1 Esercizi in pseudocodice

Questa dispensa propone esercizi sulla scrittura di algoritmi in un linguaggio semi-formale, utile all'acquisizione delle abilità essenziali per implementare algoritmi in qualsiasi linguagio di programmazione.

1.1 Algoritmi ed esecutori

Dato un problema ed un opportuno metodo risolutivo, la *risoluzione* di tale problema è quel processo che trasforma i *dati in ingresso* nei corrispondenti *dati finali*.

Si dice *esecutore* quella macchina astratta capace eseguire le istruzioni specificate dall'algoritmo.

Affinché la risoluzione di un problema possa essere realizzata attraverso l'uso di un calcolatore come esecutore, il metodo risolutivo deve poter essere definito come una sequenza di azioni-istruzioni elementari: occorre codificare la soluzione del problema in un algoritmo.

Un *algoritmo* è una sequenza finita di passi, definiti con precisione, che portano alla risoluzione di un compito. Un algoritmo è tale se possiede le seguenti proprietà:

Eseguibilità Ogni azione dev'essere eseguibile dall'esecutore in tempo finito.

Non-ambiguità Ogni azione dev'essere univocamente interpretabile dall'esecutore.

Finitezza Il numero totale di azioni da eseguire, per ogni insieme di dati in ingresso, dev'essere finito.

1.2 Il linguaggio "SIMITA"

Introduciamo uno pseudolinguaggio di programmazione, cioè l'italiano strutturato per ripetizioni e scelte tra alternative, che chiameremo SIMITA.

Immaginiamo di poter scrivere i valori dei dati (sia quelli in ingresso, sia quelli ottenuti durante l'esecuzione del programma) in dei "foglietti".

Sui foglietti possiamo *scrivere*, *leggere*, *cancellare* e *riscrivere*, come nella memoria dei calcolatori.

Per indicare scrittura di un numero N su foglietto f:

$f \leftarrow N$

Allo stesso modo indichiamo la scrittura del contenuto del foglietto g nel foglietto f come:

$f \leftarrow g$

Altre espressioni per codificare delle funzionalità di cui dispone l'elaboratore:

$leggi(f)$ indica l'Operazione di lettura di un valore introdotto dall'utente (ad esempio da tastiera) e la scrittura di tale valore nel foglietto f;

$stampa(f)$ indica l'operazione di lettura del valore contenuto nel foglietto f e la stampa (ad esempio a video) di tale valore. L'istruzione $stampa$ permette anche di stampare dei caratteri, ad esempio $stampa$("ciao come stai") stampa a video i caratteri "ciao come stai".

SIMITA possiede il costrutto condizionale:

```
Istruzione 0

Se condizione allora
    Istruzione 1
    Istruzione 2
    ...
Altrimenti
    Istruzione 3
    Istruzione 4
    ...
chiudi Se
```

```
Istruzione N
```

Dopo aver eseguito "Istruzione 0", viene valutata "condizione". Se "condizione" risulta vera, allora vengono eseguite "Istruzione 1", "Istruzione 2", etc. In caso contrario vengono eseguite solamente "Istruzione 3", "Istruzione 4", etc. Al termine viene eseguita "Istruzione N". Il blocco "**Altrimenti**" è opzionale: se assente, l'esecuzione passa direttamente ad "Istruzione N".

SIMITA possiede anche i costrutti ciclici:

```
Istruzione 0        //questo è un commento

Finché condizione esegui
    Istruzione 1
    Istruzione 2
    ...
chiudi Finché

Istruzione N
```

La sequenza di istruzioni 1 e 2 viene eseguita un numero *finito* di volte. Solo alla prima passata viene eseguita "Istruzione 0". Poi viene valutata "condizione": se risulta vera, allora vengono eseguite "Istruzione 1", "Istruzione 2", etc. In ogni caso viene rivalutata "condizione": se vera, si ripete l'esecuzione di "Istruzione 1" e "Istruzione 2", etc. L'iterazione *termina*, e quindi non vengono più eseguite "Istruzione 1" ed "Istruzione 2", solo quando "condizione" diviene falsa. Quando questo avviene, si esce dal ciclo e si esegue "Istruzione N".

SIMITA ha anche la nozione di *blocchi di foglietti*:

$B[1]$	$B[2]$	$B[3]$	$B[i]$	foglietti
1	2	3	i	indice

$B[0]$ indica il primo foglietto del blocco

$B[1]$ indica il secondo foglietto del blocco

...

$B[n]$ indica l'n-esimo foglietto del blocco

Si noti che i singoli foglietti sono, a tutti gli effetti, dei normali foglietti. Quindi, ad

esempio, le istruzioni seguenti sono valide:

```
i ← 1
f ← B[i]        //scrive il contenuto del primo foglietto in f

B[i] ← 3        //scrive 3 nel primo foglietto

stampa(B[4])        //stampa il contenuto del quarto foglietto

leggi(B[i])       //legge un dato e lo scrive nel foglietto 1
```

1.3 Esercizi

Esercizio 1.1

1. Calcolare il prodotto di due numeri interi positivi e stamparlo a video. Il vostro calcolatore è in grado di eseguire solamente somme e sottrazioni.

2. Produrre una variante ottimizzata dell'algoritmo proposto nella parte precedente.

Esercizio 1.2

Calcolare il fattoriale di un numero e stamparlo a video.

Esercizio 1.3

Calcolare la radice quadrata intera di un numero intero (assunto maggiore di 10) e stamparla a video.

Esercizio 1.4

Determinare se una funzione continua ha uno zero nell'intervallo $[a, b]$ conoscendo il seguente teorema (teorema di Bolzano):

Ipotesi: $f : [a, b] \mapsto \mathbb{R}$, continua in $[a, b] \subseteq \mathcal{R}$, tale che $f(a) \cdot f(b) < 0$

Tesi: $\exists c \in [a, b]$ tale che $f(c) = 0$ (zero di f)

Per valutare la funzione f in un punto generico a potete utilizzare l'istruzione $calcola(f, a, g)$, che scrive il risultato sul foglietto g.

Esercizio 1.5

Determinare almeno uno zero di una funzione continua nell'intervallo $[a, b]$.

Esercizio 1.6

Scrivere un programma che data una sequenza di numeri (positivi) in ingresso restituisce il maggiore e la sua posizione nella sequenza. Supponiamo che la sequenza abbia 10 numeri.

Esercizio 1.7

Si provi ad implementare un algoritmo analogo a quello dell'esercizio precedente, utilizzando un solo ciclo ed eliminando l'assunzione che i numeri inseriti siano positivi.

1. Si estenda l'algoritmo per calcolare la media m dei valori inseriti.

2. Si estenda l'algoritmo per calcolare e stampare a video la differenza $N[i] - m$ tra ogni elemento della sequenza e la media della sequenza.

3. È possibile implementare questi algoritmi senza l'uso di blocchi di foglietti?

4. Si estenda l'algoritmo per calcolare anche la deviazione standard della sequenza, supponendo di avere a disposizione una funzione che calcola la radice quadrata di un numero. È possibile risolvere questo problema senza ricorrere ai blocchi di fogli?