Programmazione I (Tucci/Distasi)

PR1 MT/RD 28/06/2022

Modello: 1

Cognome: _____

Nome: _____

Matricola: _____

Email: _____

Regole del gioco: Compilare i dati personali prima d'incominciare. Una volta iniziata la prova, non è consentito lasciare l'aula. Usare questi stessi fogli (compreso il retro, dove necessario) per rispondere. *Buon lavoro!*

1. L'*indice di massa corporea* (Body Mass Index, BMI) è una misura della relazione fra massa e statura che ci dice se siamo sottopeso, nella norma o sovrappeso. Si calcola come rapporto fra la massa corporea espressa in chilogrammi e il quadrato della statura espressa in metri. Le sue unità di misura sono quindi $\mathrm{kg/m^2}$.

   $$\mathrm{BMI} = \frac{\text{peso in kg}}{(\text{statura in metri})^2}.$$

   Consideriamo il seguente tipo di dati

   ```
   typedef struct {
       char nome[50];      // nome, cognome o pseudonimo
       double peso;        // in kg
       double statura;     // in m
   } Paziente;
   ```

   e scriviamo una funzione

   ```
   Paziente * solo_magri(Paziente * pazienti, int n, int * nmagri)
   ```

   che prende in input un array `pazienti[]` di n oggetti di tipo `Paziente` e restituisce un nuovo array contenente soltanto i pazienti con un BMI che non supera $18.5\,\mathrm{kg/m^2}$. Il risultato sarà allocato dinamicamente in dipendenza del numero di elementi che soddisfano la condizione. Il parametro output `nmagri` rifletterà la dimensione dell'array risultato. Nel caso nessun elemento soddisfi la condizione, la funzione `solo_magri()` ritornerà `NULL` e imposterà `nmagri` a zero.

   Per lo svolgimento di questo esercizio, scrivere ed usare una funzione ausiliaria

   ```
   double BMI(Paziente *p)
   ```

   che, dato un puntatore `p` a un oggetto di tipo `Paziente`, ne restituisce il valore di BMI.

2. Con riferimento all'esercizio precedente, scrivere una funzione

```
int lista_normali(FILE *fout, Paziente pazienti[], int n)
```

che, ricevendo un vettore `pazienti[]` di `n` oggetti `Paziente`, per ogni paziente con BMI compreso fra 18.5 e 24.9 $\mathrm{kg/m^2}$ scrive in un file di testo già aperto (e puntato da `fout`) una riga con nome, statura e peso separati da spazi. La funzione `lista_normali()` restituisce 1 se tutto è andato bene, 0 se c'è stato qualche errore.

Risposte per il modello 1

1. L'*indice di massa corporea* (Body Mass Index, BMI) è una misura della relazione fra massa e statura che ci dice se siamo sottopeso, nella norma o sovrappeso. Si calcola come rapporto fra la massa corporea espressa in chilogrammi e il quadrato della statura espressa in metri. Le sue unità di misura sono quindi $\mathrm{kg/m^2}$.

$$\mathrm{BMI} = \frac{\text{peso in kg}}{(\text{statura in metri})^2}.$$

Consideriamo il seguente tipo di dati

```
typedef struct {
    char nome[50];      // nome, cognome o pseudonimo
    double peso;        // in kg
    double statura;     // in m
} Paziente;
```

e scriviamo una funzione

```
Paziente * solo_magri(Paziente * pazienti, int n, int * nmagri)
```

che prende in input un array `pazienti[]` di n oggetti di tipo `Paziente` e restituisce un nuovo array contenente soltanto i pazienti con un BMI che non supera $18.5\,\mathrm{kg/m^2}$. Il risultato sarà allocato dinamicamente in dipendenza del numero di elementi che soddisfano la condizione. Il parametro output `nmagri` rifletterà la dimensione dell'array risultato. Nel caso nessun elemento soddisfi la condizione, la funzione `solo_magri()` ritornerà `NULL` e imposterà `nmagri` a zero.

Per lo svolgimento di questo esercizio, scrivere ed usare una funzione ausiliaria

```
double BMI(Paziente *p)
```

che, dato un puntatore `p` a un oggetto di tipo `Paziente`, ne restituisce il valore di BMI.

Risposta
Ecco una possibile soluzione.

```
#define BMI_MAGREZZA    18.5
#define BMI_SOVRAPPESO  24.9


double BMI(Paziente * p)
{
  return p->peso / (p->statura * p->statura);
}

Paziente *copy_paziente(Paziente * dest, Paziente * source)
{
  strcpy(dest->nome, source->nome);
  dest->peso = source->peso;
  dest->statura = source->statura;
  return dest;
}
```

```
Paziente *solo_magri(Paziente * pazienti, int n, int *nmagri)
{
  int i, j, contamagri = 0;
  Paziente *magri;              // sara' array risultato

  for (i = 0; i < n; i++)       // contiamo i "magri"
    {
      if (BMI(&pazienti[i]) <= BMI_MAGREZZA)
        {
          contamagri++;
        }
    }
  if (contamagri == 0)
    {
      *nmagri = 0;
      return NULL;
    }
  // else
  magri = malloc(sizeof(Paziente) * contamagri);
  if (magri == NULL)
    {
      fprintf(stderr, "malloc(%ld) failed. Sorry, bye.\n",
              sizeof(Paziente) * contamagri);
      exit(-1);
    }
  j = 0;
  for (i = 0; i < n; i++)
    {
      if (BMI(&pazienti[i]) <= BMI_MAGREZZA)
        {
          copy_paziente(&magri[j], &pazienti[i]);
        }
    }
  *nmagri = contamagri;
  return magri;
}
```

2. Con riferimento all'esercizio precedente, scrivere una funzione

   ```
   int lista_normali(FILE *fout, Paziente pazienti[], int n)
   ```

   che, ricevendo un vettore `pazienti[]` di `n` oggetti `Paziente`, per ogni paziente con BMI compreso fra 18.5 e 24.9 $\mathrm{kg/m^2}$ scrive in un file di testo già aperto (e puntato da `fout`) una riga con nome, statura e peso separati da spazi. La funzione `lista_normali()` restituisce 1 se tutto è andato bene, 0 se c'è stato qualche errore.

   Risposta
   Ecco una possibile soluzione.

```
#include <stdio.h>
#include "paziente.h"
#include "protos.h"
#include "bmi.h"

int lista_normali(FILE * fout, Paziente pazienti[], int n)
{
  int i, status;
  double this_BMI;

  for (i = 0; i < n; i++)
    {
      this_BMI = BMI(&pazienti[i]);
      if (this_BMI >= BMI_MAGREZZA && this_BMI <= BMI_SOVRAPPESO)
        {
          status = fprintf(fout, "%s %g %g\n",
                           pazienti[i].nome, pazienti[i].statura,
                           pazienti[i].peso);
          if (status != 3)      // some error
            {
              fclose(fout);
              return 0;
            }

        }
    }
  fclose(fout);
  return 1;                     // everything OK
}
```