



SELEZIONI SCOLASTICHE 2018

COME FUNZIONA LO PSEUDOCODICE

Lo Staff

4 novembre 2018

Da questa edizione delle fasi scolastiche delle Olimpiadi Italiane di Informatica nella sezione “Problemi di Programmazione” non vi saranno più esercizi in C/C++ o Pascal, ma sarà introdotto il formalismo dello **pseudocodice**. Lo pseudocodice è un linguaggio di programmazione per modellare e dettagliare algoritmi prescindendo da qualsiasi implementazione, ma al tempo stesso sottolineando con sufficiente chiarezza ogni aspetto che preme comprendere compiutamente in quel particolare contesto.

La sintassi ammessa è quella che segue:

Pseudocodice	Significato
<code>a ←</code>	è un assegnamento
<code>leggi a;</code> <code>scrivi a;</code>	sono le istruzioni per leggere da tastiera e scrivere a video
<code>scrivi(_spazio_);</code> <code>scrivi(_a_capo_);</code>	scrivono su video rispettivamente uno spazio e un a capo
<code>scrivi("s");</code> <code>scrivi("s", var);</code>	scrive la stringa <code>s</code> a video scrive <code>s</code> e poi il contenuto della variabile <code>var</code>
<code>;</code>	il punto e virgola deve essere scritto alla fine di ogni istruzione, salvo dopo le parole chiave in grassetto
<code>array[7] ← 0;</code> <code>mat[5][28] ← 1</code>	esempi di inizializzazione a un valore degli elementi di array e matrici, indicizzati da 1 a n
<code>+</code> <code>-</code> <code>*</code> <code>/</code> <code>div</code> <code>mod</code>	sono i simboli che indicano rispettivamente le operazioni aritmetiche di somma, differenza, prodotto, divisione, divisione intera e resto
<code>=</code> <code><</code> <code>></code> <code><=</code> <code>>=</code> <code>!=</code>	sono i simboli che indicano i sei tipi di confronto: uguale, minore, maggiore, minore o uguale, maggiore o uguale, diverso
<code>&</code> <code> </code> <code>!</code>	sono i simboli che indicano le operazioni logiche AND, OR, NOT
<code>(x, y)</code>	esiste la struttura dati “coppia di valori”

Pseudocodice	Significato
Start ... End	delimitano un blocco di istruzioni
se condizione allora ... fine se	è un costrutto condizionale
finchè condizione ripeti ... fine finchè	è un ciclo a condizione iniziale (pre-posta)
ripeti ... finchè condizione	è un ciclo a condizione finale (post-posta)
funzione nome_funzione(p1:tipo, ..., pn:tipo) :tipo_restituito ... ritorna valore_restituito; fine funzione	sintassi di una funzione
procedura nome_procedura(p1:tipo, ..., pn:tipo) ... ritorna; fine procedura	sintassi di una procedura

Di seguito alcuni esempi tratti dalle fasi scolastiche 2017 con la versione equivalente in pseudocodice.

Esercizio n°6

Si consideri la seguente funzione:

C/C++	Pascal
<pre>int fun(int p) { printf("%d -> ", p); if (p%2 == 0) printf("condizione 1\n"); if (p == 7) printf("condizione 2\n"); else if ((p-5)%2 == 0) printf("condizione 3\n"); return p; }</pre>	<pre>function fun(p:integer):integer; begin write(p); write(' -> '); if (p mod 2 = 0) then writeln('condizione 1'); if (p = 7) then writeln('condizione 2') else if ((p-5) mod 2 = 0) then writeln('condizione 3'); fun:=p; end;</pre>

La versione in pseudocodice segue:

Pseudocodice esercizio 6

```
1: funzione fun(p:intero):intero
2:   finchè i=0 ripeti
3:     scrivi(p, "->");
4:   fine finchè
5:   se p mod 2 = 0 allora
6:     scrivi("condizione 1");
7:     scrivi(_a_capo_);
8:   fine se
9:   se p = 7 allora
10:    scrivi("condizione 2");
11:    scrivi(_a_capo_);
12:  altrimenti
13:    se (p - 5) mod 2 = 0 allora
14:      scrivi("condizione 3");
15:      scrivi(_a_capo_);
16:    fine se
17:  fine se
18:  ritorna p;
19: fine funzione
```

Quale delle seguenti affermazioni è errata?

- (a) La funzione, se p è pari, scrive a video il valore di p seguito dalla stringa `-> condizione 1` e ritorna p
- (b) La funzione, se p non è dispari, scrive a video il valore di p seguito dalla stringa `-> condizione 2` e ritorna p
- (c) La funzione, se p è 7, scrive a video il valore di p seguito dalla stringa `-> condizione 2` e ritorna p
- (d) La funzione, se p è dispari, scrive a video p seguito dalla stringa `-> condizione 2` o `-> condizione 3` e ritorna p

Esercizio n°7

È dato il seguente programma:

C/C++	Pascal
<pre>#include <stdio.h> #include <math.h> int main() { int x,y,a,p; float l, d; x=20; y=10; a=x*y; p=2*x+2*y; l=p/4; d=sqrt(2)*l; printf("%f cm", d); if(d*2 -720 == 0) d=2; else d=1; return 0; }</pre>	<pre>program E7(input,output); var x,y,a,p:integer; l,d:real; begin x:=20; y:=10; a:=x*y; p:=2*x+2*y; l:=p/4; d:=sqrt(2)*l; writeln(d:7:6, ' cm'); if(d * 2 -720 = 0) then d:=2 else d:=1; end.</pre>

La versione in pseudocodice segue:

Pseudocodice esercizio 7

```
1: Start
2:   x ← 20;
3:   y ← 10;
4:   a ← x * y;
5:   p ← 2 * x + 2 * y;
6:   l ← p/4;
7:   d ← sqrt(2) * l;
8:   stampa(d, "cm");
9:   se D * 2 - 720 = 0 allora
10:      d ← 2;
11:   altrimenti
12:      d ← 1;
13:   fine se
14: End
```

Cosa viene visualizzato a video dall'esecuzione del programma qui sopra?

- (a) 2.000000 cm
- (b) 3.000000 cm
- (c) 21.213203 cm
- (d) 36.243204 cm

Esercizio n°8

Si consideri la seguente funzione:

C/C++	Pascal
<pre>int myster(int c, int d) { if(c==d) return c; if(c>d) return myster(c-d, d); return myster(c, d-c); } int mcm(int a, int b) { return myster(b,a); }</pre>	<pre>function myster(c:longint; d:longint): longint; begin if c=d then myster:=c else if c>d then myster:=myster(c-d, d) else myster:=myster(c, d-c); end; function mcm(a:longint; b:longint): longint; begin mcm:= myster(b,a); end;</pre>

La versione in pseudocodice segue:

Pseudocodice esercizio 8

```
1: funzione myster(c:intero, d:intero):intero  
2:     se c = d allora  
3:         ritorna c;  
4:     fine se  
5:     se c > d allora  
6:         ritorna myster(c-d, d);  
7:     fine se  
8:     ritorna myster(c, d-c);  
9: fine funzione  
  
10: funzione mcm(a:intero, b:intero):intero  
11:     ritorna myster(b, a);  
12: fine funzione
```

Quale delle seguenti modifiche fa sì che la funzione mcm ritorni il minimo comune multiplo tra a e b?

- (a) sostituire `myster(b,a);` con `myster(a,b);`
- (b) sostituire `myster(b,a);` con `(a*b)/myster(b,a);`
- (c) sostituire `myster(b,a);` con `myster(a-b,b);`
- (d) sostituire `myster(b,a);` con `myster(a,b-a);`