Paolo Torroni, *Fondamenti di Informatica L-A* (*Elettronica e Telecomunicazioni*) **17-18/10/2006**

1. Quali sono le tre strutture di controllo nella programmazione strutturata?
2. L'alternativa è una struttura di controllo che rientra nella programmazione strutturata?
3. Il salto incondizionato rientra nella programmazione strutturata?
4. Perché è importante programmare in modo strutturato?
5. Cosa significa programmare in modo *top-down*?
6. Cosa significa che un linguaggio di programmazione è "completo"?
7. Quali sono le istruzioni che rendono il C un linguaggio completo?
8. È possibile dichiarare variabili all'interno di una istruzione composta?
9. Qual è l'output del seguente programma?

```
int x=0;
{    int x=1,y=1;
     { int x=2; printf( " %d %d", x, y ); }
     printf( " %d %d", x, y );
}
printf( "%d", x );
```

10. Si considerino le seguenti istruzioni:

```
int x=2; char A='a', B='f';
if( x==A ) x++;                      // istruzione 1
if( A && B ) A++ else A--;           // istruzione 2
if( A+x == B-x ) B+=x else B-=x;     // istruzione 3
```

    10.1. Quanto vale la condizione dell'**if** della **istruzione 1**?
    10.2. Quanto vale la condizione dell'**if** della **istruzione 2**?
    10.3. Quanto valgono **A** e **B** dopo che è stata eseguita la **istruzione 3**?

11. Si consideri la seguente istruzione (dove **x** è una variabile di tipo **int**):

```
if( 0 || 1 ) {
     if( !( 2 && 3 ) ) x=4; else x=5;
} else x=6;
```

    Quanto vale **x** dopo che l'istruzione è stata eseguita?

12. Si consideri la seguente istruzione composta (**A** è una variabile di tipo **char**):

```
if( scanf( "%c", &A ) ) {
     int x=1;
     printf( "%c", A+x ); }
```

    12.1. Si disegni il relativo diagramma di flusso.
    12.2. Quanto vale la condizione dell'**if**?
    12.3. Quanto valgono **A** e **x** dopo l'esecuzione di questa istruzione?

13. Qual è l'output della seguente istruzione (dove **x** è una variabile di tipo **int**)?

```
if( scanf( " %d", &x ) )
     switch( x ) {
          case 'a': printf( "%c", x );
          case  0 : printf( "%c", x+'A' );
          default : printf( " %d", x );
     }
```

14. Si scriva diagramma di flusso e codice C di un algoritmo che, dato un intero (**x**) in ingresso, stampi tutti i numeri pari da 0 a **x** se **x** è pari e positivo, tutti i multipli di 5 da 0 a **x** se **x** è dispari e positivo, il valore **0** se **x** è minore o uguale a zero.

15. Si considerino le seguenti istruzioni (dove **i** è una variabile di tipo **int** e **A** è una variabile di tipo **char**):

```
i=5;
while( scanf( "%c", &A )&& i )
     { printf( "%c", A-- ); i--; }
```

    15.1. Si disegni il relativo diagramma di flusso.
    15.2. Quando termina questo ciclo?
    15.3. Quanto vale **i** all'uscita del ciclo?
    15.4. È possibile scrivere un ciclo **for** equivalente al ciclo **while**?

16. Si considerino le seguenti istruzioni (dove **i** è una variabile di tipo **int** e **A** è una variabile di tipo **char**):

```
i=5;
while( scanf( "%c", &A )&& i ) {
     while( i>0 ) {
          printf( "%c", A-- );
          i--;
     }
}
```

    16.1. Si disegni il relativo diagramma di flusso.
    16.2. Quando termina questo ciclo?
    16.3. Quanto vale **i** all'uscita del ciclo?
    16.4. È possibile implementare in C un algoritmo equivalente senza utilizzare il costrutto **while**?

17. Si consideri la seguente istruzione (dove **i** è una variabile di tipo **int** e **A** è una variabile di tipo **char**):

```
i=5;
do   {
     while( i>0 ) {
          printf( "%c", 'a' + i-- );
     }
} while( scanf( "%c", &A )&& i=5 );
```

    17.1. Si disegni il relativo diagramma di flusso.
    17.2. Quando termina questo ciclo?
    17.3. Si implementi in C un algoritmo equivalente senza utilizzare i costrutti **while** e **do…while**.

18. Si consideri il seguente algoritmo:

```
int i,j=5;
for(i=1;i<5;i++) {
     int j=3;
     switch( j/i ) {
          case 3: break;
          case 1: while( j ) printf( " %d", j-- );
          default: printf( " %d", j/2 );
     }
}
printf( " %d", i+j );
```

    18.1. Quali sono i valori prodotti in output?
    18.2. Si implementi in C un algoritmo equivalente senza utilizzare i costrutti **for** e **switch**.