ESERCIZI DI PROGRAMMAZIONE DA SVOLGERE INDIVIDUALMENTE - parte 1 -

1: ESERCIZI DI BASE

- di livello di difficoltà inferiore rispetto all'esame -
- propedeutici per affrontare esercizi più complicati -

Esercizio 1

Acquisire da tastiera un numero intero *n* (ripetendo l'acquisizione in caso di numero negativo) e un numero reale *x*.
Calcolare la sommatoria $\sum_{i=0}^{n} x^i$

Esercizio 2

Scrivere un programma che acquisisce dall'utente due numeri, quindi chiede di inserire la somma. Fino a quando l'utente non inserisce la somma corretta, il programma stampa la frase "Errato: riprova" e ripete l'acquisizione; appena l'utente inserisce la somma corretta, il programma stampa la parola "Bravo" e termina.

Esercizio 3

Acquisire dall'utente la lunghezza dei tre lati di un triangolo fintantoché queste lunghezze non sono positive e non soddisfano la disuguaglianza triangolare (il lato maggiore è inferiore alla somma degli altri due) e calcolare il perimetro del triangolo.

Esercizio 4

Scrivere un programma che calcoli il fattoriale di un numero intero fornito dall'utente.

Esercizio 5

Scrivere un programma che riceva in ingresso una sequenza arbitraria di interi terminata da uno zero e produca come risultato la media intera dei valori di ingresso (escludendo lo zero).

Esercizio 5.2

Scrivere un programma che riceva in ingresso una sequenza arbitraria di interi positivi terminata da uno zero e produca come risultato la media intera dei valori di ingresso (escludendo lo zero).

Ogni volta che viene inserito un numero, il programma deve controllare che sia maggiore o uguale a zero e, nel caso non lo sia, avvisare l'utente dell'errore e riacquisire un nuovo numero.

Esercizio 6

Scrivere un programma che continua ad acquisire un intero fino a quando l'utente inserisce il numero 0, quindi stampa il minimo tra i valori inseriti (zero escluso).

Ad esempio, se l'utente inserisce 10 3 4 7 2 9 9 2 0
il programma stampa 2.

2: ESERCIZI DA ESAME

ESERCIZI DA SVOLGERE DOPO L'ESERCIZIO 1 DEI LUCIDI "ESERCIZI DI PROGRAMMAZIONE 1" (MACCHINETTA DEL CAFFE')

Esercizio 7

Scrivere un programma che riceva in ingresso un numero positivo N e determini il massimo intero K tale che la somma dei primi K interi sia minore o uguale a N.

Ad esempio, se N=20 allora K risulta 5, infatti

$1 + 2 + 3 + 4 + 5 = 15$ mentre

$1 + 2 + 3 + 4 + 5 + 6 = 21$

[Disponibile soluzione alla fine]

Esercizio 8

Acquisire un numero positivo N e calcolarne la radice quadrata intera (ovvero il massimo intero x tale che $x^2 \leq N$).

Esercizio 9

Scrivere un programma che acquisisca da tastiera un numero intero assicurandosi che sia positivo e, successivamente, stampi a video i 5 anni bisestili strettamente superiori al numero acquisito.

[Disponibile soluzione alla fine]

ESERCIZI DA SVOLGERE DOPO GLI ESERCIZI 2 e 3 DEI LUCIDI "ESERCIZI DI PROGRAMMAZIONE 1" (DUE MINIMI e FIBONACCI)

Esercizio 10

Acquisire una serie di numeri da tastiera finché viene inserito lo zero e stampare la somma degli ultimi tre numeri inseriti (0 escluso).

Esercizio 11

Esercizio 8 del Tema d'esame del 7 settembre 2009

Esercizio 12

Acquisire una sequenza di numeri interi e calcolare la somma di quelli positivi. Il programma deve terminare non appena l'utente inserisce per due volte consecutive un valore negativo.

ESERCIZI DA SVOLGERE DOPO L'ESERCIZIO 4 DEI LUCIDI "ESERCIZI DI PROGRAMMAZIONE 1" (VERIFICA NUMERO PRIMO)

Esercizio 13

Calcolare il massimo comun divisore e il minimo comune multiplo di due numeri interi forniti dall'utente.

SOLUZIONI DISPONIBILI

Gli studenti sono fortemente incoraggiati a sperimentare le proprie soluzioni al calcolatore. Nel correggere gli errori che inevitabilmente saranno commessi, si suggerisce di non procedere per tentativi bensì di identificare precisamente la causa del comportamento inatteso del programma sviluppato.

Esercizio 4

Scrivere un programma che calcoli il fattoriale di un numero intero fornito dall'utente.

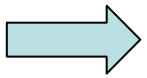
Primo passo: capire il testo e soprattutto il problema (se "non si sa da che parte prendere", provare a risolvere qualche istanza a mano)

5! = 5*4*3*2

3! = 3*2

1! = 1

0! = 1



In generale, N! è pari a:

1	se N = 0 o N = 1
2*…*N	se N > 1

Esercizio 4

Scrivere un programma che calcoli il fattoriale di un numero intero fornito dall'utente.

Secondo passo: individuare un metodo risolutivo

Usare una variabile *fatt*, posta inizialmente a 1.
Successivamente moltiplicarla per 2, …,N
per ottenere il fattoriale:

```
fatt=1
fatt=fatt*2
fatt=fatt*3
…
fatt=fatt*N
```

⇨ Si userà un ciclo per ripetere la moltiplicazione

Esercizio 4

Scrivere un programma che calcoli il fattoriale di un numero intero fornito dall'utente.

Terzo passo: sviluppare l'algoritmo

In questo caso, probabilmente uno arriva quasi al codice…

Come visto, abbiamo dei casi particolari (n=0 e n=1)
e il caso generale

CONSIGLIO: pensare prima al caso generale,
poi considerare i casi particolari!

Esercizio 4

Scrivere un programma che calcoli il fattoriale di un numero intero fornito dall'utente.

```
printf("Inserire il numero N:\n");
scanf("%d",&n);

fatt=1;

for(i=2; i<=n; i++)          // i: numero per cui devo moltiplicare
   fatt=fatt*i;

printf("Fattoriale di %d = %d\n",n,fatt);
```

NB: ci si accorge che i casi particolari sono già risolti!!!
per n = 0 o 1, *fatt* risulta correttamente 1
(il ciclo *for* non viene mai eseguito)

Ovviamente, il programma completo sarà il seguente

```
#include<stdio.h>
#include<stdlib.h>

main(){
int n, fatt, i;

printf("Inserire il numero N:\n");
scanf("%d",&n);

fatt=1;
for(i=2; i<=n; i++)
    fatt=fatt*i;

printf("Fattoriale di %d = %d\n",n,fatt);

system("pause");
}
```

Esercizio 5

Scrivere un programma che riceva in ingresso una sequenza arbitraria di interi terminata da uno zero e produca come risultato la media intera dei valori di ingresso (escludendo lo zero).

Primo passo: capire il problema

Se ho acquisito num_1, num_2, …, num_n:

devo calcolare $$media = \frac{num_1 + num_2 + \ldots + num_n}{n}$$

Esercizio 5

Scrivere un programma che riceva in ingresso una sequenza arbitraria di interi terminata da uno zero e produca come risultato la media intera dei valori di ingresso (escludendo lo zero).

Primo passo: capire il problema

Se ho acquisito num_1, num_2, …, num_n:

devo calcolare $$media = \frac{num_1 + num_2 + \ldots + num_n}{n}$$

Secondo passo: l'idea

Ciclo per acquisire i numeri (esce quando acquisisce 0):

- durante il ciclo, tengo conto di:
  somma + quanti numeri inseriti

Terzo passo: sviluppare l'algoritmo, eventualmente per "raffinamenti successivi" (si può partire da "pseudocodice")

Come sviluppo il ciclo? do-while o while?

- con do-while (una acquisizione si fa in ogni caso)

```
do{
   scanf("%d",&num);
   if(num>0)
     aggiorna somma e la quantità di numeri inseriti;}
while(num>0);

calcola e stampa la media
```

Notare che if(num>0) è fondamentale, ma non serve se uso while…

• con while:

scanf(“%d”,&num);

while(num>0)
  aggiorna somma e la quantità di numeri inseriti;
  scanf(“%d”,&num);

calcola e stampa la media

NB: va bene qualsiasi algoritmo.

Vediamo il codice risultante usando il ciclo while

```
int somma, i, num, media;

printf("Inserire i numeri positivi, 0 per terminare\n");

somma=0;                    // somma parziale
i=0;                        // quanti numeri già inseriti e sommati
scanf("%d",&num);
while(num>0){               // num acquisito ma non ancora sommato
   somma=somma+num;
   i++;
   scanf("%d",&num);
}

if(i!=0){
  media=somma/i;
  printf("Media dei numeri inseriti=%d\n", media);
  }
  else printf("Devi inserire almeno un numero\n");
```

Esercizio 5.2

Scrivere un programma che riceva in ingresso una sequenza arbitraria di interi positivi terminata da uno zero e produca come risultato la media intera dei valori di ingresso (escludendo lo zero).

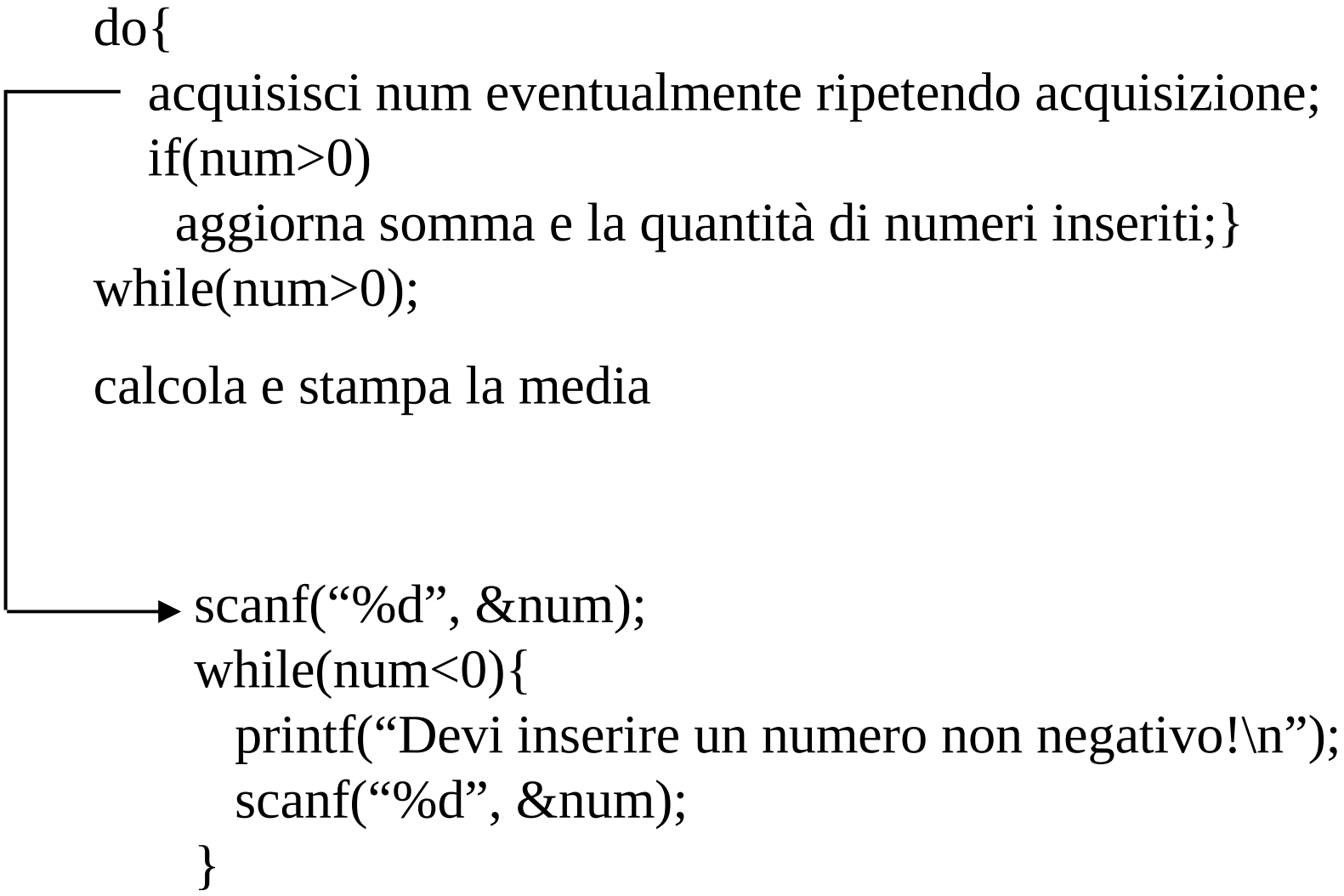
Ogni volta che viene inserito un numero, il programma deve controllare che sia maggiore o uguale a zero e, nel caso non lo sia, avvisare l'utente dell'errore e riacquisire un nuovo numero.

Sviluppare l'algoritmo

dato che l'acquisizione ora è più articolata, uso il do-while…

```
do{
   acquisisci num eventualmente ripetendo acquisizione;
   if(num>0)
     aggiorna somma e la quantità di numeri inseriti;}
while(num>0);
```

calcola e stampa la media


do{
acquisisci num eventualmente ripetendo acquisizione;
if(num>0)
aggiorna somma e la quantità di numeri inseriti;}
while(num>0);
calcola e stampa la media
scanf("%d", &num);
while(num<0){
printf("Devi inserire un numero non negativo!\n");
scanf("%d", &num);
}

```
int somma, i, num, media;
printf("Inserire i numeri positivi, 0 per terminare\n");
somma=0;                          // somma parziale
i=0;                              // quanti numeri già inseriti e sommati
do{
   scanf("%d",&num);              // acquisisci num (con ripetizioni)
   while(num<0){
      printf("Devi inserire un numero non negativo!\n");
      scanf("%d",&num);
      }
   if(num>0){                     // aggiorna somma e i
      somma=somma+num;
      i++; }
while(num>0);

if(i!=0){
  media=somma/i;
  printf("Media dei numeri inseriti=%d\n", media);
  }
  else printf("Devi inserire almeno un numero\n");
```

Esercizio 6

Scrivere un programma che continua ad acquisire un intero fino a quando l'utente inserisce il numero 0, quindi stampa il minimo tra i valori inseriti (zero escluso).

Ad esempio, se l'utente inserisce 10 3 4 7 2 9 9 2 0
il programma stampa 2.

Primo passo: capire il testo e soprattutto il problema (se “non si sa da che parte prendere”, provare a risolvere qualche istanza a mano)

10 3 4 7 2 9 9 2 0

Se avessimo una lista lunga, non potremmo usare il “colpo d’occhio”: partiamo dal primo elemento e proseguiamo a destra individuando il minimo:

10 3 3 3 2 2 2 2 ~~0~~

Secondo passo: individuare un metodo risolutivo

Uso una variabile *min* che tiene traccia del minimo corrente e scandisco gli elementi della sequenza aggiornando *min* ad ogni elemento, fino alla fine

10 3 4 7 2 9 9 2 0

min = 10
3: 3<*min*, quindi *min* diventa 3
4: 4>*min*, quindi *min* rimane uguale (3)
7: 7>*min*, quindi *min* rimane uguale (3)
2: 2<*min*, quindi *min* diventa 2
….

⇒ Si userà un ciclo per scandire la sequenza!

Terzo passo: sviluppare un algoritmo

Prima "bozza":

```
min = primo_numero (da acquisire)
while(numero!=0){
   acquisisci numero
   if(numero!=0)
     aggiorna min}
```

se il numero è 0 non devo aggiornare il minimo!

Conviene usare un do-while? All'inizio devo partire comunque da un numero acquisito…

```
min = primo_numero (da acquisire)
do{
   acquisisci numero
   …. }
   while(numero!=0);
```

SE l'utente inserisce subito zero? Potremmo ipotizzare che questo non avvenga, ma comunque conviene continuare con la prima idea…

```
int num, min;

printf(“Inserire un numero ”);
scanf(“%d”,&num);
min=num;

while(num!=0){
    printf(“Inserire un numero ”);
    scanf(“%d”,&num);
    if(num!=0 && num<min)
      min=num;
    }

printf(“Il minimo è %d\n", min);
```

SOLUZIONE ESERCIZIO 7

Scrivere un programma che riceva in ingresso un numero positivo N e determini il massimo intero K tale che la somma dei primi K interi sia minore o uguale a N.

Ad esempio, se N=20 allora K risulta 5, infatti

$1 + 2 + 3 + 4 + 5 = 15$ mentre

$1 + 2 + 3 + 4 + 5 + 6 = 21$

Primo passo (se si è in difficoltà): provare a risolvere a mano qualche istanza del problema

P. es. N=6

$1 + 2 + 3 = 6 \leq 6$

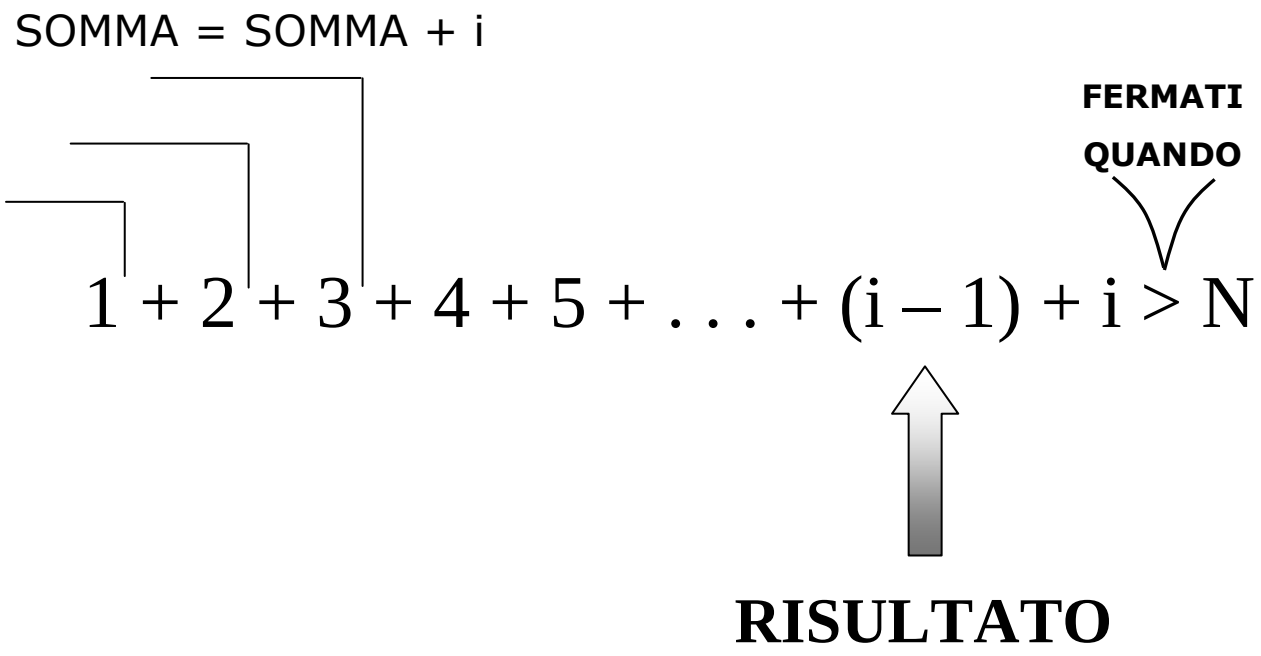
$1 + 2 + 3 + 4 = 10 > 6$

$\Rightarrow$ Risultato: 3

Secondo passo: individuare un metodo risolutivo

Dato N

- una variabile *i* scandisce i numeri 1, 2, 3, … (userò un ciclo)
- una variabile *somma* tiene conto della somma parziale



Terzo passo: sviluppare l'algoritmo:

è importante dare un significato preciso alle variabili!

```
int i, somma, N, K;

printf("Inserire il numero positivo N\n");
scanf("%d", &N);

i=1;                    // ultimo indice già sommato
somma=1;                // somma corrente (avendo già sommato i)
while(somma<=N){        // esci quando somma supera strettamente N
   i++;
   somma=somma+i;
}

K = i-1;
```

SOLUZIONE (PARZIALE) ESERCIZIO 9

Scrivere un programma che acquisisca da tastiera un numero intero assicurandosi che sia positivo e, successivamente, stampi a video i 5 anni bisestili strettamente superiori al numero acquisito.

Sul problema c'è poco da capire: la difficoltà sta nell'algoritmo…

```
acquisisci N;
numbisestili=0;          // numero di anni trovati (e stampati)
ultbisestile=N;          // ultimo anno bisestile trovato (quasi)
while(numbisestili<5){   // esci quando sono già stati trovati 5 anni
    calcola e stampa il primo anno bisestile superiore a ultbisestile;
    aggiorna ultbisestile con l'anno trovato;
    numbisestili++;
}
```

VEDIAMO LA PARTE CENTRALE DEL CODICE…

```
numbisestili=0;                 // numero di anni trovati e stampati
ultbisestile=n;                 // (serve trovare il successivo bisestile)

while(numbisestili<5){
        ultbisestile++;
        while(   !(  ((ultbisestile % 100 != 0) && (ultbisestile % 4 ==0))
                   || (ultbisestile % 400 ==0))  )
                ultbisestile++;
        numbisestili++;
        printf("Anno bisestile successivo numero %d = %d\n",
                                                numbisestili, ultbisestile);
}
```

Trova bisestile successivo