



**Note allo svolgimento.** Ti sono stati consegnati:

- a — il *testo del compito*;
- b — una *scheda anagrafica* con i tuoi dati: sul retro della scheda vi sono due adesivi che riportano un codice a barre;
- c — una *scheda delle risposte*;
- d — un *foglio di istruzioni* per la compilazione della scheda delle risposte.

Per un corretto svolgimento della prova, è necessario seguire in sequenza le seguenti istruzioni:

1. applicare *uno* dei due adesivi contenenti il codice a barre nell'apposito riquadro sulla scheda anagrafica;
2. scrivere il tipo di compito assegnato (**H**) sulla scheda anagrafica, di fianco al proprio nome;
3. compilare la scheda delle risposte, indicando sulla stessa con una croce × le risposte ritenute corrette, (vedi foglio di istruzioni). A questo proposito, è importante sapere che:
  - ogni errore determinerà un **punteggio negativo**;
  - se a un quesito non si fornisce alcuna risposta, si otterrà per quel quesito il **punteggio 0**;
4. al termine della prova, applicare il secondo codice a barre adesivo nell'apposito riquadro sulla scheda delle risposte.

Consegnare i due fogli su cui è stato applicato il codice a barre (scheda anagrafica e scheda delle risposte).

È severamente proibito consultare libri, appunti, manuali, o strumenti elettronici (computer, cellulari, palmari ecc.).

La prova dura 1 ora.

<p><b>1.</b> [3.5]</p>	<p>Si consideri l'architettura di un Personal Computer.</p> <p>A Il bus può essere utilizzato per il trasferimento di dati e istruzioni tra memoria e CPU.</p> <p>B Nella fase di fetch viene eseguita l'istruzione indirizzata dal Program Counter.</p> <p>C Il registro Flag (o PSW) contiene informazioni riguardanti l'ultima operazione eseguita dalla ALU.</p> <p>D Ad ogni ciclo il registro IR contiene l'istruzione da eseguire nello stesso ciclo.</p> <p>E La memoria centrale può contenere i dati del programma correntemente in esecuzione.</p>	<p><b>2.</b> [3.5]</p>	<p>Linguaggi di programmazione.</p> <p>A Il debugger serve per trovare più facilmente errori sintattici nei programmi.</p> <p>B Un programma che viene compilato correttamente può contenere errori.</p> <p>C Un programma espresso in linguaggio macchina e' una sequenza di bit.</p> <p>D Un compilatore traduce un programma scritto con un linguaggio di alto livello in uno scritto con un linguaggio a basso livello, ma non viceversa.</p> <p>E Lo sviluppo di programmi in linguaggio assembler necessita dell'uso di un traduttore.</p>
<p><b>3.</b> [4]</p>	<p>Si consideri il seguente programma C:</p> <pre>#include &lt;stdio.h&gt;  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&gt;'B'? B--: B/A);     /* punto 3*/   }   /*punto 4*/ }</pre> <p>A L'istruzione 1 effettua 2 assegnamenti.</p> <p>B Al punto 1 la variabile A ha valore 0.</p> <p>C Al punto 1 la variabile C assume il valore 'B'.</p> <p>D L'istruzione 1 e' equivalente alla sequenza di istruzioni: A=A+3; B=A; --A;</p> <p>E Al punto 2 la variabile A ha valore 2.</p>	<p><b>4.</b> [4]</p>	<p>Si consideri il seguente programma C:</p> <pre>#include &lt;stdio.h&gt;  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 Immediatamente dopo l'esecuzione dell'istruzione 1, la variabile A ha il valore corrispondente al carattere ASCII 'A'.</p> <p>B Immediatamente dopo l'esecuzione dell'istruzione 3, le variabili C e ch hanno lo stesso valore.</p> <p>C Immediatamente dopo l'esecuzione dell'istruzione 2, la variabile A ha il valore 2.</p> <p>D Immediatamente dopo l'esecuzione dell'istruzione 3, la variabile C ha valore 'C'.</p> <p>E L'operatore di casting nell'istruzione 1 e' ininfluente.</p>

<p><b>5.</b> [4]</p>	<p>Si consideri il seguente programma C:</p> <pre>#include &lt;stdio.h&gt;  main() {int x[4]={0, 3, 4, 8};  int j, A, B=0, C;  for(j=3; j&gt;=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 Al termine dell'esecuzione, il valore di V[3] e' uguale a 7.  B Immediatamente dopo l'istruzione 3, la variabile A ha il valore 1.  C Immediatamente dopo il ciclo for, la variabile B ha il valore 1.  D L'esecuzione del ciclo for provoca 4 iterazioni.  E Immediatamente dopo il ciclo for, la variabile C vale 0.</p>	<p>Si consideri il seguente programma C:</p> <pre>#include &lt;stdio.h&gt; #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&lt;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&lt;=4;) printf("%c", y[i++]);   }  /* punto 2*/ }</pre> <p>A Al punto 2: la variabile i ha valore N-3.  B Il programma stampa "Ciao!".  C Al punto 1: il valore di x[1] e' 3.  D Nel programma non vengono usati tipi di dato non primitivi.  E Il programma contiene un errore di sintassi.</p>
<p><b>7.</b> [3.5]</p>	<p>Si considerino i vettori nel linguaggio C:</p> <p>A L'indice di un elemento non puo' essere una variabile di tipo float.  B E' possibile copiare gli elementi di un vettore in un altro vettore.  C Gli elementi di uno stesso vettore possono essere di tipo diverso.  D E' possibile stampare il contenuto di un vettore di 5 interi con una sola printf.  E E' possibile ordinare un vettore di interi in ordine crescente.</p>	<p>Si considerino i tipi di dato scalari primitivi nel linguaggio C:</p> <p>A Non e' possibile eseguire una somma tra una variabile double e una variabile char  B L'<i>overloading</i> degli operatori permette di denotare operatori diversi con lo stesso simbolo.  C E' possibile applicare l'operatore % (modulo, o resto) a una variabile float e una variabile int.  D La sottrazione tra un float e un char produce un errore in compilazione.  E Nel linguaggio C il vettore e' un tipo scalare primitivo.</p>
<p><b>9.</b> [3.5]</p>	<p>Si considerino le istruzioni del linguaggio C:</p> <p>A Nell'istruzione <code>do..while</code> il blocco di istruzioni puo' non essere eseguito neanche una volta.  B Nell'istruzione <code>do..while</code> se la condizione non e' verificata, si esce dal ciclo.  C Nell'istruzione <code>switch</code> puo' essere eseguito al più un solo blocco di istruzioni.  D Ogni istruzione <code>switch</code> e' sempre esprimibile mediante uno o piu' <code>if</code> (eventualmente annidati).  E Nell'istruzione <code>for</code> l'espressione di modifica viene sempre eseguita alla fine di ogni iterazione.</p>	