

### 1.3.1 Soluzioni

#### Soluzione dell'esercizio 1.1

1. Dopo aver letto i due numeri ed averli memorizzati in due "foglietti" distinti,  $a$  e  $b$ , calcoliamo il prodotto in modo cumulativo utilizzando un ciclo.

```

leggi(f)      //leggi il primo numero
leggi(g)      //leggi il secondo numero

risultato ← 0

Finché  $g > 0$  esegui
    risultato ← risultato + f      //aggiungi f per g volte
     $g \leftarrow g - 1$ 
chiudi Finché

stampa(risultato)

```

2. Una caratteristica fondamentale degli algoritmi è l'efficienza, ci aspettiamo che le soluzioni devono essere fornite nel modo più veloce possibile.

```

leggi(f)      //leggi il primo numero
leggi(g)      //leggi il secondo numero
risultato ← 0

Se  $g < f$  allora
    Finché  $g > 0$  esegui
        risultato ← risultato + f      //aggiungi f per g volte
         $g \leftarrow g - 1$ 
    chiudi Finché
Altrimenti
    Finché  $f > 0$  esegui
        risultato ← risultato + g      //aggiungi g per f volte
         $f \leftarrow f - 1$ 
    chiudi Finché
chiudi Se

stampa(risultato)

```

Si noti che è possibile produrre un algoritmo analogo, comunque ottimizzato, ma con meno istruzioni.

```

leggi(f)      //leggi il primo numero
leggi(g)      //leggi il secondo numero

risultato ← 0

Se  $g < f$  allora
     $min \leftarrow g$ 
Altrimenti
     $min \leftarrow f$ 
chiudi Se

Finché  $min > 0$  esegui
     $risultato \leftarrow risultato + f$       //aggiungi  $f$  per  $min$  volte
     $min \leftarrow min - 1$ 
chiudi Finché

stampa(risultato)

```

L'efficienza di un algoritmo si misura anche sulla base della memoria consumata; si preferiscono algoritmi che impiegano meno memoria possibile. Abbiamo introdotto 4 variabili, non sono troppe?

```

leggi(f)      //leggi il primo numero
leggi(g)      //leggi il secondo numero

tmp ← 0

Se  $g < f$  allora      //assicuriamoci che  $f < g$ 
     $tmp \leftarrow f$       //metto  $f$  su un foglietto temporaneo
     $f \leftarrow g$       //g contiene il numero minore
     $g \leftarrow tmp$ 
Altrimenti
     $tmp \leftarrow g$       //l'altro lo scrivo in  $g$ 
chiudi Se

Finché  $f > 1$  esegui      //ripeti la somma  $f - 1$  volte
     $tmp \leftarrow tmp + g$       //aggiungi  $f$  per  $min$  volte
     $f \leftarrow f - 1$ 
chiudi Finché

stampa(risultato)

```

**Soluzione dell'esercizio 1.2**

Si legge il numero e lo si scrive in  $f$ , si esegue un ciclo moltiplicando il fattoriale (inizialmente pari ad  $f$ ) ad  $f$  per un numero di volte pari ad  $f - 1$ .

```
leggi(f)
fattoriale ← f

Finché  $f > 1$  esegui
     $f \leftarrow f - 1$ 
     $fattoriale \leftarrow fattoriale * f$ 
chiudi Finché

stampa(fattoriale)
```

**Soluzione dell'esercizio 1.3**

Si legge il radicando e lo si scrive in  $r$ . Partendo da  $r/2$  si cerca quel numero  $x$  tale per cui  $x^2 = r$  (condizione di ciclo).

```
leggi(r)
 $x = r/2$ 

Finché  $x * x > r$  esegui
     $x \leftarrow x - 1$ 
chiudi Finché
```

**Soluzione dell'esercizio 1.4**

```
leggi(a)      //lettura estremo inferiore
leggi(b)      //lettura estremo superiore

calcola(f, a, A)
calcola(f, b, B)

Se  $A * B < 0$  allora
    stampa("esiste almeno uno zero")
Altrimenti
    stampa("potrebbe non esistere alcuno zero")
chiudi Se
```

## Soluzione dell'esercizio 1.5

```

leggi(a)      //lettura estremo inferiore
leggi(b)      //lettura estremo superiore

calcola(f, a, A)
calcola(f, b, B)

Se  $A * B > 0$  allora
    stampa("potrebbe non esistere alcuno zero")
Altrimenti
     $K \leftarrow 1$ 

    Finché  $K \neq 0$  esegui
         $c \leftarrow (b - a)/2$ 
        calcola(f, c, K)

        Se  $A * K < 0$  allora
             $b \leftarrow c$ 
        chiudi Se

        Se  $B * K < 0$  allora
             $a \leftarrow c$ 
        chiudi Se
    chiudi Finché
chiudi Se

    stampa(c)

```

**Attenzione:** Il programma potrebbe non terminare nel caso in cui  $c \leftarrow (b - a)/2$  non sia esattamente il valore dello zero. Questo perché, essendo in  $\mathbb{R}$ , ci si può avvicinare sempre di più allo zero senza mai arrivarci esattamente.

Inoltre, si osservi che il programma non gestisce il caso in cui lo zero  $c$  sia  $c = a$  o  $c = b$ .

## Soluzione dell'esercizio 1.6

```
max ← 0
i ← 0
p ← i

Finché  $i \leq 10$  esegui
    leggi( $N[i]$ )
     $i \leftarrow i + 1$ 
chiudi Finché

i ← 0

Finché  $i < 10$  esegui
    Se  $N[i] > max$  allora
         $max \leftarrow N[i]$ 
         $p \leftarrow i$ 
    chiudi Se
     $i \leftarrow i + 1$ 
chiudi Finché

stampa("il valore massimo è:")
stampa(max)

stampa("ed è in posizione:")
stampa(p)
```

### Soluzione dell'esercizio 1.7

Prima di tutto notiamo che il primo ciclo, quello di acquisizione, memorizza i numeri nel blocco di foglietti. Tuttavia, il secondo ciclo, non fa altro che esaminarli (nello stesso ordine). Quindi, è possibile eliminare del tutto la necessità di memorizzare i numeri, e svolgere le istruzioni per il controllo "maggiore di" subito dopo l'acquisizione.

Inoltre, eliminando l'assunzione di numeri positivi, non abbiamo un estremo inferiore (zero). Perciò è necessario inizializzare il massimo al primo valore incontrato (che potrebbe essere inferiore a zero).

```
i ← 0
p ← i
n ← 0      //foglietto per il numero letto

Finché i < 10 esegui
    leggi(n)

    Se i == 0 ∧ n > max allora      //primo numero o maggiore
        max ← n      //salvo il valore massimo
        p ← i      //salvo la posizione
    chiudi Se

    i ← i + 1      //incremento il contatore
chiudi Finché

stampa("il valore massimo è:")
stampa(max)

stampa("ed è in posizione:")
stampa(p)
```

1. Il calcolo della media richiede che, durante l'acquisizione, si calcoli anche la somma incrementale di tutti i numeri letti. Al termine del ciclo, tale somma verrà divisa per il contatore.

```
i ← 0
p ← i
media ← 0
n ← 0      //foglietto per il numero letto

Finché i < 10 esegui
    leggi(n)

    Se i == 0 ∧ n > max allora      //primo numero o maggiore
        max ← n      //salvo il valore massimo
        p ← i      //salvo la posizione
    chiudi Se

    media ← media + n      //somma incrementale
    i ← i + 1      //incremento il contatore
chiudi Finché

media ← media / (i + 1)

stampa("il valore massimo è:")
stampa(max)

stampa("ed è in posizione:")
stampa(p)

stampa("il valore medio è:")
stampa(media)
```

2. Per calcolare lo scarto dalla media di ogni elemento è necessario memorizzare tutti gli elementi, perché la media sarà calcolabile solo dopo aver letto tutti i numeri. Quindi:

```

i ← 0
p ← i
media ← 0
n ← 0      //foglietto per il numero letto

Finché i < 10 esegui
    leggi(n)

    Se i == 0 ∧ n > max allora      //primo numero o maggiore
        max ← n      //salvo il valore massimo
        p ← i      //salvo la posizione
    chiudi Se

    N[i] ← n      //memorizzazione elemento i-esimo
    media ← media + n      //somma incrementale
    i ← i + 1      //incremento il contatore
chiudi Finché

media ← media / (i + 1)
i ← 0

Finché i < 10 esegui      //ciclo su tutti i foglietti
    stampa("lo scarto dalla media di ")
    stampa(N[i])
    stampa("è: ")
    stampa(N[i] - media)      //scarto dalla media
    i ← i + 1
chiudi Finché

stampa("il valore massimo è:")
stampa(max)

stampa("ed è in posizione:")
stampa(p)

stampa("il valore medio è:")
stampa(media)

```

3. No, per i motivi spiegati nella precedente risposta.
4. La deviazione standard è definita come la media degli scarti quadratici dalla media, ovvero:



$$\text{devstd}(\{x_1, \dots, x_M\}) = \sqrt{\frac{\sum_{i=0}^M (x_i - m)^2}{M}}$$

si deve quindi calcolare la somma degli scarti quadratici  $N[i] - m$ , dividere tale numero per il contatore ed estrarne la radice quadrata.

```

i ← 0
p ← i
media ← 0
devstd ← 0
n ← 0      //foglietto per il numero letto

Finché i < 10 esegui
    leggi(n)

    Se i == 0 ∧ n > max allora      //primo numero o maggiore
        max ← n      //salvo il valore massimo
        p ← i      //salvo la posizione
    chiudi Se

    N[i] ← n      //memorizzazione elemento i-esimo
    media ← media + n      //somma incrementale
    i ← i + 1      //incremento il contatore
chiudi Finché

media ← media / (i + 1)
i ← 0

Finché i < 10 esegui      //ciclo su tutti i foglietti
    scarto ← (N[i] − media)
    scarto ← scarto * scarto      //scarto quadratico
    devstd ← devstd + scarto      //somma scarti
    i ← i + 1
chiudi Finché

devstd ← devstd / (i + 1)      //media somma scarti
devstd ← sqrt(devstd)      //deviazione standard

stampa("il valore massimo è:")
stampa(max)

stampa("ed è in posizione:")
stampa(p)

stampa("il valore medio è:")
stampa(media)

stampa("la deviazione standard è:")
stampa(devstd)

```

Come nel caso degli scarti dalla media, anche in questo caso non è possibile risolvere questo problema senza poter memorizzare tutta la sequenza di numeri.