UNIVERSITÀ DEGLI STUDI DI CATANIA
Anno Accademico 2024 - 2025
Corso di Laurea in Informatica
Test di ingresso INFORMATICA
Prova del 28/09/24

COGNOME e NOME: (IN STAMPATELLO)	
FIRMA:	
N. MATRICOLA (qualora non si fosse ancora in possesso di matricola fornire codice fiscale):	

NON saranno soggette alla valutazione le prove MANCANTI del Cognome e Nome, nonché della Firma

Non sono consentiti formulari, appunti, libri e calcolatori; non è consentito comunicare con i colleghi; ogni mezzo di comunicazione elettronico deve essere tenuto spento. Durante la prova non è possibile uscire dall'aula prima di avere consegnato definitivamente il compito.

Per ciascuna delle seguenti dodici domande indicare l'unica risposta corretta.

1) Un algoritmo:

- ☐ È una sequenza ordinata e finita di passi che produce un risultato che dipende, tra la altre cose, dall'istante di esecuzione e dall'esecutore;
- ☐ È una sequenza ordinata e finita di passi che produce un risultato in un tempo finito o infinito;
- ☐ Produrrà un risultato parziale quando la risoluzione di uno o più problemi non è stata ben specificata nell'algoritmo;
- ☒ Produrrà generalmente un output differente in presenza di input differenti e/o di variabili di stato differenti;
- ☐ È una sequenza ordinata ed infinita di passi che produce un ben determinato risultato in un tempo finito.

---

2) La rappresentazione dei numeri decimali nei calcolatori (standard IEEE 754):

- ☐ un bit è riservato alla rappresentazione della virgola;
- ☐ un bit è riservato alla rappresentazione del segno, solo se il numero ha segno negativo;
- ☐ la differenza della rappresentazione in doppia precisione rispetto alla singola precisione è rappresentata unicamente dal maggior numero di bit assegnati al campo mantissa, che permette di rappresentare un maggior numero di cifre;
- ☐ include una rappresentazione in complemento a due del campo esponente;
- ☒ prevede che due combinazioni di bit del campo esponente siano riservate per usi speciali;

---

3) La cosiddetta macchina o architettura di Von Neumann:

- ☐ prevede che il programma sia memorizzato nella parte inziale della memoria centrale, mentre i dati siano memorizzati nei registri della CPU;
- ☐ prevede che il program counter memorizzi la prossima istruzione da eseguire;
- ☐ prevede che l'unità logico-artitmetica si occupi di coordinare il program counter e i registri della CPU;
- ☒ Nessuna delle risposte precedenti è corretta.

---

4) Nella rappresentazione dei numeri in virgola mobile che si basi sullo standard IEEE 754, il cosiddetto campo mantissa (M):

- ☐ si ottiene dalla moltiplicazione della rappresentazione in base due del numero stesso per 2^{-x} o 2^{+x}, con $x = 32$ (singola precisione) o $x = 64$ (doppia precisione);
- ☐ si ottiene dalla moltiplicazione della rappresentazione in complemento a due del numero stesso per 2^{-x} o 2^{+x}, con $x = 32$ (singola precisione) o $x = 64$ (doppia precisione);
- ☐ si ottiene dalla moltiplicazione della rappresentazione in complemento a due del numero stesso per 2^x, con x tale che $|A_{[2c]}| = 1.M \times 2^x$
- ☒ si ottiene dalla moltiplicazione della rappresentazione in base due del numero stesso per 2^x, con x tale che $|A_{[2]}| = 1.M \times 2^x$
- ☐ nessuna della risposte precedenti è corretta;

---

5) Il paradigma di programmazione modulare:

- ☐ si basa sul concetto di oggetti e moduli che si scambiano messaggi;
- ☐ si basa sul concetto di oggetti che vengono aggregati in moduli, i quali si scambiano messaggi;
- ☐ si basa sul concetto di moduli che "espongono" una interfaccia fatta di dati e oggetti;
- ☒ si basa sul concetto di moduli che "espongono" una interfaccia costituita da una o più procedure (o funzioni);
- ☐ nessuna delle risposte precedenti è corretta;

---

6) compilatore vs interprete:

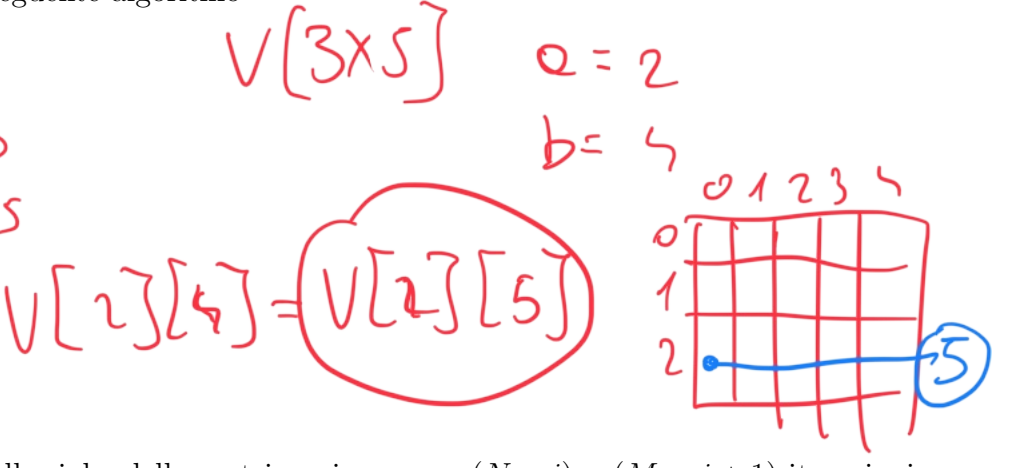
- ☐ minore efficienza del compilatore rispetto ad un interprete;
- ☐ in linea di principio, un programma che viene eseguito mediante interprete, risulta più veloce rispetto ad un programma compilato;
- ☒ in linea di principio, un programma che viene eseguito mediante interprete, risulta meno veloce rispetto ad un programma compilato;
- ☐ un programma compilato occupa generalmente più memoria rispetto ad un programma interpretato.

COMPILATORE È PIÙ VELOCE

---

7) Siano N, M, a e b numeri interi positivi. Si supponga inoltre che V sia una matrice di $N \times M$ elementi, e che $a < N$ e che $b < M$. Il seguente algoritmo

```
Leggi N, M, V, a, b
i ← a
While ( i < N ) Do
  j ← b
  While ( j < M ) Do
    V[i][j] ← V[i][j + 1]
    j ← j + 1
  EndWhile
  i ← i + 1
EndWhile
```



- ☐ procede ordinatamente sulle righe della matrice, si avranno $(N - i) \times (M - j + 1)$ iterazioni;
- ☒ contiene un errore sintattico;
- ☐ procede ordinatamente sulle colonne della matrice, si avranno $(N - a) \times (M - b)$ iterazioni;
- ☒ contiene un errore logico;
- ☐ procede ordinatamente sulle righe della matrice, si avranno $(N - a) \times (M - b + 1)$ iterazioni;

8) Sia N un intero positivo, C un intero positivo tale che $0 < C < N$, T una matrice quadrata di dimensioni $N \times N$. Completare opportunamente il seguente algoritmo per il calcolo della somma di $C+1$ elementi della diagonale secondaria di T:

Leggi N
Leggi C
Leggi T
??
$S \leftarrow 0$
While $(i >= 0)$ **Do**
    $S \leftarrow T[i][C-i] + S$
    $i \leftarrow i - 1$
Endwhile
Stampa S

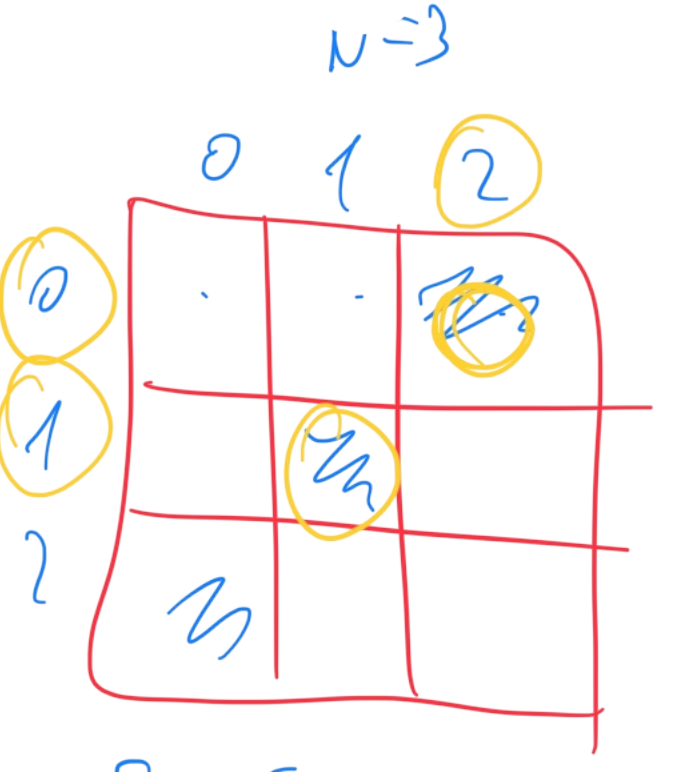
- ☐ $i \leftarrow C + 1$;
- ☐ $i \leftarrow 0$;
- ☐ $i \leftarrow C$;
- ☐ $i \leftarrow C - 1$;

---

9) Sia Y un numero intero positivo, sia V un vettore di Y elementi. Si consideri un algoritmo che calcoli e stampi i valori somma di ogni sottoarray di lunghezza α di V. Selezionare opportuni predicati da inserire al posto dei simboli $\bigtriangledown$ e $\bigcirc$:

Leggi Y;
Leggi V;
Leggi α;
$i \leftarrow 0$;
While ($\bigtriangledown$) **Do**
  $k \leftarrow 0$
  $S \leftarrow 0$;
  While($\bigcirc$) **Do**
    $S \leftarrow S + V[i+k]$;
    $k \leftarrow k + 1$
  EndWhile
  Stampa S
  $i \leftarrow i + 1$
EndWhile

- ☐ $\bigtriangledown$: $i < \alpha$
  $\bigcirc$: $k < Y - \alpha$
- ☐ $\bigtriangledown$: $i < Y - \alpha$
  $\bigcirc$: $k < \alpha$
- ☐ $\bigtriangledown$: $i < Y$
  $\bigcirc$: $k < Y - \alpha$
- ☐ $\bigtriangledown$: $i < \alpha$
  $\bigcirc$: $k < \alpha + Y$



V[i][N-1-i]

V[0][3-1-0]

V[1][3-1-1]

DIAGONALE SECONDARIA

8) Sia N un intero positivo, C un intero positivo tale che $0 < C < N$, T una matrice quadrata di dimensioni $N \times N$. Completare opportunamente il seguente algoritmo per il calcolo della somma di $C + 1$ elementi della diagonale secondaria di T:

Leggi N
Leggi C
Leggi T
??
$S \leftarrow 0$
While $(i >= 0)$ **Do**
    $S \leftarrow T[i][C - i] + S$
    $i \leftarrow i - 1$
Endwhile
Stampa S

- □ $i \leftarrow C + 1$;
- □ $i \leftarrow 0$;
- □ $i \leftarrow C$;
- □ $i \leftarrow C - 1$;

---

9) Sia Y un numero intero positivo, sia V un vettore di Y elementi. Si consideri un algoritmo che calcoli e stampi i valori somma di ogni sottoarray di lunghezza α di V. Selezionare opportuni predicati da inserire al posto dei simboli $\bigtriangledown$ e $\bigcirc$:

Leggi Y;
Leggi V;
Leggi α;
$i \leftarrow 0$;
While ($\bigtriangledown$) **Do**
  $k \leftarrow 0$
  $S \leftarrow 0$;
  While($\bigcirc$) **Do**
    $S \leftarrow S + V[i + k]$;
    $k \leftarrow k + 1$
  EndWhile
  Stampa S
  $i \leftarrow i + 1$
EndWhile

- □ $\bigtriangledown$: $i < \alpha$
  $\bigcirc$: $k < Y - \alpha$
- □ $\bigtriangledown$: $i < Y - \alpha$
  $\bigcirc$: $k < \alpha$
- □ $\bigtriangledown$: $i < Y$
  $\bigcirc$: $k < Y - \alpha$
- □ $\bigtriangledown$: $i < \alpha$
  $\bigcirc$: $k < \alpha + Y$

---

10 Al fine di convertire un numero decimale $X < 1$ in base 2, quindi ottenere una rappresentazione in base due $.a_1 a_2 \ldots a_n$

- ☐ Si divide il numero X per due e, successivamente anche il quoziente di tale divisione, e cosi via, finchè non si ottiene quoziente 0, conservando ad ogni passo il resto della divisione. I resti costituiranno, uno dopo l'altro, dal primo all'ultimo, le cifre $a_1, a_2, \ldots, a_n$.
- ☐ Si moltiplica il numero X per due e, successivamente anche la parte frazionaria del risultato, e cosi via, conservando ad ogni passo il riporto. I riporti costituiranno, uno dopo l'altro, dal primo all'ultimo, le cifre $a_1, a_2, \ldots, a_n$.
- ☐ Si divide il numero X per due e, successivamente anche il quoziente di tale divisione, e cosi via, finchè non si ottiene quoziente 0, conservando ad ogni passo il resto della divisione. I resti costituiranno, uno dopo l'altro, dal primo all'ultimo, le cifre $a_n, a_{n-1}, \ldots, a_1$.
- ☐ Si moltiplica il numero X per due e, successivamente anche la parte frazionaria del risultato, e cosi via, conservando ad ogni passo il riporto. I riporti costituiranno, uno dopo l'altro, dal primo all'ultimo, le cifre $a_n, a_{n-1}, \ldots, a_1$.

---

11 Sia N un numero intero positivo. Il seguente algoritmo in Notazione Lineare Strutturata (NLS)

Leggi N
$M \leftarrow -1$
While ($M < N$) **Do**
  $M \leftarrow M + 2$
  Stampa M
EndWhile

N = 5
M = -1
M = +1

produrrà il seguente output

- ☐ tutti i numeri interi dell'intervallo [-1, N]
- ☐ tutti i numeri interi pari dell'intervallo [1, N]
- ☒ tutti i numeri interi dispari dell'intervallo [1, N+1]
- ☐ tutti i numeri interi dispari dell'intervallo [1, N]

---

12 Il cosiddetto paradigma di programmazione imperativa:

- ☐ si basa sull'esecuzione di una serie di funzioni matematiche; → FUNZIONALE
- ☐ si basa sulla descrizione logica del problema; → LOGICA
- ☒ si basa su sequenze di istruzioni da impartire al calcolatore; → IMPERATIVA
- ☐ nessuna delle risposte precedenti è corretta;