

Fondamenti di Informatica  
Ingegneria Clinica  
Università di Roma “La Sapienza”

**Prova del 25/09/2009 – Prova A**

Le risposte corrette valgono 1 punto, quelle errate -0,25 punti, quelle non date 0 punti

Nota: Tra le proposte per ciascuna domanda una sola è esatta

1. Si consideri l'architettura di un Personal Computer.

**A Il clock è un circuito che emette una serie di impulsi con una frequenza costante.**

B Durante un ciclo di esecuzione il registro PC contiene l'indirizzo dell'istruzione da eseguire nel ciclo corrente.

D L'accesso ai registri della CPU è più lento rispetto all'accesso alla cache di primo livello.

E L'IR contiene il programma correntemente in esecuzione.

2. Si consideri ancora l'architettura di un Personal Computer.

**A La memoria centrale può contenere i dati del programma correntemente in esecuzione.**

B L'accesso alla cache di secondo livello è più lento rispetto all'accesso alla RAM.

C La memoria di massa è volatile.

D Il bus non può essere utilizzato per il trasferimento di dati e istruzioni tra memoria e CPU.

3. Linguaggi di programmazione.

A Un programma compilato e linkato è sicuramente privo di errori.

**B Il debugger è utile nella fase di sviluppo di un programma.**

C Le istruzioni in linguaggio ASSEMBLER sono espresse mediante una sequenza di bit.

D Un programma di alto livello non deve essere tradotto per poter essere eseguito.

4. Si consideri il seguente programma C:

```
#include <stdio.h>
main()
{ int x=2;
  float y=x--;
  char z='y';
  if (--y)
  if (--x)
  z='a';
  else
  z='b';
  else z=x;
  /* punto 1 */
  y=(x+=3, x--); /*istruzione 1*/
  /* punto 2 */
  { float A;
    A=(z>'a'? --y: y/x);
    /* punto 3 */
  }
  /*punto 4*/
}
```

A L'istruzione 1 è equivalente alla sequenza di istruzioni:  $x=x+3$ ;  $x--$ ;  $y=x$ ;

B Al punto 2 le variabili x e y hanno lo stesso valore.

C Al punto 3 la variabile A vale 8.

**D L'istruzione 1 effettua 3 assegnamenti.**

5. Si consideri il seguente programma C:

```
#include <stdio.h>
main()
{int A; float B; char C, ch;
A=6; B=4; C='C'; ch='D';
A=(A%(int)B ?++A: ch-C); /* istruzione 1 */
B=A++/B; /* istruzione 2 */
ch=C+=1; ch-=C; /* istruzione 3 */
}
```

- A Immediatamente dopo l'esecuzione dell'*istruzione 1*, la variabile A ha il valore corrispondente al carattere ASCII 'F'.
- B Il programma può generare errori durante l'esecuzione.
- C L'operatore di casting nell'*istruzione 1* è ininfluenza.
- D Immediatamente dopo l'esecuzione dell'*istruzione 2*, la variabile B ha il valore 1.75.

6. Si consideri ancora il precedente programma C:

```
#include <stdio.h>
main()
{int A; float B; char C, ch;
A=4; B=-2; C='C'; ch='D';
A=(A%(int)B ?++A: ch-C); /* istruzione 1 */
B=A++/B; /* istruzione 2 */
ch=C+=1; ch-=C; /* istruzione 3 */
}
```

- A L'operatore di casting nell'*istruzione 1* è ininfluenza.
- B Il programma può generare errori durante l'esecuzione.
- C Immediatamente dopo l'esecuzione dell'*istruzione 2*, la variabile B ha il valore -0.5.
- D Immediatamente dopo l'esecuzione dell'*istruzione 1*, la variabile A ha il valore corrispondente al carattere ASCII 'F'.

7. Si considerino i vettori nel linguaggio C:

- A L'indice di un elemento può essere una variabile di tipo float.
- B L'indice di un elemento di un vettore deve essere minore della sua dimensione logica.
- C Non è possibile stampare il contenuto di un vettore di 3 interi con una sola printf.
- D Gli elementi di un vettore sono tutti dello stesso tipo.

8. Si considerino ancora i vettori nel linguaggio C:

- A Non è possibile stampare il contenuto di un vettore di 5 interi con una sola printf.
- B Non è possibile ordinare un vettore di interi in ordine decrescente.
- C Nella definizione di un vettore, la dimensione può essere espressa mediante una espressione matematica.
- D L'indice di un elemento di un vettore deve essere maggiore della sua dimensione fisica.

9. Si considerino le caratteristiche del linguaggio C:

A E' possibile che in un programma vengano definite due variabili diverse aventi lo stesso identificatore, ed entrambe allocate nel data segment.

B Lo stack pu' contenere variabili allocate dinamicamente (tramite la `malloc` ).

**C Gli identificatori `stdin`, `stdout` e `stderr` denotano puntatori a FILE.**

D All'interno di una funzione e' possibile definire blocchi di istruzioni contenenti nuove variabili, il cui campo d'azione diventa l'intero programma.

**10. Si consideri il seguente programma C:**

```
#include <stdio.h>
#define N 9
int f(int *a, int b) {
    if( *a < *(a+b) ) /* punto 1 */
        return b;
    else
        return *a=f( a, b+1 )+b;
}
main() {
    int i, V[N]={0,1,2,3,4,5,6,7,8};
    for ( i=N-1; i>=0; i-=2 )
        V[i]=(i+5)/2;
    /*punto 2 */
    printf("%d\n\n", f(V,0));
    /* punto 3 */
    for (i=0; i<N; i++) /* punto 4*/
        printf("%d\n", V[i]);
}
```

A Il ciclo al punto 4 stampa i seguenti valori (in linee separate): 2,1,3,3,4,9,5,7,6.

B Al punto 3 la variabile `i` ha valore 0.

C L'istruzione prima del punto 1 (`if( *a<*(a+b) )`) viene eseguita 7 volte.

**D Al punto 1 la variabile `V` non e' visibile.**