

<p>1. [3.5]</p>	<p>Si consideri l'architettura di un Personal Computer.</p> <p>A [V] La memoria centrale puo' contenere i dati del programma correntemente in esecuzione.</p> <p>B [V] L'accesso alla cache di secondo livello e' piu' veloce rispetto all'accesso alla RAM.</p> <p>C [F] La memoria di massa e' volatile.</p> <p>D [V] La Control Unit (CU) esercita il controllo sui trasferimenti tra CPU e memoria.</p> <p>E [V] Il bus puo' essere utilizzato per il trasferimento di dati e istruzioni tra memoria e CPU.</p>	<p>2. [3.5]</p>	<p>Linguaggi di programmazione.</p> <p>A [V] Lo sviluppo di programmi in linguaggio assembler necessita dell'uso di un traduttore.</p> <p>B [F] Un compilatore per il linguaggio C e' un programma che e' indipendente dall'architettura HW/SW del computer utilizzato.</p> <p>C [V] L'esecuzione di un programma compilato e' tipicamente più veloce dell'esecuzione dello stesso programma interpretato.</p> <p>D [F] Un programma scritto in linguaggio assembler non necessita di traduzione.</p> <p>E [F] Il debugger serve per trovare più facilmente errori sintattici nei programmi.</p>
<p>3. [4]</p>	<p>Si consideri il seguente programma C:</p> <pre>#include <stdio.h> main() { int A=2; float B=A--; char C=A; if (--B) if(--A) C='B'; else C='A'; else C=A; /* punto 1 */ B=(A+=3, A--); /*istruzione 1*/ /* punto 2 */ { float num; num=(C>'B'? B--: B/A); /* punto 3*/ } /*punto 4*/ }</pre> <p>A [V] Al punto 2 la variabile A ha valore 2.</p> <p>B [V] Al punto 2 le variabili A e B hanno valore diverso.</p> <p>C [V] Al punto 3 la variabile num ha valore 1.5.</p> <p>D [F] Al punto 4 la variabile num e' visibile.</p> <p>E [F] L'istruzione 1 effettua 2 assegnamenti.</p>	<p>4. [4]</p>	<p>Si consideri il seguente programma C:</p> <pre>#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 */ }</pre> <p>A [F] L'operatore di casting nell'istruzione 1 e' ininfluenza.</p> <p>B [F] Il programma puo' generare errori durante l'esecuzione.</p> <p>C [V] Immediatamente dopo l'esecuzione dell'istruzione 2, la variabile B ha il valore -0.5.</p> <p>D [V] Immediatamente dopo l'esecuzione dell'istruzione 3, la variabile ch ha valore '0'.</p> <p>E [F] Immediatamente dopo l'esecuzione dell'istruzione 1, la variabile A ha il valore corrispondente al carattere ASCII 'A'.</p>

<p>5. [4]</p>	<p>Si consideri il seguente programma C:</p> <pre>#include <stdio.h> main() {int x[4]={0, 3, 4, 8}; int j, A, B=0, C; for(j=3; j>=0; j--) /*istruzione 1*/ switch(x[j]%3-1) /*istruzione 2*/ {case 0: x[j]--; case 1: A=x[j]; break; case 2: C=0; break; default: B--A; } C=x[0]+x[2] - B; /* istruzione 3*/ }</pre> <p>A [F] Immediatamente dopo il ciclo for, la variabile C vale 0. B [V] Il blocco di istruzioni corrispondente al case 1: viene eseguito 2 volte. C [F] Immediatamente dopo l'istruzione 3, la variabile C ha il valore 3. D [F] Al termine dell'esecuzione, il valore di due elementi di V risultano modificati rispetto ai loro valori iniziali. E [F] Al termine dell'esecuzione, il valore di V[3] e' uguale a 7.</p>	<p>Si consideri il seguente programma C:</p> <pre>#include <stdio.h> #define N 8 main() {typedef float V1[N]; typedef char V2[N]; V1 x={0,0.5,1,1.5,2.0,2.5,3.0,3.5}; V2 y={'C','i','a','o',0,1,2,3}; int i; x[N-1]=0; for (i=1; i<N; i+=i) x[i]=y[N-i]/i; /* punto 1 */ if (x[--i]?1:x[0]) printf("Ciao!\n"); else {i=x[0]; for (;i<=4;) printf("%c", y[i++]); } /* punto 2*/ }</pre> <p>A [F] Il programma contiene un errore di sintassi. B [V] Al punto 2: x[i-1] ha valore 0.0. C [F] Al punto 1: il valore di x[4] e' 1.5. D [F] Al punto 1: almeno un elemento di y e' stato modificato. E [V] Al punto 2: la variabile i ha valore N-3.</p>
<p>7. [3.5]</p>	<p>Si considerino i vettori nel linguaggio C:</p> <p>A [V] E' possibile stampare il contenuto di un vettore di 5 interi con una sola printf. B [V] E' possibile ordinare un vettore di interi in ordine crescente. C [V] Nella definizione di un vettore, la dimensione puo' essere espressa mediante una espressione matematica. D [V] L'indice di un elemento di un vettore deve essere minore della sua dimensione fisica. E [V] L'indice di un elemento puo' essere una variabile di tipo char.</p>	<p>Si considerino i tipi di dato scalari primitivi nel linguaggio C:</p> <p>A [F] La sottrazione tra un float e un char produce un errore in compilazione. B [F] Nel linguaggio C il vettore e' un tipo scalare primitivo. C [F] L'applicazione della regola di conversione implicita produce una espressione in cui tutti gli operandi assumono uno stesso tipo. D [F] La conversione esplicita puo' trasformare un dato scalare in uno strutturato. E [V] Il criterio di equivalenza strutturale puo' consentire l'assegnamento del valore di una variabile di tipo non primitivo a una variabile di tipo scalare primitivo.</p>
<p>9. [3.5]</p>	<p>Si considerino le istruzioni del linguaggio C:</p> <p>A [V] Ogni istruzione switch e' sempre esprimibile mediante uno o piu' if (eventualmente annidati). B [V] Nell'istruzione for l'espressione di modifica viene sempre eseguita alla fine di ogni iterazione. C [V] Ogni istruzione for e' sempre esprimibile mediante while. D [F] Ogni istruzione if e' esprimibile mediante un unico ciclo while. E [F] Non e' possibile usare un if all'interno di uno switch.</p>	