirsi ai link alle variabili e ai metodi
- [ ] Riferirsi all'istanza corrente
- [ ] Riferirsi agli oggetti che di collegano a questo oggetto

06. Si utilizzano più costruttori quando:

- [ ] Si vuole cambiare attributi velocemente
- [ ] Si vuole avere più metodi di inizializzazione
- [ ] Non si utilizzano mai
- [ ] Si vuole istanziare più volte lo stesso oggetto

07. Se in una classe non è stato definito alcun costruttore:

- [ ] La classe non potrà essere compilata
- [ ] Vuol dire che esiste solo il distruttore
- [ ] E' considerato un errore
- [ ] Non è un errore

08. In una classe Java "Punto", il seguente metodo "Punto() { this(0.0, 0.0); }":

- [ ] E' un costruttore che invoca un altro costruttore
- [ ] E' errato perchè manca il tipo di ritorno
- [ ] Genera un errore di compilazione
- [ ] E' incompleto

Lezione 043

01. Quale di queste affermazioni risulta corretta?

- [ ] La parola chiave this non si usa mai all'interno della classe
- [ ] La parola chiave this nel costruttore si usa per ovviare a problemi di sovrapposizione di nomi
- [ ] La parola chiave this nel costruttore non si usa mai
- [ ] La parola chiave this si usa solo per compilare e nel costruttore

02. In Java il concetto di incapsulamente serve a:

- [ ] Incapsulare le istanze dei dati
- [ ] Nascondere il codice
- [ ] Nascondere le classi
- [ ] Nascondere i dettagli implementativi di un Tipo di Dato Astratto

03. L'incapsulamento in Java:

- [ ] Consente di nascondere i dati
- [ ] Consente di avere classi astratte
- [ ] Consente di avere diverse implementazioni
- [ ] Non si usa in Java

04. Uno dei vantaggi dell'incapsulamento è:

- [ ] Evitare che i metodi di una classe possano essere invocati da classi esterne
- [ ] Nascondere i metodi di una classe
- [ ] Bloccare l'accesso ai costruttori di una classe
- [ ] Limitare gli effetti derivanti dalle eventuali modifiche future ad un sistema software

05. Per implementare l'incapsulamento in Java:

- [ ] Si evita l'accesso diretto alla classe da parte di classi esterne
- [ ] Si evita l'accesso diretto ai metodi da parte del sistema operativo
- [ ] Si evita l'accesso diretto ai metodi della classe da parte di classi esterne
- [ ] Si evita l'accesso diretto agli attributi della classe da parte di classi esterne

06. Cosa possiamo dire della visibilità di una variabile in Java?

- [ ] E' visibile solo nei blocchi di codice che contengono il blocco in cui è dichiarata
- [ ] E' visibile solo nel blocco in cui è stata dichiarata e nei blocchi in esso contenuti
- [ ] E' visibile solo dal metodo main
- [ ] E' visibile solo dai metodi che sono figli della classe di appartenenza

07. Una variabile dichiarata in un ciclo for è disponibile:

- [ ] Solo all'esterno del ciclo for e nel metodo
- [ ] Solo all'esterno della classe e del ciclo for
- [ ] Solo all'interno del ciclo for
- [ ] Solo all'interno del main e nella classe

08. Una variabile locale ad un metodo:

- [ ] Si vede dal punto in cui viene dichiarata fino alla fine del blocco del metodo
- [ ] Si vede nel metodo e nella classe a cui appartiene
- [ ] Si vede al di fuori del blocco di codice in cui è stata dichiarata
- [ ] Si vede dal punto in cui l'istanza viene creata con la new fino alla fine del programma che la contiene

09. Descrivere che cos'è l'incapsulamento e come si può implementare nel linguaggio Java.

10. Descrivere cosa si intende per scopo di una variabile.

Lezione 044

01. In Java si usano le eccezioni:

- [ ] Per gli errori in fase di compilazione
- [ ] Solo per gestire gli errori di input/output da tastiera
- [ ] Per gli errori in fase di scrittura
- [ ] Per gli errori a run time per evitare interruzioni inattese del codice

02. Nel codice Java "catch (NumberFormatException ex) {...}" la variabile "ex":

- [ ] Indica una istanza di un metodo di emergenza
- [ ] Indica una istanza di salvataggio del codice
- [ ] Indica una istanza dell'errore che viene passata da chi genera l'errore
- [ ] Indica una istanza di variabile per stampare messaggi a schermo

03. Quale delle seguenti affermazioni è certamente falsa?

- [ ] java.lang.Exception è la classe che gestisce gli errori a run time
- [ ] java.lang.Exception serve a scrivere codice in modo eccezionalmente efficiente
- [ ] java.lang.Exception è inclusa nelle API di Java
- [ ] Le eccezioni sono sottoclassi di java.lang.Exception

04. Il seguente metodo: "public int f(int num) throws DivisionePerZero {...}":

- [ ] Evita che venga generato un errore dal metodo
- [ ] Solleva sempre un errore di DivisionePerZero al chiamante
- [ ] Notifica sempre un errore al chiamante
- [ ] Solleva un errore di DivisionePerZero per qualche input

05. Il blocco catch {...} serve a:

- [ ] Non serve in Java
- [ ] Invoca il modulo di scrittura a schermo
- [ ] Scrivere metodi per gestire gli errori di compilazione
- [ ] Scrivere codice che gestisca l'eccezione a tempo di esecuzione

06. L'istruzione throw si usa per:

- [ ] Lanciare un metodo a tempo di compilazione
- [ ] Lanciare una eccezione in modo programmatico
- [ ] Gestire gli errori di compilazione
- [ ] Bloccare il programma ed uscire

07. Il blocco try catch serve a:

- [ ] Provare ad agganciare un errore a tempo di compilazione
- [ ] Agganciare un errore e gestirlo nel proprio codice
- [ ] Agganciare gli errori di input output
- [ ] Provare a scrivere meglio il codice

08. In Java la classe Exception:

- [ ] Serve a segnalare eccezioni di compilazione
- [ ] Non si usa
- [ ] Serve a definire dei metodi eccezionali
- [ ] Serve a catturare gli errori come oggetto istanza della classe eccezione

Lezione 045

01. Descrivere cosa si intende per ereditarietà ed illustrarlo con un esempio in codice Java.

Lezione 046

01. Descrivere cosa si intende per polimorfismo ed illustrarlo con un esempio in codice Java.

Lezione 047

01. Descrivere che cos'è l'overloading degli operatori e fare un esempio in codice Java.

02. Scrivere una classe Somma che dimostri l'overloading degli operatori in Java.

Lezione 048

01. Nell'ereditarietà delle classi:

- [ ] Una classe si estende mentre una interfaccia non si usa
- [ ] Una classe si implementa (implements) mentre una interfaccia si estende (extends)
- [ ] Una classe si estende (extends) mentre una interfaccia si implementa (implements)
- [ ] Una classe si estende mentre una superclasse non può essere usata

02. In Java le classi che implementano una interfaccia:

- [ ] Possono implementare tutti i metodi dell'interfaccia
- [ ] Devono implementare tutti i metodi dell'interfaccia
- [ ] Non implementano tutti i metodi dell'interfaccia
- [ ] Implementano alcuni i metodi dell'interfaccia

03. In una interfaccia Java i metodi:

- [ ] Contengono instruzioni astratte
- [ ] Non contengono istanze
- [ ] Non contengono istruzioni
- [ ] Contengono parametri

04. Una interfaccia in Java:

- [ ] Non può mai essere instanziata
- [ ] Si può sempre istanziare
- [ ] Si può istanziare ma solo nel metodo main()
- [ ] Non si può mai istanziare tranne che nel metodo main()

05. Quale dei seguenti esempi rappresenta la definizione di una interfaccia in Java?

- [ ] private extends MiaInterfaccia
- [ ] public MiaInterfaccia
- [ ] public extends MiaInterfaccia
- [ ] public interface MiaInterfaccia

06. Cosa è vero se ci si riferisce all'ereditarietà multipla in Java?

- [ ] Non esiste in Java
- [ ] Una classe può avere più superclassi
- [ ] Si implementa tramite l'uso di interfacce
- [ ] Una classe può avere una e una sola sottoclasse

07. Data una classe Java Persona e le interfaccie Cantante e Vip, quali fra le seguenti definizioni della classe CantanteVip è la più corretta?

- [ ] public class CantanteVip extends Vip implements Persona, Cantante {...}
- [ ] public class CantanteVip implements Persona, Cantante, Vip {...}
- [ ] public class CantanteVip extends Persona implements Cantante, Vip {...}
- [ ] public class CantanteVip extends Cantante implements Persona, Vip {...}

08. Tutti i metodi di una interfaccia:

- [ ] Devono essere privati
- [ ] Devono essere pubblici
- [ ] Devono essere implementati
- [ ] Devono essere liberi

09. Descrivere il problema del diamante nell'ereditarietà multipla e mostrare un esempio.

10. Si descriva come il problema dell'ereditarietà multipla viene risolto nel linguaggio Java.

11. Descrivere la differenza tra ereditarietà normale ed ereditarietà multipla e mostrare degli esempi.

Lezione 049

01. Dato un oggetto P di una classe C, a seguito dell'operazione "p=null;" si ottiene che:

- [ ] Lo spazio di memoria occupato dall'oggetto p viene marcato come da conservare
- [ ] Lo spazio di memoria di p contiene 0
- [ ] Vengono settati a null tutti gli attributi dell'oggetto p
- [ ] Lo spazio di memoria occupato dall'oggetto p viene marcato come da cancellare

02. L'errore "error: variable p might not have been initialized" si genera se:

- [ ] La variabile p punta a dati non azzerati
- [ ] La variabile p potrebbe non essere stata inizializzata
- [ ] La variabile p non esiste
- [ ] La variabile p esiste ma non ha dati diversi da zero

03. L'errore "java.lang.NullPointerException" si genera:

- [ ] Se un oggetto contiene troppi dati
- [ ] Se i puntatori degli oggetti si incrociano
- [ ] Quando un oggetto è stato allocato ma gli attributi sono vuoti
- [ ] Se si cerca di accedere ad una variabile che è stata associata a null

04. Le seguenti istruzioni in Java: Persona p=null; p.nome="Mario"; provocano:

- [ ] Un errore perchè p non punta ad alcuna locazione di memoria
- [ ] Nessun effetto perchè p non esiste in memoria
- [ ] La memorizzazione della stringa "Mario" alla variabile nome di p
- [ ] L'associazione di null alla variabile nome di p

05. Il Garbage Collector della Java Virtual Machine:

- [ ] Interviene quando si compila un programma
- [ ] Interviene per liberare la memoria dopo che ad un oggetto viene assegnato null
- [ ] Interviene quando si esegue un programma
- [ ] Interviene per liberare la memoria dopo che termina un programma

06. In Java, quando si assegna il valore null ad un oggetto:

- [ ] Esso non esiste in memoria
- [ ] Viene salvato come valore zero
- [ ] Viene salvato come valore -1
- [ ] Viene salvato come valore speciale e memorizzato nella locazione di memoria dell'oggetto

07. Una stringa String s che non è inizializzata:

- [ ] Contiene un valore nullo memorizzato in memoria
- [ ] Non esiste come variabile e non può essere usata
- [ ] Contiene un valore nullo ovvero non esiste in memoria
- [ ] Non si può usare perchè vuota

08. In Java, relativamente ai valori nulli, le variabili definite come tipi complessi:

- ☐ Possono contenere valori nulli
- ☐ Non possono contenere valori nulli
- ☐ Possono contenere esclusivamente valori nulli
- ☐ Possono contenere solo valori nulli

09. Definire che cos'è il valore null in java, cosa significa e come si usa. Fornire inoltre un esempio di codice che controlli se una variabile contiene il valore nullo.

Lezione 050

01. Descrivere che cos'è e a che cosa serve un metodo astratto in Java e fornire un esempio di codice.

02. Descrivere che cos'è e a che cosa serve una classe astratta in Java e fornire un esempio di codice.

Lezione 051

01. Dove si trovano le funzioni della classe Math?

- [ ] Nel package java.hello
- [ ] Nel package java.lang
- [ ] Nel package java.Geometry
- [ ] Nel package java.io

02. In Java un package:

- [ ] E' un pacchetto di strumenti per stampare
- [ ] E' un pacchetto di variabili
- [ ] E' uno strumento per raggruppare classi che abbiano una qualche similarità
- [ ] E' uno strumento per raggruppare variabili

03. La sintassi gerarchica di un package Java è:

- [ ] dominio1.dominio2.dominio3
- [ ] dominio1.Link.dominio2.Link.dominio3...
- [ ] dominio1/dominio2/dominio3...
- [ ] dominio1->dominio2->dominio3...

04. Un package si può utilizzare nel proprio codice usando la direttiva:

- [ ] define
- [ ] export
- [ ] import
- [ ] use

05. Le classi del package java.lang:

- [ ] Sono importate automaticamente
- [ ] Sono inutili
- [ ] Sono rilevanti
- [ ] Vanno importate dall'utente

06. Un package "miopackage" si definisce all'inizio del codice con la dicitura:

- [ ] package newpackage miopackage
- [ ] package miopackage
- [ ] miopackage package
- [ ] package new miopackage

07. Per importare tutte le classi al package "miopackage" si usa:

- [ ] import All miopackage
- [ ] import miopackage.*
- [ ] import miopackage
- [ ] import All class in miopackage

08. In relazione ai package, un programmatore può:

- [ ] Importare solo package che non siano i suoi
- [ ] Definire ma non includere un proprio package
- [ ] Definire ma non usare un proprio package
- [ ] Definire ed usare un proprio package

09. Che cos'è e a che cosa serve un package nel linguaggio Java? Fornire un esempio di codice Java che dichiari una classe "Analisi" appartenente al package "it.ecampus.corsi" e spiegare in quale directory dovrà essere salvato il file sorgente della classe.

Lezione 052

01. A cosa servono i modificatori di accesso?

- [ ] A stabilire la correttezza di un attributo o di un metodo
- [ ] A stabilire le politiche di visibilità di un attributo o di un metodo
- [ ] A modificare l'accesso di un attributo ma non di un metodo
- [ ] A verificare le politiche di visibilità di un attributo o di un metodo

02. Come si definisce in Java la politica di accesso di un metodo?

- [ ] Verificandone la correttezza formale
- [ ] Con l'indice di livello del metodo
- [ ] Con le politiche di programmazione
- [ ] Con un modificatore di accesso

03. A cosa serve la visibilità public?

- [ ] A inibire l'accesso di un metodo alle sottoclassi
- [ ] A rendere accessibile il metodo main al Sistema Operativo
- [ ] A pubblicare un metodo in un package
- [ ] A inibire l'accesso di un metodo alla superclasse

04. Per quale motivo si usa il livello di visibilità private?

- [ ] Per rendere metodi e attributi inaccessibili anche dalla classe che li contiene
- [ ] Per rendere attributi accessibili solo dalle sottoclassi
- [ ] Per rendere metodi inaccessibili e incapsulare attributi
- [ ] Per rendere metodi accessibili solo da classi dello stesso package

05. Per quale motivo si usa il livello di visibilità protected?

- [ ] Per permettere l'accesso ad attributi e metodi anche alle sottoclassi
- [ ] Per permettere l'accesso solo da classi dello stesso package
- [ ] Per proteggere l'accesso di metodi e attributi da parte di tutte le altre classi
- [ ] Per rendere metodi accessibili solo da classi dello stesso package

06. Per quale motivo si usa il livello di visibilità package?

- [ ] Per permettere l'accesso solo da classi dello stesso package
- [ ] Per rendere metodi inaccessibili e incapsulare attributi
- [ ] Per rendere package inaccessibili
- [ ] Per permettere l'accesso ad attributi e metodi anche alle sottoclassi

07. Nel linguaggio Java esiste la possibilità di rendere una risorsa non accessibile anche alla classe cui appartiene?

- [ ] No
- [ ] Si
- [ ] Dipende dal contesto
- [ ] Nessuna risposta è quella corretta

08. Quanti sono e con quale parola chiave si indicano i livelli di visibilità in Java?

- [ ] Sono 4, indicati con "public", "private", "protected" e ""
- [ ] Sono 2, indicati con "public" e "private"
- [ ] Sono 3, indicati con "public", "private" e "protected"
- [ ] Sono 4, indicati con "public", "private", "protected" e "package"

09. Spiegare quali sono i modificatori di accesso, a cosa servono e come si indicano nel codice con degli esempi.

Lezione 053

01. Il metodo "public boolean equals(Object obj)":

- [ ] Ritorna 0 se l'oggetto passato come parametro è uguale a quello corrente
- [ ] Ritorna false se l'oggetto passato come parametro è uguale a quello corrente
- [ ] Ritorna true se l'oggetto passato come parametro è uguale a quello corrente
- [ ] Ritorna 1 se l'oggetto passato come parametro è uguale a quello corrente

02. La classe Object di Java fa parte di:

- [ ] Del package java.input
- [ ] Del package java.lang
- [ ] Del package java.output
- [ ] Di nessun package

03. La funzione "public String toString()" cosa fa?

- [ ] Ritorna una rappresentazione in formato float dell'oggetto
- [ ] Ritorna una rappresentazione in formato stringa dell'oggetto
- [ ] Ritorna una rappresentazione in formato intera dell'oggetto
- [ ] Ritorna una rappresentazione in formato double dell'oggetto

04. L'espressione "!(obj instanceof NomeClasse)":

- [ ] Genera un errore se obj è una istanza della classe NomeClasse
- [ ] Ritorna true se obj è una istanza della classe NomeClasse
- [ ] Ritorna sempre false
- [ ] Ritorna false se obj è una istanza della classe NomeClasse

05. Per definire un ordinamento esiste l'interfaccia Comparable. Per usarla bisogna:

- [ ] Definire oggetti dell'interfaccia
- [ ] Implementare i metodi dell'interfaccia
- [ ] Istanziare oggetti dell'interfaccia
- [ ] Usare i metodi dell'interfaccia

06. La funzione "protected Object clone()" della classe Object:

- [ ] Crea e ritorna un puntatore all'oggetto
- [ ] Crea e ritorna una copia degli oggetti in memoria
- [ ] Crea e ritorna una copia di tutti gli Object
- [ ] Crea e ritorna una copia di questo oggetto

07. La funzione della classe oggetto "int compareTo(Object o)":

- [ ] Ritorna false se l'oggetto su cui si invoca il metodo non è uguale a "o"
- [ ] Ritorna uno se l'oggetto su cui si invoca il metodo è uguale a "o"
- [ ] Ritorna uno se l'oggetto su cui si invoca il metodo è maggiore di "o"
- [ ] Ritorna true se l'oggetto su cui si invoca il metodo è uguale a "o"

08. Il metodo "protected void finalize()" è chiamato:

- [ ] dal Garbage Collector quando esso rileva che l'oggetto corrente non è più raggiungibile
- [ ] dal compilatore quando esso rileva che l'oggetto corrente non è più raggiungibile
- [ ] dal Sistema Operativo quando esso rileva che l'oggetto corrente non è più raggiungibile
- [ ] dall'utente quando esso rileva che l'oggetto corrente non è più raggiungibile

09. Descrivere quali sono i metodi più importanti ereditati dalla classe Object.

10. Spiegare a cosa serve e come funziona l'interfaccia Comparable, fornendo un esempio in codice Java di una classe che la implementi.

Lezione 055

01. Usando i metodi della classe Math di Java per la conversione degli angoli espressi in gradi o in radianti, scrivere un esempio di codice che converta l'angolo pi greco mezzi in gradi e l'angolo 15 gradi in radianti.

02. Elencare e descrivere le funzioni per il calcolo dei logaritmi presenti nella classe Math di Java.

Lezione 056

01. Il metodo Math.random():

- [ ] Genera numeri double pseudocasuali fra 0 e 1
- [ ] Genera numeri double pseudocasuali fra 1 e 10
- [ ] Genera numeri interi pseudocasuali fra 0 e 1
- [ ] Genera numeri interi pseudocasuali fra 1 e 10

02. E' possibile generare numeri casuali:

- [ ] usando esclusivamente hardware
- [ ] sia usando algoritmi software che, in alcuni casi, usando hardware
- [ ] usando esclusivamente codice scritto dall'utente
- [ ] usando esclusivamente codice del sistema operativo

03. Come si possono generare numeri pseudocasuali in Java?

- [ ] Si può utilizzare la classe RandomNumber
- [ ] Si può utilizzare sia la classe Random che la classe Math
- [ ] Si può utilizzare il metodo Math.pseudorandom()
- [ ] Si può utilizzare la classe PseudoRandom

04. Quali sono le proprietà che deve soddisfare una sequenza di numeri pseudocasuali?

- [ ] devono essere indipendenti
- [ ] devono essere distribuiti uniformemente e devono essere indipendenti fra di loro
- [ ] devono essere distribuiti in modo costante
- [ ] devono essere generati uno di seguito all'altro nell'intervallo di definizione

05. Considerando l'intervallo [1,10], la sequenza di numeri {1,2,3,4,5,6,7,8,9,10} non è pseudocasuale perchè:

- [ ] non soddisfa la proprietà di indipendenza fra i numeri
- [ ] i numeri sono interi e non double
- [ ] i numeri non sono distribuiti equamente nell'intervallo
- [ ] i numeri non sono sufficientemente casuali

06. La classe Random:

- [ ] genera numeri pseudocasuali solo interi
- [ ] genera numeri pseudocasuali solo interi e solo nell'intervallo [0,n]
- [ ] genera numeri pseudocasuali solo double
- [ ] genera numeri pseudocasuali interi e double

07. Da cosa è generato un numero pseudocasuale?

- [ ] da un algoritmo ricorsivo
- [ ] da un algoritmo deterministico
- [ ] da un algoritmo casuale
- [ ] da un algoritmo random

08. Quali caratteristiche ha un numero pseudocasuale?

- [ ] ha proprietà sia ricorsive che iterative
- [ ] ha proprietà numeriche di tipo esponenziale
- [ ] ha proprietà statistiche simili a quelle di un processo casuale
- [ ] ha proprietà sia simmetriche che transitive

Lezione 057

01. Quale delle seguenti frasi è errata se riferita alla classe Nodo di una Lista?

- [ ] La classe Nodo rappresenta un elemento di una Lista
- [ ] La classe Nodo contiene le informazioni strutturali di una Lista
- [ ] La classe Nodo permette di manipolare gli elementi di una Lista
- [ ] La classe Nodo contiene le informazioni topologiche di una Lista

02. Cosa sono le proprietà topologiche di una lista?

- [ ] Le sue proprietà di memorizzazione
- [ ] Le sue proprietà dimensionali
- [ ] Le sue proprietà dinamiche
- [ ] Le sue proprietà geometriche

03. Perchè si introduce il Tipo di Dato Astratto Lista?

- [ ] Per migliorare la gestione della memoria di Java
- [ ] Per supportare meglio gli array nella gestione della memoria
- [ ] Per superare le limitazioni del tipo di dato array
- [ ] Per allocare zone di memoria contigue più ampie rispetto agli array

04. Cos'è un oggetto della classe Lista?

- [ ] Un tipo di dato base
- [ ] Un oggetto dinamico
- [ ] Un oggetto statico
- [ ] Un tipo di dato astratto

05. Quale fra queste caratteristiche non è corretta se riferita alle Liste?

- [ ] La sua lunghezza non è fissa
- [ ] I suoi elementi occupano celle di memoria contigue
- [ ] I suoi elementi non occupano celle di memoria contigue
- [ ] Aggiungere o eliminare un elemento è indipendente dalla sua posizione

06. Quale fra le seguenti metafore è stata usata a lezione per descrivere le Liste?

- [ ] La metafora delle perle e dei ganci
- [ ] La metafora dell'ago e del filo
- [ ] La metafora del nastro illimitato
- [ ] La metafora del filo annodato

07. Lo schema del Nodo studiato a lezione:

- [ ] contiene un solo attributo
- [ ] contiene due attributi, info e next
- [ ] contiene due attributi, uno che punta al nodo precedente ed uno al successivo
- [ ] contiene l'attributo next e un puntatore a se stesso

08. La definizione dell'attributo next nella classe Nodo:

- [ ] ne rende la definizione ricorsiva
- [ ] ne rende la definizione nulla
- [ ] ne rende la definizione iterativa
- [ ] ne rende la definizione incongruente

Lezione 058

01. Quando una classe Lista semplicemente linkata viene creata, come viene inizializzata la testa della lista?

- [ ] Eliminando il nodo di testa
- [ ] Azzerando il nodo di testa
- [ ] Con il valore null
- [ ] Con il valore -1

02. Un elemento nodo di una Lista semplicemente linkata:

- [ ] Contiene un attributo info e un riferimento al nodo precedente
- [ ] Contiene un riferimento al nodo successivo
- [ ] Contiene un riferimento al nodo precedente ed al successivo
- [ ] Contiene un riferimento al nodo precedente

03. Gli attributi info e next della classe Nodo di una Lista semplicemente linkata:

- [ ] sono dichiarati private e sono accessibili tramite i metodi getter e setter
- [ ] sono dichiarati private e non sono accessibili se non dalla classe
- [ ] sono solitamente dichiarati static
- [ ] sono solitamente dichiarati public

04. Qual è il simbolo per raffigurare graficamente che l'attributo next di un Nodo di una Lista vale null?

- [ ] Un quadrato vuoto
- [ ] La cifra zero
- [ ] Il simbolo di insieme vuoto
- [ ] Il simbolo elettrico di messa a terra

05. Quale fra le seguenti caratteristiche non è riferita ad una Lista semplicemente linkata?

- [ ] Gli elementi che la compongono non occupano locazioni di memoria contigue
- [ ] Il numero dei suoi elementi viene fissato dal costruttore
- [ ] In generale, sono più efficienti nell'aggiunta di elementi rispetto agli array
- [ ] Il numero dei suoi elementi può variare durante l'esecuzione del programma

06. Nella classe Lista semplicemente linkata:

- [ ] Esiste un riferimento a tutti i nodi
- [ ] Esiste un riferimento alla coda
- [ ] Non esiste riferimento ad alcun nodo
- [ ] Esiste un riferimento alla testa

07. Di solito, quali attributi sono presenti nella classe Nodo di una Lista semplicemente linkata?

- [ ] next e previous
- [ ] head e tail
- [ ] head e next
- [ ] info e next

08. Dato il nodo n di una Lista semplicemente linkata definito come "Nodo n = new Nodo(1);" come posso collegargli un nuovo nodo contenente il valore 2?

- ☐ n.addNodo(new Nodo(2));
- ☐ n.setNext(new Nodo(2));
- ☐ n+=new Nodo(2);
- ☐ n.setNext(2);

09. Scrivere in codice Java e commentare il metodo toString() della classe Lista semplicemente linkata.

10. Descrivere il contenuto della classe Nodo e fare un esempio di implementazione della classe in codice Java.

Lezione 059

01. Scrivere in codice Java e commentare un metodo che verifichi se una Lista semplicemente linkata è vuota oppure no (restituisce treu se vuota, false altrimenti).

02. Scrivere in codice Java e commentare il metodo di inserimento in testa della classe Lista semplicemente Linkata.

Lezione 060

01. Che differenza c'è tra la ricerca di un nodo in una Lista ordinata o non ordinata?

- [ ] In media la ricerca in una lista ordinata è più efficiente
- [ ] Possiamo effettuare la ricerca sono in liste ordinate
- [ ] Possiamo partire dalla testa nel primo caso e dalla coda nel secondo caso
- [ ] Sono identiche

02. Se effettuiamo la ricerca di un nodo in una Lista vuota:

- [ ] La ricerca non si può fare e viene generata un'eccezione di lista vuota
- [ ] La ricerca va avanti e restituisce comunque un risultato
- [ ] La JVM gestisce automaticamente l'eccezione rilevando che il primo nodo è null
- [ ] Il compilatore genera un errore perchè rileva che il primo nodo è null

03. Dato un nodo di una classe Lista, l'istruzione "aux = aux.getNext()":

- [ ] Scorre in avanti al prossimo nodo assegnando al valore aux l'indirizzo contenuto in aux.next
- [ ] E' ricorsiva e non si può eseguire
- [ ] Scorre in avanti al prossimo nodo assegnando al valore aux l'indirizzo contenuto in aux
- [ ] Scorre indietro al nodo precedente assegnando al valore aux l'indirizzo contenuto in aux

04. L'espressione "(aux!=null && aux.getInfo()<key)" con aux variabile ausiliaria Nodo di una Lista:

- [ ] Vale true se aux punta ad un nodo inesistente
- [ ] Vale true se aux punta ad un nodo esistente (non nullo) o se il valore info contenuto nel nodo è minore di "key"
- [ ] Vale true se aux punta ad un nodo esistente (non nullo) e se il valore info contenuto nel nodo è minore di "key"
- [ ] Vale true se aux punta ad un nodo esistente (non nullo) e se l'attributo info contenuto nel nodo è maggiore di "key"

05. Nel caso di ricerca di un nodo avente "info=3" in una Lista ordinata di interi:

- [ ] La ricerca visita la lista e per ogni nodo se trova il valore 3 si ferma e ritorna il riferimento al nodo
- [ ] La ricerca visita la lista e per ogni nodo se trova il valore 3 continua a visitare e ritorna il riferimento al nodo
- [ ] La ricerca visita tutta la lista e se trova il valore 3 in una info ritorna true e continua a visitare la lista
- [ ] La ricerca visita tutta la lista e se trova il valore 3 in una info cancella il resto della lista e ritorna il riferimento al nodo

06. L'espressione "(aux != null && aux.getInfo() == key)" ritorna:

- [ ] True se il valore della variabile aux è null e se il contenuto dell'info nel nodo è diverso al valore key
- [ ] True se il valore della variabile aux è non null e se il contenuto dell'info nel nodo è diverso al valore key
- [ ] True se il valore della variabile aux è non null e se il contenuto dell'info nel nodo è pari al valore key
- [ ] True se il valore della variabile aux è non null o se il contenuto dell'info nel nodo è diverso al valore key

07. La ricerca di un nodo in una Lista ordinata:

- [ ] E' possibile ma più complicata se la lista è ordinata
- [ ] Non è possibile
- [ ] E' possibile e se ordinata è più veloce
- [ ] E' possibile ma dipende dal tipo di nodo

08. Per ricercare un valore in una lista:

- [ ] Si parte facendo assegnando null alla variabile temporanea aux di tipo Nodo
- [ ] Si parte facendo puntare una variabile temporanea aux di tipo Nodo alla testa della lista
- [ ] Si parte facendo puntare una variabile temporanea aux di tipo Nodo al centro della lista
- [ ] Si parte facendo puntare una variabile temporanea aux di tipo Nodo alla coda della lista

09. Scrivere in codice java e commentare un metodo di ricerca di un nodo in una Lista semplicemente linkata non ordinata.

Lezione 061

01. Se una Lista contiene un solo nodo:

- [ ] L'eliminazione di un nodo non si può fare
- [ ] L'eliminazione di un nodo genera una lista vuota e la testa sarà settata automaticamente a null
- [ ] L'eliminazione di un nodo si può fare ma la lista punta sempre alla testa
- [ ] L'eliminazione di un nodo genera una lista vuota e la testa deve essere settata a null

02. Per eliminare il primo nodo di una Lista non vuota l'istruzione corretta è:

- [ ] "head.info = null" che setta a null il valore del puntatore di testa
- [ ] "head = head.getNext()" che setta la testa al prossimo nodo
- [ ] "head = null" che setta a null il nodo di testa
- [ ] "head.Next = null" che setta a null il prossimo nodo

03. Se una Lista contiene due nodi e si vuole eliminare l'ultimo nodo:

- [ ] Il nodo di testa deve essere modificato settando il campo next pari a null
- [ ] Il nodo di testa deve esser modificato settando il campo next pari al nodo eliminato
- [ ] Il nodo di testa va eliminato
- [ ] Il nodo di testa non deve essere modificato

04. Dati due nodi Nodo1 e Nodo2 di una Lista, cosa fa l'istruzione "Nodo1.setNext(Nodo2.getNext())"?

- [ ] Fa sì che entrambi i nodi abbiano lo stesso nodo successivo
- [ ] Fa sì che il secondo nodo sia sostituito dal primo
- [ ] Elimina la testa della Lista
- [ ] Elimina il primo e il secondo nodo

05. Per eliminare l'ultimo nodo di una Lista semplicemente linkata:

- [ ] Basta partire dal nodo in coda
- [ ] Basta eliminare il nodo di testa
- [ ] Devo scorrere prima tutti i nodi della Lista partendo dalla coda
- [ ] Devo scorrere prima tutti i nodi della Lista partendo dalla testa

06. In caso di cancellazione di un nodo intermedio di una Lista:

- [ ] Basta usare solo il riferimento al nodo di testa
- [ ] Non serve nessuna variabile nodo ausiliaria perchè la lista rimane collegata
- [ ] Serve avere un riferimento al nodo precedente ed uno al successivo per collegare la lista
- [ ] Servono due riferimenti, uno alla testa ed uno alla coda della lista

07. La seguente istruzione "head = head.getNext();" che effetto ha su una Lista?

- [ ] Si usa per inserire un elemento in testa ad una lista
- [ ] Si usa per cancellare una intera lista
- [ ] Si usa per cancellare il primo elemento di una lista
- [ ] Si usa per duplicare l'elemento in testa ad una lista

08. Per eliminare un nodo da una Lista:

- [ ] E' necessario manipolare gli attributi next dei nodi in modo da lasciare la struttura sempre concatenata
- [ ] E' necessario manipolare solo la testa della lista
- [ ] Non serve considerare la posizione del nodo
- [ ] E' necessario manipolare gli attributi info dei nodi in modo da lasciare la struttura sempre concatenata

09. Scrivere in codice Java e commentare un metodo che effettui l'eliminazione del nodo di testa di una Lista semplicemente linkata.

Lezione 062

01. In caso di Lista doppiamente linkata:

- [ ] E' possibile scorrere la lista solo a partire dalla testa
- [ ] Non è possibile scorrere la lista in due direzioni
- [ ] E' possibile scorrere la lista solo a partire dalla coda
- [ ] E' possibile scorrere la lista in due direzioni

02. Il Nodo di una Lista può contenere una stringa?

- [ ] No perchè non fa parte del dominio numerico
- [ ] Basta creare opportunamente l'attributo info della classe Nodo
- [ ] Nella classe Nodo possono essere memorizzati solo tipi numerici
- [ ] Non si può rappresentare come stringa ma solo come array di caratteri

03. Nell'implementazione di una Lista doppiamente linkata, quando si trova l'istruzione "head=new NodoDL(val)"?

- [ ] Quando la lista è non vuota e si vuole inserire un nodo valore "val"
- [ ] Quando la lista è vuota e si vuole inserire un nodo con valore "val"
- [ ] Quando si vuole inserire un nodo con valore "val" in testa
- [ ] Mai perchè è errata

04. In caso di Lista doppiamente linkata contenente solo nodo:

- [ ] Il puntatore alla coda punta a null
- [ ] Il puntatore alla coda e alla testa non hanno lo stesso valore
- [ ] Il puntatore alla coda e alla testa puntano a null
- [ ] Il puntatore alla coda e alla testa hanno lo stesso valore

05. Nell'implementazione di una Lista, l'istruzione "head.getNext()==null" ritorna true se:

- [ ] La lista non contiene alcun nodo
- [ ] La lista contiene almeno un nodo null
- [ ] La lista contiene uno e un solo nodo
- [ ] La lista contiene due nodi di cui il secondo punta a null

06. Una Lista doppiamente linkata ha il nodo di testa tale che:

- [ ] Ha un puntatore punta al primo nodo e uno che punta a null
- [ ] Ha un puntatore punta al secondo e uno al terzo nodo
- [ ] Ha un puntatore punta al primo e uno al terzo nodo
- [ ] Ha un puntatore punta al primo e uno al secondo nodo

07. In una Lista doppiamente linkata:

- [ ] Si può inserire un nodo intermedio solo creando una copia dei nodi
- [ ] La rimozione di un nodo intermedio è più semplice rispetto alla Lista semplicemente linkata
- [ ] Non si può inserire direttamente il nodo di coda senza scorrere tutti gli elementi
- [ ] Inserire un nodo solo partendo dalla testa

08. Una lista doppiamente linkata:

- [ ] E' una lista in cui ogni nodo ha un puntatore al successivo ed uno al precedente
- [ ] E' una lista doppia
- [ ] E' una lista in cui ogni nodo può memorizzare due valori
- [ ] E' l'unico tipo di lista in cui i nodi sono collegati fra loro

09. Descrivere la struttura di una Lista doppiamente linkata e scrivere il codice della classe in linguaggio Java.

10. Descrivere la struttura di un Nodo per una Lista doppiamente linkata e scrivere il codice della classe in linguaggio Java.

Lezione 063

01. In un Albero Binario Completo (ovvero ogni nodo ha due figli) al livello K ci sono:

- [ ] 2^(K-1) nodi
- [ ] K^2 nodi
- [ ] 2^K nodi
- [ ] 2^(K-2) nodi

02. In una struttura dati Albero, un nodo senza figli si chiama:

- [ ] Radice
- [ ] Figlio
- [ ] Ramo
- [ ] Foglia

03. In un Albero Binario:

- [ ] Ogni nodo ha due archi (binari)
- [ ] Ogni nodo ha due padri
- [ ] Solo la Radice può avere due figli
- [ ] Ogni nodo ha al più due figli

04. Che caratteristica deve avere un nodo interno di un Albero?

- [ ] deve avere almeno due figli
- [ ] deve avere almeno un figlio
- [ ] deve avere dei fratelli
- [ ] deve essere un nodo Foglia

05. Un Albero è una struttura dati fatta da nodi e connessioni detti rami dove:

- [ ] Ogni nodo è autonomo e non raggiungibile
- [ ] Ogni nodo è raggiungibile da ogni altro nodo
- [ ] Ogni nodo è il padre di tutti i nodi
- [ ] Ogni nodo (eccetto il nodo radice) è connesso tramite un ramo ad un altro nodo, che ne è il padre, e di cui rappresenta il figlio

06. In una struttura dati Albero, un cammino con K nodi ha:

- [ ] K archi e lunghezza K-1
- [ ] K-1 archi e lunghezza K
- [ ] K+1 archi, ovvero ha lunghezza K+1
- [ ] K-1 archi, ovvero ha lunghezza K-1

07. Dato un Albero, il numero di archi coinvolti in un cammino si chiama:

- [ ] Altezza del cammino
- [ ] Lunghezza del cammino
- [ ] Altezza dell'Albero
- [ ] Profondità del cammino

08. La ricerca di nodi in un Albero implica:

- [ ] Uguale numero di passi rispetto ad una lista
- [ ] Un numero di passi maggiore rispetto a una lista
- [ ] Nessun attraversamento di nodi
- [ ] Un numero di passi minore rispetto a una lista

Lezione 064

01. La struttura Nodo di un Albero binario ricorda:

- [ ] L'implementazione degli array nelle stringhe
- [ ] L'implementazione delle stringhe di caratteri
- [ ] L'implementazione dei vettori negli array
- [ ] L'implementazione dei Nodi della struttura dati Lista

02. La classe Nodo di un Albero Binario:

- [ ] Deve contenere due attributi info
- [ ] Deve contenere almeno due attributi nodo e un attributi info
- [ ] Deve contenere almeno due attributi info e un attributo nodo
- [ ] Deve contenere almeno due attributi info

03. In un Albero Binario la visita Preorder:

- [ ] Visita la radice, visita ricorsivamente il sotto-albero sinistro, visita ricorsivamente il sotto-albero destro
- [ ] Visita il nodo destro, visita la radice, visita ricorsivamente il sotto-albero sinistro
- [ ] Visita la radice, visita ricorsivamente il sotto-albero destro, visita ricorsivamente il sotto-albero sinistro
- [ ] Visita il nodo sinistro, visita la radice, visita ricorsivamente il sotto-albero destro

04. In un Albero Binario la visita Postorder:

- [ ] Visita la radice, visita ricorsivamente il sotto-albero destro, visita il sotto-albero sinistro
- [ ] Visita il nodo destro, visita la radice, visita ricorsivamente il sotto-albero sinistro
- [ ] Visita il ricorsivamente sotto-albero sinistro, visita ricorsivamente il sotto-albero destro, visita la radice
- [ ] Visita il nodo sinistro, visita la radice, visita ricorsivamente il sotto-albero destro

05. Un Albero Binario si costruisce con una classe Nodo che contiene:

- [ ] Un intero per il figlio sinistro e un intero per il figlio destro
- [ ] L'informazione, un oggetto Nodo figlio sinistro e un oggetto Nodo figlio destro
- [ ] Un intero per il figlio sinistro, un intero per il figlio destro, un oggetto Nodo per l'informazione
- [ ] L'informazione, un intero per il figlio sinistro e un intero per il figlio destro

06. In un Albero Binario:

- [ ] La connessione tra nodi si implementa con i numeri
- [ ] La connessione tra nodi si implementa con i puntatori
- [ ] La connessione tra nodi si implementa con i double
- [ ] La connessione tra nodi si implementa con gli interi

07. In quale visita degli Alberi Binari si stampa prima l'info del nodo e poi si visitano il sotto-albero sinistro e poi il sotto-albero destro?

- [ ] Nella visita Inorder
- [ ] Nella visita Firstorder
- [ ] Nella visita Preorder
- [ ] Nella visita Postorder

08. Nelle visite Preoder, Inorder e Postorder di un Albero Binario, quale tecnica si usa per visitare i nodi?

- [ ] Si usa la riflessione
- [ ] Si usa un ciclo for
- [ ] Si usa la ricorsione
- [ ] Si usa l'iterazione

09. Descrivere la struttura del Nodo di un Albero Binario e scrivere il codice della lcasse in linguaggio Java.

10. Descrivere la struttura di un Albero Binario e scrivere il codice della lcasse in linguaggio Java.

Lezione 065

01. Il Collections Framework di Java contiene:

- [ ] Solo interfacce e classi astratte
- [ ] Solo classi astratte
- [ ] Interfacce, classi astratte e classi concrete
- [ ] Solo interfacce

02. Le interfacce del Collections Framework di Java sono contenute:

- [ ] Nel package java.io
- [ ] Nel package javax.swing
- [ ] Nel package java.util
- [ ] Nel package java.lang

03. Le Liste del Collections Framework di Java:

- [ ] Non hanno nulla a che vedere con il concetto matematico di insieme
- [ ] Sono collezioni di tipi base di Java
- [ ] Implementano l'intefaccia Collection
- [ ] Non hanno nulla a che vedere con le collezioni

04. Il metodo "boolean add(Object o)" di una Collection Java:

- [ ] Verifica se si può aggiungere un elemento alla collezione (ritorna false se non è possibile)
- [ ] Aggiunge un elemento alla collezione (ritorna true se non è possibile aggiungerlo)
- [ ] Verifica se si può aggiungere un elemento alla collezione (ritorna true se è possibile)
- [ ] Aggiunge un elemento alla collezione (ritorna false se non è possibile aggiungerlo)

05. Il metodo "boolean removeAll(Collection c)" di una Collection Java:

- [ ] Rimuove gli elementi della collezione passata come parametro che esistono nella collezione corrente e torna true se la collezione è stata modificata
- [ ] Rimuove alcuni elementi della collezione passata come parametro che esistono nella collezione corrente e torna un boolean
- [ ] Rimuove dalla collezione passata come parametro gli elementi che esistono anche nella collezione corrente e torna true se la collezione c è stata modificata
- [ ] Svuota la collezione c passata come parametro

06. Il metodo boolean "containsAll(Collection c)" di una Collection Java:

- [ ] Ritorna true se tutti gli elementi della collezione c passata come parametro sono contenuti anche nella collezione corrente
- [ ] Ritorna true se la collezione c passata come parametro contiene gli stessi elementi della collezione corrente
- [ ] Ritorna true se la collezione c è piena
- [ ] Ritorna true se la collezione c è vuota

07. Il metodo void "clear()" di una Collection Java:

- [ ] Elimina dalla collezione gli elementi nulli
- [ ] Elimina la collezione dalla memoria
- [ ] Elimina dalla collezione di tutti gli elementi
- [ ] Elimina dalla collezione gli elementi non permanenti

08. Il metodo "Collection.iterator()" di una Collection Java:

- [ ] Ritorna una istanza di Iterator che possiamo usare la testa della collezione
- [ ] Ritorna una istanza di Iterator che possiamo usare per leggere il primo elemento della collezione
- [ ] Ritorna una istanza di Iterator che possiamo usare per scorrere ed eliminare gli elementi della collezione
- [ ] Ritorna una istanza di Iterator che possiamo usare per rimuovere solo gli elementi finali della collezione

09. Descrivere il ruolo di ciascuna delle interfacce Collection, List, Set e Queue e il rapporto che esiste fra di loro in termini di gerarchia.

10. Descrivere che cos'è il Collections Framework di Java ed elencare le interfacce principali.

Lezione 066

01. Quale fra le seguenti affermazioni è falsa se riferita ad un ArrayList?

- [ ] La sua implentazione interna è basata su array
- [ ] E' un oggetto immutabile
- [ ] Eredita caratteristiche da Collection
- [ ] L'aggiunta di nuovi elementi può essere molto poco efficiente

02. Quale fra le seguenti affermazioni è falsa se riferita ad una LinkedList?

- [ ] Non è immutabile
- [ ] L'accesso ai suoi elementi è meno efficiente rispetto ad un ArrayList
- [ ] L'inserimento e la rimozione di elementi è più efficiente rispetto ad un ArrayList
- [ ] L'inserimento e la modifica degli elementi sono in media sono poco efficienti

03. Quale fra le seguenti inizializzazioni di un ArrayList è errata?

- [ ] ArrayList a = new ArrayList();
- [ ] Collection a = new ArrayList();
- [ ] ArrayList a = new ArrayList(10);
- [ ] LinkedList a = new ArrayList();

04. Quale fra le seguenti affermazioni è falsa se riferita ad un ArrayList?

- [ ] L'accesso ai suoi elementi è più efficiente rispetto ad una LinkedList
- [ ] Le operazioni di inserimento sono potenzialmente poco costose
- [ ] L'inserimento e la rimozione di elementi è più meno efficiente rispetto ad un ArrayList
- [ ] Non è immutabile

05. Come si scorrono gli elementi di una Collection?

- [ ] Dipende dalla sottoclasse concreta che si sta considerando
- [ ] Con un oggetto di tipo Iterator
- [ ] Iterando un oggetto wrapper numerico
- [ ] Con un puntatore a Collection

06. Qual è la differenza tra le classi ArrayList e LinkedList?

- [ ] ArrayList è implementata con un array, LinkedList con una lista
- [ ] Sono identiche
- [ ] ArrayList contiene solo oggetti di tipo array
- [ ] ArrayList contiene array, LinkedList contiene liste

07. Come vengono utilizzate le classi wrapper dei tipi numerici nelle Liste?

- [ ] A permettere alle sottoclassi del Collections Framework di contenere numeri
- [ ] A permettere a un Iterator di manipolare numeri
- [ ] A permettere alle sottoclassi dei Collection di leggere numeri dalla tastiera
- [ ] A permettere a una Collection di wrappare oggetti

08. Qual è lo scopo delle classi wrapper dei tipi numerici?

- [ ] A wrappare oggetti primitivi
- [ ] A contenere tipi numerici primitivi
- [ ] A contenere oggetti di tipo numerico
- [ ] A manipolare oggetti numerici

09. Descrivere le principali differenze tra ArrayList e LinkedList, fornendo degli esempi di codice.

10. Descrivere le principali classi wrapper Java dei tipi numerici.

Lezione 067

01. A cosa serve l'operatore diamond di Java?

- [ ] Permette di aggiungere tipi numerici in una Lista
- [ ] Permette di restringere gli oggetti che si possono aggiungere a una Collection
- [ ] A far funzionare il Collections Framework
- [ ] Permette di utilizzare i tipi wrapper numerici

02. Cosa è falso se riferito all'operatore diamond di Java?

- [ ] Permette di definire metodi generici
- [ ] Funziona con tutti le classi Java
- [ ] Permette di definire classi generiche
- [ ] Funziona solo con le classi wrapper numeriche

03. L'operatore diamond di Java:

- [ ] Permette di specificare vincoli sui tipi che si possono passare ad un metodo
- [ ] Permette di evitare cast impliciti
- [ ] Permette di restringere i tipi utilizzabili
- [ ] Permette di evitare cast espliciti

04. La correttezza delle operazioni vincolate con l'operatore diamond:

- [ ] Viene controllata dal compilatore
- [ ] Viene controllata a tempo di esecuzione
- [ ] Deve essere controllata dall'utente
- [ ] Viene controllata dalla Java Virtual Machine

05. Cosa è falso se riferito alla definizione "LinkedList list = new LinkedList<>();":

- [ ] Demandiamo il controllo al compilatore circa gli oggetti inseriti in list
- [ ] Se nella lista inseriamo un Double, a tempo di esecuzione verrà lanciata una eccezione
- [ ] Restringiamo gli oggetti che si possono inserire in list alla classe Integer
- [ ] Definiamo una Lista che accetta solo istanze della classe Integer

06. Come possiamo limitare gli oggetti generici di un metodo o di una classe?

- [ ] In nessun modo
- [ ] Possiamo restringerli alle classi di un certo package
- [ ] Possiamo restringerli ai tipi base
- [ ] Possiamo restringerli ad una certa gerarchia di classi

07. Il metodo "public Collection operazione(T[] a)":

- [ ] Ritorna una istanza della classe generica T
- [ ] Per farlo funzionare dobbiamo definire la classe T nel file T.java
- [ ] Ritorna un oggetto del tipo base T
- [ ] Ritorna una istanza di Collection contenente oggetti di tipo T

08. Il metodo definito come "public T metodo()":

- [ ] Restringe il tipo T alla interfaccia Comparable ed estende Number
- [ ] Dice che il tipo T può estendere sia Number che Comparable
- [ ] Restringe il tipo T alla classe Number o alla interfaccia Comparable
- [ ] Dice che il tipo T deve estendere Number ed implementare Comparable

09. Descrivere che cos'è e a che cosa serve l'operatore diamond in Java e fornire degli esempi di codice.

10. Descrivere che cos'è un metodo generico in Java e fornire un esempio utilizzando il codice Java.

Lezione 068

01. Che cos'è la complessità computazionale?

- [ ] Una misura del tempo di esecuzione di un algoritmo sun certo hardware
- [ ] Una misura legata all'efficienza di un algoritmo al variare dell'input
- [ ] Un modo per capire quanta memoria occupa un algoritmo
- [ ] Un modo per misurare quanto è difficile un algoritmo

02. In termini matematico-insiemistici, un problema può essere visto come:

- [ ] Una funzione tra due insiemi, l'insieme degli input e quello degli output
- [ ] Una funzione tra i parametri e il tipo di ritorno
- [ ] Una funzione tra due insiemi, l'insieme delle istanze e quello delle soluzioni
- [ ] L'algoritmo che lo implementa

03. Per definizione, un Algoritmo deve essere:

- [ ] non ambiguo
- [ ] finito, non ambiguo e terminante
- [ ] finito e non ambiguo
- [ ] finito, efficiente e terminante

04. Un Algoritmo che non termina mai:

- [ ] è inaccettabile perchè occupa inutilmente risorse computazionali
- [ ] in realtà non è un Algoritmo
- [ ] è molto poco efficiente
- [ ] è accettabile perchè non spreca memoria

05. In generale, il tempo di esecuzione di un Algoritmo:

- [ ] Aumenta con l'aumentare della dimensione dell'input
- [ ] Dipende da molti fattori che non possono essere previsti
- [ ] Dipende dall'hardware su cui gira ma è indipendente dai dati
- [ ] E' sempre costante

06. La dipendenza di un Algoritmo tra la dimensione dell'input e il tempo di esecuzione:

- [ ] Dipende da come l'Algoritmo è stato implementato
- [ ] Dipende solo dall'hardware
- [ ] Dipende da innumerevoli fattori
- [ ] Non può essere determinato

07. Per effettuare l'Analisi di complessità di un Algoritmo:

- [ ] Bisogna considerare la quantità di RAM occupata e la potenza della CPU
- [ ] Bisogna tenere in considerazione l'hardware su cui gira
- [ ] Non è utile considerare il suo tempo di esecuzione
- [ ] Bisogna considerare il linguaggio di programmazione utilizzato

08. Per valutare le prestazioni di un Algoritmo:

- [ ] Non possiamo usare nulla
- [ ] Si usa il tempo di esecuzione
- [ ] Si usa la quantità di memoria occupata
- [ ] Non viene mai usato il tempo di esecuzione

Lezione 070

01. Descrivere che cos'è la notazione O-grande e fornire degli esempi.

Lezione 071

01. Per rilevare l'istante temporale in cui si verifica un evento, la classe StopWatch:

- [ ] Chiede il tempo alla Java Virtual Machine
- [ ] Chiede il tempo al Sistema Operativo
- [ ] Chiede il tempo ad una classe della libreria standard di Java
- [ ] Chiede il tempo alla superclasse

02. Per misurare l'efficienza dei propri algoritmi possiamo:

- [ ] Utilizzare la classe StopWatch
- [ ] Utilizzare un cronografo automatico di precisione
- [ ] Utilizzare un cronometro
- [ ] Utilizzare la classe Swatch

03. Cosa è falso se riferito alla classe StopWatch?

- [ ] Posso misurare il tempo trascorso dall'avvio della Java Virtual Machine
- [ ] Posso misurare l'efficienza di un algoritmo
- [ ] Posso misurare il tempo trascorso tra due eventi
- [ ] Posso misurare sia i tempi intermedi che il tempo totale

04. Cosa è falso se riferito alla funzione System.currentTimeMillis()?

- [ ] Si può utilizzare per misurare il tempo CPU del Sistema Operativo
- [ ] Si può utilizzare per misurare il tempo trascorso fra due eventi
- [ ] Si può utilizzare per misurare l'efficienza di un algoritmo
- [ ] Si può utilizzare per contare i millisecondi trascorsi dal 1 gennaio 1970

05. La classe StopWatch misura il tempo:

- [ ] Calcolando la differenza in millisecondi tra due misurazioni
- [ ] Effettuando una chiamata alla Java Virtual Machine
- [ ] Effettuando una chiamata al Sistema Operativo
- [ ] Calcolando la differenza in secondi tra due misurazioni

06. Dopo aver creato una nuova istanza di StopWatch con "StopWatch t = new StopWatch();" con t.start():

- [ ] Si avvia il timer della Java Virtual Machine
- [ ] Si avvia il timer di Sistema
- [ ] Si avvia il timer dell'algoritmo
- [ ] Si avvia il timer dell'oggetto StopWatch

07. Il metodo getElapsedTime() della class StopWatch:

- [ ] Misura il tempo trascorso dall'avvio della macchina
- [ ] Misura il tempo trascorso fra due eventi in secondi
- [ ] Misura il tempo trascorso fra due eventi in millisecondi
- [ ] Misura il tempo trascorso dalla new della classe StopWatch

08. La classe StopWatch vista a lezione:

- [ ] Serve ad implementare un orologio grafico
- [ ] Serve a fermare ed avviare un algoritmo
- [ ] Serve a migliorare le prestazioni di un algoritmo
- [ ] Serve a misurare il tempo fra due eventi

Lezione 072

01. La funzione di complessità "T(n)=3*n^2+log(n)+5" ha complessità asintotica:

- [ ] Lineare
- [ ] Logaritmica
- [ ] Quadratica
- [ ] Costante

02. Cosa possiamo concludere sulla funzione di complessità "T(n)=n+1000"?

- [ ] La complessità asintotica è lineare
- [ ] La complessità può essere costante per qualche n
- [ ] La complessità è costante
- [ ] La complessità è sempre lineare

03. La funzione di complessità "T(n)=log(n)+3n+5" ha complessità asintotica:

- [ ] è O(log(n))
- [ ] è O(n+log(n))
- [ ] è O(n)
- [ ] è O(n^2)

04. La funzione di complessità "T(n)=n^3+7n+2" ha complessità asintotica:

- [ ] Cubica
- [ ] Costante
- [ ] Lineare
- [ ] Quadratica

05. Quali tipi di complessità esistono?

- [ ] La complessità del caso peggiore
- [ ] La complessità asintotica e quella lineare
- [ ] La complessità del caso peggiore, del caso migliore e del caso medio
- [ ] La complessità del caso peggiore e del caso migliore

06. Cosa si può concludere circa la complessità asintotica di un doppio ciclo for innestato?

- [ ] La sua complessità è O(n^2)
- [ ] La sua complessità è sempre quadratica
- [ ] Se entrambe le variabili di ciclo variano da 0 ad n, la sua complessità è n^2
- [ ] La sua complessità è lineare

07. Cosa è falso se riferito alla complessità di un doppio ciclo for innestato?

- [ ] La sua complessità è O(n^2)
- [ ] La sua complessità dipende dal numero di cicli
- [ ] La sua complessità dipende dal numero di operazioni
- [ ] La sua complessità massima è O(n^2)

08. Cosa è falso se riferito alla complessità di un triplo ciclo for innestato?

- [ ] La sua complessità massima è O(n^3)
- [ ] La sua complessità dipende dal numero di cicli
- [ ] La sua complessità dipende dal numero di operazioni
- [ ] La sua complessità è O(n^3)

09. Dato un certo problema P e due algoritmi A e B che lo risolvono esntrambi ma in modo diverso, come faccio a scegliere il migliore fra i due? Discutere la problematica in termini di complessità degli algoritmi e fornire un esempio usando la notazione O-grande.

10. Dato l'array "int[][] a = new int[n][n];" scrivere il codice per inizializzare i suoi elementi tutti al valore 5 e poi calcolarne la funzione di complessità asintotica.

Lezione 073

01. Come si fa ad avviare un nuovo Thread in Java?

- [ ] Si usa il metodo thread()
- [ ] Si usa il metodo avvia()
- [ ] Si usa il metodo start()
- [ ] Si usa il metodo run()

02. Un Thread in Java:

- [ ] Viene eseguito all'interno del contesto di esecuzione di un processo (o programma)
- [ ] Viene eseguito all'esterno del contesto di esecuzione di un processo (o programma)
- [ ] Viene eseguito in autonomia da un file .exe
- [ ] Viene eseguito in parallelo del contesto di esecuzione di un processo (o programma)

03. Per implementare un Thread in Java:

- [ ] Si estende la classe "Execute" oppure si implementa l'interfaccia "Thread"
- [ ] Si estende la classe "Thread" oppure si implementa l'interfaccia "Runnable"
- [ ] Si estende la classe "Runnable" oppure si implementa l'interfaccia "Thread"
- [ ] Si estende l'interfaccia "Thread" oppure si implementa l'interfaccia "Thread"

04. All'interno di un programma Java:

- [ ] E' possibile eseguire più di un thread
- [ ] E' possibile eseguire un solo thread
- [ ] E' possibile eseguire solo thread multipli dello stesso tipo
- [ ] Non è possibile eseguire un solo thread ma più di uno

05. Le principali fasi di un Thread sono:

- [ ] Creazione, esecuzione, sospensione e terminazione
- [ ] Creazione, dedalock e cancellazione
- [ ] Sospensione, terminazione, cancellazione
- [ ] Cancellazione, creazione, esecuzione continua

06. Due Thread Java possono essere eseguiti contemporaneamente?

- [ ] Vero ma solo se sono Thread con metodi run() vuoti
- [ ] Vero, basta avviarli uno dopo l'altro
- [ ] Dipende da quante CPU ha la macchina su cui vengono eseguiti
- [ ] Falso non possono andare insieme in esecuzione

07. Se T1 è un Thread Java, l'invocazione al metodo T1.stop():

- [ ] Serve a fermare l'esecuzione di T1
- [ ] Causa lo stop della Java Virtual Machine
- [ ] Serve a sospendere temporaneamente l'esecuzione di T1
- [ ] Serve a chiudere T1 e tornare al Sistema Operativo

08. Il metodo suspend() della classe Thread:

- [ ] Blocca l'esecuzione del Thread in attesa di succesiva resume
- [ ] Blocca l'esecuzione del Thread in attesa di succesiva rerun
- [ ] Blocca l'esecuzione del Thread in attesa di succesiva start
- [ ] Cancella l'esecuzione del Thread in attesa di succesiva resume

09. Descrivere cosa sono e a che servono i Thread in Java e illustrare le modalità per crearli in termini di codice Java.

10. Usando il codice Java scrivere una semplice classe Thread che stampi 1000 volte il nome del proprio genere musicale preferito a video.

Lezione 074

01. Nel formato DateFormat.SHORT, la data 4 Luglio del 2021 viene formattata come:

- [ ] 04/07/21
- [ ] data di domenica, 4 luglio 2021
- [ ] 4 luglio 2021
- [ ] domenica 4 luglio 2021

02. Data l'istruzione Java "Date d = new Date(long n);":

- [ ] Crea una data a partire dall'anno 0 + "n" giorni
- [ ] Crea una data a partire da numeri lunghi
- [ ] Crea una data a partire dal 1 gennaio 1970 + "n" millisecondi
- [ ] Crea una data a partire dall'anno "n"

03. La classe SimpleDateFormat di Java:

- [ ] Permette di convertire stringhe in date e viceversa
- [ ] Converte stringhe in intervalli temporali
- [ ] Permette di usare solo date corte
- [ ] Converte date semplici dal formato inglese a quello italiano

04. La casse Date di Java è nel package:

- [ ] java.calendar
- [ ] java.lang
- [ ] java.io
- [ ] java.util

05. La classe Calendar di Java:

- [ ] Permette di ottenere l'ora di sistema
- [ ] Si usa per ottenere un oggetto calendario contenente tutte le date
- [ ] Permette di effettuare operazioni numeriche con le date
- [ ] Aggiorna il calendario del Sistema Operativo

06. Il metodo "Date parse(String data)" della classe DateFormat:

- [ ] Scrive una rappresentazione stringa di una data in output
- [ ] Converte una stringa di una data in millisecondi
- [ ] Converte una rappresentazione stringa di una data in una istanza della classe Date
- [ ] Converte una rappresentazione di data Date in una istanza della classe String

07. Il metodo "String format(Date data)" della classe DateFormat:

- [ ] Converte una istanza di Date in una stringa
- [ ] Converte una istanza di Millisecondi in una stringa
- [ ] Converte una istanza di Minuti in una stringa
- [ ] Converte una istanza di Ore in una stringa

08. La classe Date in Java:

- [ ] Si usa per gestire flussi di dati temporali dallo standard input
- [ ] Si usa per modificare il tempo di esecuzione
- [ ] Si usa per gestire l'orologio del PC
- [ ] Si usa per gestire variabili che contengono istanti temporali

09. Descrivere che cos'è e come si usa il metodo System.currentTimeMillis() e fornire un esempio in codice Java.

10. Descrivere le differenze principali tra le classi Date, Calendar e SimpleDateFormat.

Lezione 075

01. La classe Permission fa parte del package:

- [ ] java.runnable
- [ ] java.io
- [ ] java.thread
- [ ] java.security

02. Quale fra le seguenti è la definizione corretta per la classe Permission?

- [ ] E' una classe astratta che serve a rappresentare le varie modalità di esecuzione delle risorse del sistema
- [ ] E' una classe astratta che serve a rappresentare le varie modalità di accesso alle risorse del sistema
- [ ] E' una interfaccia che serve a rappresentare le varie modalità di accesso alle risorse del sistema
- [ ] E' una classe che serve a rappresentare le varie modalità di esecuzione delle risorse del sistema

03. La classe Java Security Manager:

- [ ] Consente a un'applicazione di determinare, a tempo di esecuzione, se un metodo fa parte di una classe
- [ ] Consente a un'applicazione di determinare, a tempo di compilazione, se un'operazione può essere eseguita
- [ ] Consente a un'applicazione di eseguire un'operazione di sicurezza
- [ ] Consente a un'applicazione di determinare, prima di eseguire un'operazione, se può essere eseguita in sicurezza

04. L'utilizzo del Java Security Manager:

- [ ] Non puo' controllare l'autorizzazione per l'esecuzione di un modulo se non a run time una volta eseguito
- [ ] Non puo' controllare l'esecuzione di un modulo prima dell'esecuzione
- [ ] Puo' consentire o meno l'esecuzione di un modulo
- [ ] Non puo' controllare l'autorizzazione per l'esecuzione di un modulo

05. La classe SecurityManager fa parte del package:

- [ ] java.util
- [ ] java.lang
- [ ] java.io
- [ ] java.start

06. I metodi di controllo della classe SecurityManager:

- [ ] Iniziano con la parola "check"
- [ ] Iniziano con la parola "runnable"
- [ ] Iniziano con la parola "verify"
- [ ] Iniziano con la parola "try"

07. I metodi della classe SecurityManager:

- [ ] Vengono eseguiti prima della compilazione
- [ ] Vengono chiamati solo dall'utente
- [ ] Vengono chiamati dietro le quinte dalle librerie Java prima di eseguire operazioni di classi potenzialmente pericolose
- [ ] Vengono verificati prima della compilazione

08. Il metodo checkAccept() della classe SecurityManager permette di controllare:

- ☐ Se si è abilitati ad accettare connessioni di rete in ingresso da un particolare host e porta
- ☐ Se si è abilitati a riscrivere le connessioni di rete in ingresso da un particolare host e porta
- ☐ Se si è abilitati a modificare le connessioni di rete in ingresso da un particolare host e porta
- ☐ Se si è abilitati a forwardare connessioni di rete in ingresso da un particolare host e porta

09. Descrivere che cos'è il Security Manager in Java e a che cosa serve.

Lezione 076

01. L'invocazione della seguente funzione "Runtime.getRuntime().exec("shutdown -s -t 0");":

- [ ] Blocca l'esecuzione di un thread
- [ ] Esegue un comando che accende immediatamente una macchina Unix
- [ ] Blocca il comando che accende immediatamente una macchina Unix
- [ ] Esegue un comando che spegne immediatamente una macchina Unix

02. La classe Runtime in Java:

- [ ] E' una interfaccia che permette di eseguire programmi nativi esterni
- [ ] E' una classe appartenente al package java.io e permette di eseguire programmi nativi esterni
- [ ] E' una classe appartenente al package java.lang e permette di eseguire programmi nativi esterni
- [ ] E' una classe appartenente al package java.util e permette di eseguire programmi nativi esterni

03. Che cos'è la JNI (Java Native Interface) di Java?

- [ ] E' un comando che permette di ottenere informazioni sull'ambiente nativo
- [ ] E' il sistema che permette di eseguore nativamente i file .class
- [ ] E' un comando del Sistema Operativo che permette di eseguire codice Java
- [ ] E' un sistema che permette l'integrazione bidirezionale fra Java e l'ambiente nativo

04. La parola chiave "native", usata nelle Java Native Interface:

- [ ] Definisce una sorta di metodo astratto che non verrà implementato in Java
- [ ] Definisce un metodo solitamente dichiarato in una classe astratta
- [ ] Definisce una sorta di metodo astratto che verrà implementato in Java
- [ ] Definisce un metodo che sarà implementato in Java puro

05. Il metodo "Runtime.getRuntime().availableProcessors()" consente di:

- [ ] Ottenere informazioni sull'ambiente di esecuzione della Java Virtual Machine
- [ ] Ottenere informazioni sull'ambiente di runtime
- [ ] Ottenere informazioni sul Sistema Operativo di esecuzione
- [ ] Ottenere informazioni sul numero di processori della macchina

06. Cosa permette di fare la JNI (Java Native Interface) di Java?

- [ ] Permette il passaggio di parametri da Java verso altre macchine in rete
- [ ] Permette di migliorare la compilazione
- [ ] Permette il passaggio di parametri da Java verso l'ambiente nativo e viceversa
- [ ] Permette il passaggio di parametri da Java verso l'utente finale

07. Cosa spinge i programmatori ad usare codice nativo?

- [ ] La necessità di gestire direttamente dispositivi hardware
- [ ] La necessità di avere codice proprietario
- [ ] La necessità di gestire direttamente le classi Java
- [ ] La necessità di gestire direttamente il processo di compilazione

08. L'istruzione "System.out.println(Runtime.getRuntime().availableProcessors())":

- [ ] Stampa true se il processore è installato sulla macchina
- [ ] Stampa il numero di processori disponibili sulla macchina
- [ ] Stampa il numero di processi disponibili sulla macchina
- [ ] Stampa il numero seriale del processore disponibile sulla macchina

09. Descrivere che cos'è e come si usa la classe Runtime di Java.

10. Descivere a cosa serve la Java Native Interface.

Lezione 077

01. Cosa consente di fare il Garbage Collector di Java?

- [ ] Liberare memoria secondaria distruggendo i dati non più utilizzati
- [ ] Liberare spazio sull'hard disk
- [ ] Liberare il floppy dai dati non più utilizzati
- [ ] Liberare memoria heap distruggendo i dati non più utilizzati

02. Per accedere alle proprietà di ambiente del Sistema Operativo si utilizza:

- [ ] La classe Output
- [ ] La classe System
- [ ] La classe Runnable
- [ ] La classe MainRuntime

03. Come viene invocato il Garbage Collector di Java?

- [ ] Il programmatore non può invocarlo direttamente
- [ ] Viene invocato con una chiamata al Sistema Operativo
- [ ] Viene invocato mandando un messaggio alla Java Virtual Machine
- [ ] Viene invocato con System.gc()

04. La classe System consente di gestire metodi per:

- [ ] La connessione tra dispositivi
- [ ] Lo standard per la rappresentazione numerica binaria
- [ ] Lo standard output e lo standard input
- [ ] La comunicazione sulla rete Internet

05. La classe System appartiene al package:

- [ ] java.io, che va importato per usare la classe
- [ ] java.lang, che non va importato per usare la classe
- [ ] java.thread, che va importato per usare la classe
- [ ] java.util, che va importato per usare la classe

06. Il metodo "static String getenv(String name)" della classe System:

- [ ] Permette di leggere dal Sistema Operativo il valore della variabile d'ambiente specificata come parametro
- [ ] Permette di cancellare dal Sistema Operativo il valore della variabile d'ambiente specificata come parametro
- [ ] Permette di scrivere nel Sistema Operativo il valore della variabile d'ambiente specificata come parametro
- [ ] Permette di socrascrivere dal Sistema Operativo il valore della variabile d'ambiente specificata come parametro

07. Il metodo "arraycopy(Object src, int srcPos, Object dest, int destPos, int length)" della classe System consente di:

- [ ] Copiare un array di lunghezza length da una sorgente (src) verso una variabile di destinazione (dest)
- [ ] Copiare un certo numero di elementi (length) da un array sorgente (src) verso un array di destinazione (dest)
- [ ] Copiare un certo numero di elementi (length) da un array sorgente (src) verso un array di destinazione (dest) della stesa dimensione
- [ ] Copiare un Oggetto sorgente (src) di lunghezza length verso un Oggetto destinazione (dest)

08. La classe System estende:

- [ ] L'Interfaccia Thread
- [ ] La classe Object
- [ ] Il package java.lang
- [ ] L'Interfaccia Runnable

09. Descrivere a che serve e come funziona il Garbage Collector di Java.

Lezione 080

01. I tag di blocco di Javadoc hanno la forma:

- [ ] <nometag></nometag>
- [ ] /** nometag */
- [ ] // nometag
- [ ] @nometag

02. Il tool Javadoc di Java consente di:

- [ ] Trasformare la documentazione in una serie di file .java
- [ ] Trasformare la documentazione in un minisito web in formato HTML
- [ ] Trasformare la documentazione in un report in formato excel
- [ ] Trasformare la documentazione in una presentazione in formato power point

03. I commenti nel codice Java:

- [ ] Si possono fare solo su più linee
- [ ] Si possono fare solo su una linea
- [ ] Non si possono fare
- [ ] Si possono fare su una o su più linee

04. La specifica Javadoc consente di:

- [ ] Commentare il codice usando testo, tag HTML e tag di blocco
- [ ] Commentare codice usando solo il linguaggio HTML
- [ ] Commentare codice usando un linguaggio proprietario della libreria Javadoc
- [ ] Commentare il codice usando solo il testo

05. Nel codice Java i commenti multilinea hanno la forma:

- [ ] /** COMMENTO **/
- [ ] /** COMMENTO */
- [ ] // COMMENTO
- [ ]

06. Per dire a Javadoc di generare, nella documentazione di una classe, un collegamento alla documentazione di un'altra classe, si usa il tag di blocco:

- [ ] @connect NomeClasse
- [ ] @link NomeClasse
- [ ] @doc NomeClasse
- [ ] @goto NomeClasse

07. In Javadoc, per generare l'autore nella documentazione di una classe, si usa il tag di blocco:

- [ ] @programmer
- [ ] @person
- [ ] @name
- [ ] @author

08. Il tool Javadoc di Java:

- ☐ Serve per consultare la documentazione del codice a partire dai suoi commenti
- ☐ Serve per sovrascrivere la documentazione del codice a partire dai suoi commenti
- ☐ Serve per generare la documentazione del codice a partire dai suoi commenti
- ☐ Serve per salvare la documentazione del codice a partire dai suoi commenti

09. Descrivere il tool Javadoc di Java, spiegare a cosa serve e fornire un esempio in codice Java.

Lezione 081

01. Descrivere le principali fasi di Unified Process.

02. Descrivere le differenze tra il modello di Progettazione a Cascata e il Modello Incrementale.

Lezione 082

01. Usando un modello scritto nel linguaggio UML si può:

- [ ] Generare automaticamente il comportamento di un sistema hardware
- [ ] Generare automaticamente la documentazione di un progetto software
- [ ] Generare automaticamente la struttura di un sistema hardware
- [ ] Generare automaticamente la struttura di un codice sorgente Java

02. Qual è la finalità della modellazione UML?

- [ ] Specificare la struttura e il comportamento di un sistema software in maniera precisa, completa e senza ambiguità
- [ ] Specificare il comportamento di un programmatore in maniera precisa, completa e senza ambiguità
- [ ] Specificare la struttura di un linguaggio di programmazione in maniera precisa, completa e senza ambiguità
- [ ] Specificare la struttura e il comportamento di un sistema scrivendo in maniera precisa, completa e senza ambiguità il suo codice sorgente

03. A cosa serve il linguaggio UML?

- [ ] Ad effettuare la modellazione di un sistema software
- [ ] Ad effettuare la programmazione in codice C++, Java o PHP
- [ ] Ad effettuare la modellazione del codice
- [ ] Ad effettuare la modellazione della sintassi del codice

04. Cosa vuol dire l'acronimo UML?

- [ ] Unique Model Language
- [ ] Unique Modeling Language
- [ ] User Model Language
- [ ] Unified Modeling Language

05. Con quale scopo si modella un sistema informativo usando il linguaggio UML?

- [ ] Solo al fine di documentare e testare il codice
- [ ] Al fine di documentare il codice, poi si usa un linguaggio di programmazione
- [ ] Al fine di guidare tutte le fasi di un progetto, dalla progettazione all'implementazione
- [ ] Esclusivamente al fine di documentare il codice preesistente

06. A cosa servono i Diagrammi UML?

- [ ] Sono dei Diagrammi per rappresentare solo il comportamento dell'applicazione
- [ ] Sono dei Diagrammi per rappresentare dei vari aspetti dell'applicazione
- [ ] Sono dei Diagrammi che servono a programmare in Java
- [ ] Sono dei Diagrammi per rappresentare il codice sorgente

07. Quale fra le seguenti risposte descrive meglio gli aspetti che si possono modellare con il linguaggio UML?

- [ ] Astrazione, Incapsulamento, Ereditarietà e Polimorfismo
- [ ] Aspetti legati al comportamento di un sistema che niente hanno a che fare col codice
- [ ] Astrazione, Incapsulamento ed Ereditarietà
- [ ] Gli aspetti più semplici di un linguaggio di programmazione ad oggetti

08. UML è indipendente dal linguaggio di programmazione?

- [ ] Vero
- [ ] Falso
- [ ] Falso, tranne in alcuni casi
- [ ] Vero, tranne in alcuni casi

09. Descrivere che cos'è UML e quali sono i principali tipi di diagrammi.

Lezione 083

01. Le classi in un Diagramma delle Classi di UML:

- [ ] Contengono informazioni temporali
- [ ] Rappresentano gli stati del sistema
- [ ] Si rappresentano con dei rettangoli
- [ ] Si rappresentano con delle ellissi

02. In UML il Diagramma degli Oggetti contiene:

- [ ] Le interazioni fra le classi e gli oggetti esterni
- [ ] Le possibili classi del sistema
- [ ] Le possibili istanze delle classi del sistema
- [ ] Le classi e le interfacce da implementare per il sistema

03. In UML il Diagramma delle Classi rappresenta:

- [ ] I componenti di interfaccia tra le classi
- [ ] I componenti software di una applicazione
- [ ] Solo le classi astratte
- [ ] Solo le classi implementate

04. In UML a cosa servono i Diagrammi dei Casi d'Uso?

- [ ] A rappresentare le interazioni fra le classi del sistema
- [ ] A testare gli oggetti e gli utenti del sistema da implementare
- [ ] Ad effettuare un test delle istanze del sistema
- [ ] A rappresentare le interazioni fra gli utenti e le funzionalità dl sistema

05. Nei Diagrammi dei Casi d'Uso di UML, a cosa serve l'opzione "<<extends>>"?

- [ ] Serve a specificare che una funzione richiama sicuramente un'altra funzione
- [ ] Serve a specificare che una funzione estende una classe del sistema modellato
- [ ] Serve a specificare che una funzione, per essere eseguita, deve prima invocare un'altra funzione
- [ ] Serve a specificare che una funzione può essere richiamata da un'altra funzione

06. Nei Diagrammi di Casi d'Uso, a cosa serve l'opzione "<<includes>>"?

- [ ] Serve ad evitare che, prima che una funzione possa essere eseguita, sia eseguita la funzione inclusa
- [ ] Serve ad imporre che, prima che una funzione possa essere eseguita, deve eseguita prima la funzione inclusa
- [ ] Serve a ricordare che, prima che una funzione possa essere eseguita, si può eseguire opzionalmente prima la funzione inclusa
- [ ] Serve a suggerire che, prima che una funzione possa essere eseguita, potrebbe dover essere eseguita prima la funzione inclusa

07. Quale delle seguenti affermazioni è falsa se riferita ad una classe in un Diagramma delle Classi di UML?

- [ ] E' possibile indicare soltanto il nome della classe
- [ ] E' possibile indicare solo nome e gli attributi della classe
- [ ] E' possibile indicare anche solo gli attributi, senza specificare un nome
- [ ] E' possibile indicare anche solo nome e metodi della classe

08. Nel Diagramma delle Classi di UML e' possibile rappresentare interfacce?

- [ ] No, perchè esistono solo in Java e non in UML
- [ ] No perchè sarebbero confuse con le classi
- [ ] No, perchè in UML non esiste il concetto di interfaccia
- [ ] Si, e può avere la forma di un elemento circolare

09. Descrivere a cosa serve il Diagramma dei Casi d'Uso UML e da quali elementi è costituito.

10. Descrivere le principali differenze tra il Diagramma delle Classi e il Diagramma degli Oggetti di UML.

Lezione 084

01. Qual è la differenza principale tra Diagrammi di Collaborazione e di Sequenza UML?

- [ ] La sequenza temporale dei messaggi è meno evidente che nel Diagramma di Collaborazione mentre sono più evidenti i legami tra gli oggetti
- [ ] La sequenza spaziale dei messaggi è meno evidente che nel Diagramma di Sequenza mentre sono più evidenti i legami tra gli oggetti
- [ ] La sequenza temporale dei messaggi è più evidente che nel Diagramma di Sequenza mentre non vi sono rappresentati gli oggetti
- [ ] La sequenza temporale dei messaggi è più evidente che nel Diagramma di Collaborazione mentre sono meno evidenti i legami tra gli oggetti

02. Un Diagramma degli Stati UML di un oggetto di tipo Lampadina potrebbe contenere:

- [ ] on, off, accendi(), spegni()
- [ ] crea(), svita(), avvita()
- [ ] new, delete(), public, private
- [ ] new, accendi(), avvita()

03. Un Diagramma degli Stati di UML:

- [ ] Descrive in quali stati può trovarsi l'istanza di una classe
- [ ] Descrive in quali stati può trovarsi il sistema
- [ ] Descrive in quali stati può trovarsi una classe
- [ ] Descrive in quali stati può trovarsi un utente del sistema o un sistema esterno

04. In un Diagramma degli Stati UML una condizione di guardia:

- [ ] E' una condizione che genererà un ciclo for in Java
- [ ] Sarà ignorata nella traduzione da UML a Java
- [ ] E' un gestore della sicurezza di una classe
- [ ] E' associata ad una transizione di stato

05. In un Diagramma degli Stati UML una transizione di stato:

- [ ] Indica lo stato di partenza
- [ ] Indica il passaggio dallo stato di stop a quello di run
- [ ] Indica lo stato di destinazione
- [ ] Indica il passaggio dallo stato di partenza a quello di destinazione

06. Un Diagramma delle Attività di UML:

- [ ] Rappresenta un insieme di operazioni sia automatiche che umane
- [ ] Rappresenta delle transazioni del sistema
- [ ] Raffigura le classi attive e le funzioni
- [ ] Raffigura gli ogetti attivi e le loro relazioni

07. In UML i Diagrammi di Sequenza:

- [ ] Descrivono l'ordinamento in memoria dei messaggi (invocazione di metodi) scambiati tra diversi oggetti
- [ ] Descrivono l'ordinamento temporale dei tipi di dati scambiati tra diversi oggetti
- [ ] Descrivono il contenuto semantico dei messaggi scambiati tra diversi oggetti
- [ ] Descrivono l'ordinamento temporale dei messaggi (invocazione di metodi) scambiati tra diversi oggetti

08. I Diagrammi di Collaborazione di UML:

- [ ] Descrivono classi che partecipano ad uno scambio di messaggi
- [ ] Descrivono come gli oggetti collaborano per allocare e liberare memoria
- [ ] Descrivono la struttura dei dati che vengono scambiati nei messaggi
- [ ] Descrivono gli oggetti che partecipano ad uno scambio di messaggi

09. Descrivere a cosa serve un Diagramma delle Attività (o Activity Diagram) UML e fornire un esempio.

10. Descrivere a cosa serve un Diagramma degli Stati UML e fornire un esempio.

Lezione 085

01. Se un attributo di una classe raffigurata in un Class Diagram UML è rappresentato sottolineato, cosa significa?

- [ ] Che non è ancora stato istanziato
- [ ] Che la sua visibilità è protected
- [ ] Che è un attributo a livello di classe, quindi static
- [ ] Che è una costante

02. Quale fra le seguenti è una rappresentazione errata per un attributo di classe in un Class Diagram UML?

- [ ] -stipendio:double
- [ ] String matricola;
- [ ] +nome:String
- [ ] #cognome:String

03. Quale fra le seguenti è una rappresentazione errata per una classe in un Class Diagram UML?

- [ ] Nome + attributi + metodi
- [ ] Nome + attributi + metodi + stati
- [ ] Nome + attributi
- [ ] Nome

04. A differenza di UML, nel linguaggio Java:

- [ ] L'ereditarietà multipla è tollerata solo in alcuni casi
- [ ] E' sempre possibile utilizzare l'ereditarietà multipla
- [ ] E' possibile per una classe estendere due o più classi
- [ ] Non è possibile usare l'ereditarietà multipla, se non usando le interfacce

05. Qual è la rappresentazione corretta della classe java.lang.System in un Diagramma delle Classi UML?

- [ ] Si usa solo "System"
- [ ] java:lang:System
- [ ] java.lang::System
- [ ] System:java.lang

06. Quali fra le seguenti frasi è falsa se riferita al linguaggio UML?

- [ ] E' sempre possibile estendere più di una classe alla volta
- [ ] E' alcune volte possibile estendere più di una classe alla volta
- [ ] E' possibile estendere una classe alla volta, ovvero possiamo avere più di una superclasse
- [ ] Non è possibile estendere più di una classe alla volta, ovvero non possiamo avere più di una superclasse

07. Da un Diagramma delle Classi di UML:

- [ ] E' possibile generare solo un elenco di metodi e attributi delle classi
- [ ] E' possibile generare programmi nativi eseguibili
- [ ] E' possibile generare il codice Java delle classi
- [ ] E' impossibile generare il codice Java delle classi

08. In un Diagramma delle Classi UML è possibile:

- [ ] Rappresentare che una istanza Direttore sovrascrive il metodo dati() (polimorfismo)
- [ ] Rappresentare che una istanza classe Direttore sovrascrive il metodo dati() (polimorfismo)
- [ ] Rappresentare che un Direttore è un Impiegato (generalizzazione) e ne sovrascrive il metodo dati() (polimorfismo)
- [ ] Rappresentare che una istanza di Impiegato sovrascrive il metodo dati() (polimorfismo)

09. Illustrare in che forma può essere descritta una classe in un Diagramma delle Classi UML. In particolare descrivere dove vengono rappresentati il nome della classe, i suoi attributi e i suoi metodi.

10. Illustrare come possono essere descritti i livelli di visibilità degli attributi di una classe in un Diagramma delle Classi UML.

Lezione 086

01. In un Diagramma delle Classi UML come si indica l'Aggregazione?

- [ ] Con una freccia doppia
- [ ] Con un rombo pieno
- [ ] Con una freccia
- [ ] Con un rombo vuoto

02. Quali caratteristiche specificano una associazione di un Diagramma delle Classi UML?

- [ ] Un nome ed una coppia di molteplicità
- [ ] Nome, molteplicità e navigabilità
- [ ] Un nome e un verso
- [ ] Solo un nome

03. Cosa significa per due classi essere in relazione tramite una associazione in un Diagramma delle Classi UML?

- [ ] Che ciascuna avrà un attributo del tipo dell'altra classe
- [ ] Che una è sottoclasse dell'altra
- [ ] Che usa l'altra come parametro o come tipo di ritorno di un metodo
- [ ] Che sono classi simili

04. Cosa cambia aggiungere un verso ad una associazione fra classi in un Diagramma delle Classi UML?

- [ ] Serve a specificare in quale classe si troverà il riferimento all'altra classe
- [ ] Il verso serve a specificare la molteplicità dell'associazione
- [ ] Serve ad indicare la superclasse
- [ ] Niente, l'associazione avrà lo stesso significato di quella senza verso

05. A che serve una Classe di Associazione (o associativa) in un Diagramma delle Classi UML?

- [ ] A creare una associazione fra due classi
- [ ] A caratterizzare una associazione fra due classi
- [ ] A definire la navigabilità di una associazione fra classi
- [ ] A specificare la cardinalità di una associazione fra classi

06. In un Class Diagram UML, le cardinalità di un'Associazione:

- [ ] Dicono quante volte può essere chiamato un metodo
- [ ] Specificano quante sottoclassi partecipano all'associazione
- [ ] Specificano quante istanze delle classi partecipano all'associazione
- [ ] Servono a specificare quanti elementi ha un array

07. Cosa è falso dire relativamente alle differenze fra Aggregazione e Composizione di un Diagramma delle Classi UML?

- [ ] Le classi che partecipano alla prima hanno significato solo in relazione con il tutto, mentre quelle che partecipano alla seconda hanno significato anche da sole
- [ ] La prima è una relazione debole, la seconda una relazione forte
- [ ] Le classi che partecipano alla prima sono indipendenti, mentre quelle che partecipano alla seconda non lo sono
- [ ] La prima è caratterizzata da un rombo vuoto, la seconda da un rombo pieno

08. In un Diagramma delle Classi UML come si indica la Composizione?

- [ ] Con un rombo vuoto
- [ ] Con una freccia doppia
- [ ] Con una freccia
- [ ] Con un rombo pieno

09. Descrivere quali tipi di associazioni possiamo definire fra classi in un Diagramma delle Classi UML e che forma hanno.

10. In un Diagramma delle Classi UML siano date due classi, Persona ed Automobile, fra cui sia indicata una normale associazione (senza nessuna freccia, cardinalità, ecc). Mostrare una possibile implementazione in codice Java delle due classi.

Lezione 088

01. Il tool ArgoUML:

- [ ] E' un insieme di metodi di Argomenti
- [ ] E' un insieme di argomenti per UML
- [ ] E' un IDE open source per compilare
- [ ] E' un IDE open source orientato a UML

02. Il sistema Rational Rose:

- [ ] E' un insieme di metodi
- [ ] E' un IDE per UML
- [ ] E' una rosa razionale di metodi
- [ ] E' un insieme di classi

03. A che serve ArgoUML?

- [ ] Supporta la progettazione, lo sviluppo e la documentazione di applicazioni
- [ ] Compila il codice e trova errori
- [ ] Supporta solo la documentazione di applicazioni
- [ ] Supporta la compilazione, lo sviluppo e l'esecuzione di applicazioni

04. Il software ArgoUML è:

- [ ] Un software scritto in UML che supporta la progettazione UML
- [ ] Un software scritto in Java che supporta la progettazione UML
- [ ] Un software scritto in pseudocodice che supporta la progettazione UML
- [ ] Un software scritto in HTML che supporta la progettazione UML

05. Poseidon UML Community edition:

- [ ] E' un IDE che consente di progettare in UML in modo grafico
- [ ] E' un IDE che consente di progettare in UML in modo automatico
- [ ] E' un IDE che consente di progettare in UML in modo testuale
- [ ] E' un IDE che consente di scrivere in UML in modo vocale

06. Gli IDE dedicati alla progettazione nel linguaggio UML:

- [ ] Scrivono codice in modo semiautomatico
- [ ] Generano codice in modo automatico
- [ ] Hanno un repository di codice generico riutilizzabile
- [ ] Scrivono codice in modo automatico

07. Che differenza c'è tra un IDE dedicato a Java e uno dedicato a UML?

- [ ] Un IDE Java genera codice in Java, un IDE UML permette di sviluppare in UML
- [ ] Un IDE Java genera codice in Java, un IDE UML lo genera in UML
- [ ] Un IDE Java serve a scrivere codice in Java, un IDE UML genera codice anche in Java
- [ ] Un IDE Java serve a scrivere codice in Java, un IDE UML genera codice UML

08. Che cos'è un IDE per UML?

- [ ] Un sistema grafico per supportare la definizione di metodi
- [ ] Un sistema grafico per supportare la costruzione solo di attributi
- [ ] Un sistema grafico per supportare l'istanza di metodi UML
- [ ] Un sistema grafico per supportare la costruzione di diagrammi

Lezione 089

01. Il metodo "setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);" della classe JFrame di Swing:

- [ ] Indica che, in caso di uscita dalla JVM, il JFrame viene chiuso
- [ ] Provoca la terminazione del programma
- [ ] Indica che, in caso di chiusura del frame, termina il programma
- [ ] Provoca lo spegnimento del monitor

02. Le API grafiche di Java servono a:

- [ ] Definire la grafica del sistema operativo
- [ ] Definire classi grafiche
- [ ] Definire i grafici di excel
- [ ] Definire interfacce grafiche

03. Il metodo "setVisible(true);" della classe JFrame di Swing si usa per:

- [ ] Rendere visibile un frame
- [ ] Rendere visibile un programma
- [ ] Rendere visibile una istanza
- [ ] Rendere visibile una classe

04. Il metodo "setSize(300, 100);" della classe JFrame di Swing serve a settare:

- [ ] La dimensione dei punti del mouse
- [ ] La dimensione della finestra grafica
- [ ] La dimensione del monitor
- [ ] Il numero di oggetti

05. Nella tecnologia Swing una interfaccia utente:

- [ ] è composta da una Interface o una finestra di dialogo (JDialog)
- [ ] è composta da un frame (JFrame) o una finestra di dialogo (JDialog)
- [ ] è composta da un frame (JFrame) o una Interface
- [ ] è composta da un Thread o una finestra di dialogo (JDialog)

06. Le API di Java supportano la definizione di interfacce grafiche mediante:

- [ ] AWL e Swing
- [ ] ASD e String
- [ ] AWT e Swing
- [ ] AWT e String

07. L'acronimo GUI significa:

- [ ] Generalized User Impact
- [ ] Graphical User Integration
- [ ] General User Interface
- [ ] Graphical User Interface

08. Le API sono:

- [ ] Librerie utili dette Advanced Programmer Interface
- [ ] Librerie utili dette Application Programmer Interface
- [ ] Librerie utili dette Advanced Pilot Interface
- [ ] Librerie utili dette Advanced Pilot Interrupt

09. Descrivere le differenze tra una applicazione Java a riga di comando ed una a finestre. Mostrare un esempio minimale di entrambe in codice Java.

Lezione 090

01. Quale fra i seguenti non è un componente della libreria Swing?

- [ ] JArrow
- [ ] JButton
- [ ] JFrame
- [ ] JTextField

02. Cosa è falso se riferito a JComponent di Swing?

- [ ] Permette il layout automatico dei componenti
- [ ] E' una classe che fornisce le funzionalità di base degli altri componenti
- [ ] E' la superclasse di ogni componente visuale Swing
- [ ] E' una classe che si compone di altre classi

03. Un Tooltip della libreria Swing:

- [ ] Serve a progettare componenti Swing
- [ ] E' un suggerimento visivo disegnato per spiegare a che serve un componente
- [ ] Serve a visualizzare componenti Swing
- [ ] E' un abbellimento visivo che disegna i bordi di un componente

04. Un componente JPanel di Swing:

- [ ] Serve a visualizzare delle immagini in movimento in una finestra
- [ ] Serve a minimizzare o massimizzare una finestra
- [ ] Serve a pannellare lo sfondo di una finestra con un disegno ripetuto
- [ ] Serve a contenere altri componenti

05. Cosa è falso se riferito al componente JButton di Swing:

- [ ] Permette di intercettare i click del mouse
- [ ] Visivamente è disegnato come un pulsante
- [ ] La sua reazione ai click del mouse è codificata dalla libreria Swing
- [ ] La sua reazione ai click del mouse è personalizzabile da parte del programmatore

06. Cosa è falso se riferito al componente JLabel di Swing?

- [ ] Serve a visualizzare una breve stringa
- [ ] Serve a visualizzare dei messaggi
- [ ] Può visualizzare un'immagine
- [ ] Serve ad interagire con l'utente

07. A cosa serve il componente JTextField?

- [ ] Serve principalmente a rilevare i click del mouse
- [ ] Serve a leggere stringhe di testo immesse dall'utente
- [ ] Serve a visualizzare immagini
- [ ] Serve a far leggere lunghi messaggi non modificabili all'utente

08. A cosa serve il componente JTextArea?

- [ ] Serve a visualizzare del testo potenzialmente molto lungo
- [ ] E' l'area principale della finestra
- [ ] Serve principalmente a rilevare i click del mouse
- [ ] Serve a visualizzare l'area del Desktop

09. Descrivere cosa sono e a cosa servono le classi JButton, JLabel, JTextField e JTextArea.

Lezione 091

01. Il metodo add() di JPanel della libreria Swing:

- [ ] Permette di aggiungere nuovi componenti al pannello
- [ ] Aggiunge un nuovo parametro al pannello
- [ ] Permette di specificare le dimensioni del pannello
- [ ] Aggiunge una nuova classe al pannello

02. Cosa è falso se riferito ad un GridLayout di Swing?

- [ ] Dispone i componenti secondo una matrice, le cui dimensioni si specificano nel costruttore
- [ ] Possiamo specificare il numero di righe e di colonne su cui disporre i componenti
- [ ] Dispone i componenti secondo una griglia a due dimensioni
- [ ] Dispone i componenti secondi i punti cardinali

03. Quando si aggiunge un componente ad un Container di Swing:

- [ ] La sua posizione sarà decisa dall'utente
- [ ] La sua posizione sarà decisa dal costruttore
- [ ] La sua posizione sarà decisa dalla Java Virtual Machine
- [ ] La sua posizione sarà decisa dal gestore di layout del Container

04. Le classi dei gestori di layout della libreria Swing:

- [ ] Appartengono al package javax.swing
- [ ] Appartengono al package java.lang
- [ ] Appartengono al package java.awt
- [ ] Appartengono al package java.io

05. A cosa serve un gestore di layout della libreria Swing?

- [ ] A decidere quali componenti aggiungere ad una finestra
- [ ] A stabilire il comportamento di un JButton
- [ ] A stabilire cosa accade quando un JFrame viene chiuso
- [ ] A posizionare automaticamente i componenti in un contenitore

06. Il gestore di layout di default di un JFrame è:

- [ ] JFrameLayout
- [ ] BorderLayout
- [ ] GridLayout
- [ ] FlowLayout

07. Cosa fa un FlowLayout di Swing?

- [ ] Dispone i componenti in modo casuale
- [ ] Dispone i componenti in modo naturale, uno dopo l'altro
- [ ] Dispone i componenti secondo una griglia bidimensionale
- [ ] Dispone i componenti secondo i punti cardinali

08. Cosa è falso se riferito ad un BorderLayout di Swing?

- [ ] Dispone i componenti al centro, a nord, a sud, ad est e ad ovest
- [ ] Dispone i componenti secondo una griglia a due dimensioni
- [ ] Dispone i componenti secondo i 4 punti cardinali o al centro
- [ ] Dispone i componenti in 5 posizioni

09. Descrivere a cosa serve e come si usa un BorderLayout.

10. Descrivere cosa i gestori di layout e quali sono e le principali differenze tra BorderLayout e GridLayout.

Lezione 092

01. Descrivere come si fa a scrivere del codice che venga eseguito alla pressione di un pulsante JButton.

Lezione 093

01. Il metodo showConfirmDialog() di JOptionPane della libreria Swing:

- [ ] Creare una finestra di dialogo con i pulsanti "si" e "no"
- [ ] Permette di visualizzare messaggi di errore
- [ ] Chiede all'utente se vuole chiudere l'applicazione e tornare al Sistema Operativo
- [ ] Chiede all'utente di confermare a voce la sua scelta

02. Cosa è falso se riferito ad una finestra di dialogo di Swing?

- [ ] Una finestra di dialogo visualizza un messaggio per l'utente
- [ ] Una finestra di dialogo blocca l'interazione dell'utente col JFrame sottostante
- [ ] Una finestra di dialogo è un componente audio
- [ ] Una finestra di dialogo è modale

03. A che serve una finestra di dialogo di Swing?

- [ ] A nascondere un errore dell'applicazione
- [ ] A leggere comandi vocali dall'utente
- [ ] A definire le dimensioni di una finestra
- [ ] A visualizzare un messaggio o a formulare una domanda

04. Che significa che una finestra di dialogo di Swing è modale?

- [ ] Che blocca le interazioni dell'utente con la finestra principale
- [ ] Che permette di interagire con la finestra principale solo nella modalità dialogo
- [ ] Che definisce il modo in cui una finestra viene chiusa
- [ ] Che definisce il modo in cui l'applicazione dialoga con l'utente

05. La classe JOptionPane di Swing:

- [ ] Permette di definire finestre di tipo JFrame
- [ ] Serve a creare pannelli opzionali
- [ ] Serve a definire le opzioni principali di una applicazione a finestre
- [ ] Permette di definire semplici finestre di dialogo

06. Il metodo showMessageDialog() di JOptionPane della libreria Swing:

- [ ] Mostra la finestra JFrame corrente quando essa è nascosta
- [ ] Permette di creare finestre di dialogo
- [ ] Permette di dialogare con l'utente tramite messaggi vocali
- [ ] Permette di visualizzare messaggi sulla console di testo

07. Quali tipi di JDialog permette di creare il metodo JOptionPane.showMessageDialog() di Swing?

- [ ] solo di tipo messaggio
- [ ] domanda, risposta, si, no
- [ ] domanda, informazione, avvertimento, errore, messaggio
- [ ] solo di tipo errore

08. L'opzione JOptionPane.YES_NO_OPTION del metodo JOptionPane.showConfirmDialog() di Swing:

- [ ] Permette all'utente di scegliere "si", "no" oppure di chiudere il JDialog
- [ ] Permette all'utente di rispondere solo "si" o "no"
- [ ] Permette all'utente di scegliere se chiudere oppure no l'applicazione
- [ ] Permette all'utente di tornare oppure no al Sistema Operativo

09. Descrivere cosa sono, come funzionano e a cosa servono le finestre di dialogo di Swing.

Lezione 094

01. L'interfaccia MouseListener di Java:

- [ ] Permette di rilevare se il mouse è collegato alla porta USB del PC
- [ ] Permette di estendere le funzionalità del mouse di sistema
- [ ] Permette di rilevare i movimenti del mouse
- [ ] Permette di rilevare gli eventi del mouse

02. Qual è la differenza tra il metodo mousePressed() e mouseClicked() dell'interfaccia MouseListener?

- [ ] il primo rileva la pressione di un tasto della tastiera, il secondo un click del mouse
- [ ] rilevano entrambi un click del tasto sinistro del mouse
- [ ] il primo rileva la pressione del tasto sinistro del mouse, il secondo la pressione del tasto destro
- [ ] il primo rileva la pressione di un tasto del mouse, il secondo se il tasto è stato premuto e rilasciato

03. Per rilevare quale tasto del mouse è stato premuto bisogna utilizzare:

- [ ] Mouse.getButton()
- [ ] this.getButton()
- [ ] MouseEvent.getButton()
- [ ] System.mouse.getButton()

04. Quali valori sono associati ai pulsanti sinistro e destro del mouse da MouseEvent?

- [ ] MouseEvent.BUTTON0 al tasto sinistro, MouseEvent.BUTTON1 al tasto destro
- [ ] MouseEvent.BUTTON0 al tasto sinistro, MouseEvent.BUTTON2 al tasto destro
- [ ] MouseEvent.BUTTON1 al tasto sinistro, MouseEvent.BUTTON2 al tasto destro
- [ ] MouseEvent.BUTTON1 al tasto sinistro, MouseEvent.BUTTON3 al tasto destro

05. Per rilevare che il puntatore del mouse è uscito dall'area di un componente Swing:

- [ ] possiamo usare il metodo mouseOut(MouseEvent e)
- [ ] possiamo usare il metodo mouseExited(MouseEvent e)
- [ ] possiamo usare il metodo mouseOutOfArea(MouseEvent e)
- [ ] possiamo usare il metodo mousePressed(MouseEvent e)

06. Quali eventi del mouse è possibile rilevare con l'interfaccia MouseListener?

- [ ] Il click dei pulsanti, l'entrata e l'uscita dal componente
- [ ] Solo i movimenti all'interno di un componente
- [ ] Tutti i movimenti del mouse
- [ ] Solo il click dei pulsanti sul componente

07. Al fine di rilevare gli eventi generati dal mouse in un JFrame di Swing:

- [ ] Dobbiamo estendere l'interfaccia MouseListener
- [ ] Dobbiamo estendere l'interfaccia Mouse
- [ ] Dobbiamo estendere l'interfaccia MouseUsb
- [ ] Dobbiamo estendere l'interfaccia MouseMotion

08. Quale fra i seguenti eventi non è fra quelli rilevati dall'interfaccia MouseListener?

- [ ] mouseReleased()
- [ ] mouseMoved()
- [ ] mousePressed()
- [ ] mouseClicked()

09. Descrivere il funzionamento dell'interfaccia MouseListener per la cattura degli eventi del mouse.

10. Descrivere quali sono gli eventi del mouse che è possibile catturare tramite l'interfaccia MouseListener.

Lezione 095

01. Come otteniamo l'altezza della barra del titolo di un JFrame di Swing?

- [ ] con JFrame.bar.getHeight()
- [ ] con JFrame.getBar().getHeight()
- [ ] con getInsets().top
- [ ] con il metodo getBarHeight()

02. Per disegnare delle linee su un componente Swing:

- [ ] Possiamo usare il metodo drawLine() della classe Graphics2D
- [ ] Possiamo usare il metodo addLine() della classe Graphics2D
- [ ] Possiamo usare il metodo paint() della classe JComponent
- [ ] Possiamo usare il metodo showLine() della classe Graphics

03. Per disegnare linee utilizzando il colore rosso in Swing possiamo usare:

- [ ] Colore.ROSSO
- [ ] Color.RED
- [ ] Color("red")
- [ ] Color.red()

04. La classe da utilizzare per selezionare un colore di disegno in Swing è:

- [ ] java.lang.Color
- [ ] java.io.Color
- [ ] java.Color
- [ ] java.awt.Color

05. Per cancellare il contenuto di un componente Swing possiamo utilizzare:

- [ ] super.paintComponent()
- [ ] clearBackround()
- [ ] clear()
- [ ] clean()

06. Cosa si utilizza per forzare il ridisegno di un componente Swing?

- [ ] il metodo reset()
- [ ] il metodo paint()
- [ ] il metodo paintAll()
- [ ] il metodo repaint()

07. Per modificare il colore di disegno delle linee di un componente Swing:

- [ ] Possiamo utilizzare il metodo Graphics2D.setColor()
- [ ] Possiamo utilizzare il metodo Graphics2D.color()
- [ ] Possiamo utilizzare il metodo JComponent.color()
- [ ] Possiamo utilizzare il metodo JComponent.setColor()

08. Il metodo "Graphics2D.drawLine(A, B, C, D)":

- [ ] disegna un rettangolo
- [ ] disegna una linea dal punto A al punto B e una dal punto C al punto D
- [ ] disegna 3 linee: una da A a B, un'altra da B a C e la terza da C a D
- [ ] disegna una linea dal punto di coorindate (A,B) a quello di coordinate (C,D)