Fondamenti di Informatica A
Ing. Elettronica e delle Telecomunicazioni
Esercitazione 5 - Soluzioni
14-15 Novembre 2006

Puntatori e funzioni

ESERCIZIO n° 1:

Realizzare un programma che, facendo uso di un vettore allocato dinamicamente, gestisca i dati relativi ai risultati di una gara di sci. In particolare, una volta noto il numero N di partecipanti alla gara, il programma dovra` allocare dinamicamente un vettore, nel quale ogni elemento rappresenta il risultato di un diverso atleta, tenendo conto che per ogni atleta e` necessario memorizzare:

- **Numero:** un intero che rappresenta il numero di pettorale e individua univocamente lo sciatore;
- **Cognome:** una stringa che contiene il cognome dell'atleta;
- **Nome:** una stringa che contiene il nome dell'atleta;
- **Prima_manche:** un intero che rappresenta il numero di secondi impiegati per completare la prima manche della gara; (se questo valore e` uguale a -1, significa che lo sciatore e` caduto durante la manche, ed e` stato quindi eliminato)
- **Seconda_manche:** un intero che rappresenta il numero di secondi impiegati per completare la seconda manche (se questo valore e` uguale a -1, significa che lo sciatore e` caduto durante la manche, ed e` stato quindi eliminato).

Naturalmente, vince la gara lo sciatore che ha ottenuto il **tempo totale** (Prima_manche + Seconda_manche) **minimo** tra tutti.

Una volta acquisiti i dati relativi alla gara, il programma dovra`:

1. Stampare numero, nome e cognome degli sciatori eliminati;
2. Stampare numero, nome e cognome del vincitore. A tal scopo utilizzare una funzione con intestazione del tipo **`struct atleta vincitore(struct atleta *V);`**
3. Stampare la **graduatoria finale** degli sciatori non eliminati (cioe` l'elenco ordinato in ordine di tempo totale crescente), visualizzando, per ogni sciatore numero, nome, cognome e tempo totale. Utilizzare una procedura con intestazione del tipo **`void ordina(struct atleta *V);`** per ordinare gli atleti.

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

typedef struct { int numero;
                 char cognome[31];
                 char nome[31];
                 int prima;
                 int seconda;
               } Atleta;
int n;

Atleta vincitore(Atleta *V)
{int best=-1,i, index;

for (i=0; i<n; i++)
   if ((V+i)->seconda!=-1)
     {int tempo=(V+i)->prima+(V+i)->seconda;
      if ((best==-1)||(tempo<best))
        {best=tempo;
         index=i;
        }
     }

 return V[index];
}
```

```
void ordina(Atleta *V)
{int i,j,min;
 Atleta tmp;

 for (i=0; i<n; i++)
   {min=i;
    for (j=i+1; j<n; j++)
      if (((V+j)->prima+(V+j)->seconda) < ((V+min)->prima+(V+min)->seconda))
        min=j;
      if (min!=i)
        {tmp=V[i];
         V[i]=V[min];
         V[min]=tmp;
        }
   }
}

void main()
{int i ;
 printf("Quanti atleti partecipano alla gara ? ");
 scanf("%d",&n);

 Atleta* gara= (Atleta *) malloc(n*sizeof(Atleta));

 for (int i=0; i<n; i++)
   {printf("Atleta #.%d : pettorale ",i);
    scanf("%d",&(gara+i)->numero);
    fflush(stdin);
    printf("Atleta #.%d : cognome ",i);
    gets((gara+i)->cognome);
    printf("Atleta #.%d : nome ",i);
    gets((gara+i)->nome);
    printf("Atleta #.%d : tempo prima manche ",i);
    scanf("%d",&(gara+i)->prima);
    if ((gara+i)->prima!=-1)
      {printf("Atleta #.%d : tempo seconda manche ",i);
       scanf("%d",&(gara+i)->seconda);
      }
    else (gara+i)->seconda=-1;
   }

 printf("Sciatori eliminati\n");
 for (i=0; i<n; i++)
   if ((gara+i)->seconda==-1)
     printf("%s %s, pettorale #%d\n", (gara+i)->cognome, (gara+i)->nome, (gara+i)->numero);

 Atleta winner=vincitore(gara);
 printf("Vincitore\n");
 printf("%s %s, pettorale #%d : Tempo %d\n", winner.cognome, winner.nome,
                                             winner.numero , winner.prima + winner.seconda);

 int offset=0;
 for (i=0; i<n; i++)
   {if ((gara+i)->seconda==-1)
      {offset++;
       continue;
      }
    if (offset)
      gara[i-offset]=gara[i];
   }
n-=offset;

ordina(gara);

printf("Risultato finale\n");
for (i=0; i<n; i++)
   printf("%s %s, pettorale #%d : Tempo %d\n",(gara+i)->cognome, (gara+i)->nome ,
                                              (gara+i)->numero , (gara+i)->prima + (gara+i)->seconda);
}
```

ESERCIZIO n° 2:

Definire una funzione **sommatoria** che, dati due valori reali x e y, e un intero positivo N, calcoli come risultato la funzione reale:

$$\sum_{i=1}^{N}(x+y)^{i}$$

La funzione dovra` quindi prevedere un'intestazione del tipo:

float sommatoria(float x, float y, int N)

Realizzare un programma che, dati x, y e N da standard input, mediante la funzione **sommatoria** calcoli la sommatoria sui valori effettivamente letti, e successivamente la stampi.

```
#include <stdio.h>

float potenza(float b, int e)
{int i;
 float pot=1;
 for (i=0; i<e; i++)
   pot*=b;
 return pot;
}

float sommatoria(float x, float y, int N)
{int i;
 float somma=0;
 for (i=1; i<=N; i++)
   somma+=potenza(x+y, i);
 return somma;
}

void main()
{float x, y;
 int N;

 printf("Inserire i tre valori x, y e N separati da spazio : ");
 scanf("%f %f %d",&x, &y, &N);
 printf("Il risultato e' %f",sommatoria(x, y, N));
}
```

ESERCIZIO n° 3:

Realizzare un programma in grado di effettuare alcune operazioni sul figure geometriche di tipo triangolari.
A questo scopo progettare un tipo di dato strutturato non primitivo triangolo in grado di rappresentare la figura geometrica mediante i tre lati e l'altezza. Il programma dovra` essere in grado di calcolare, per ogni triangolo dato da input, mediante opportune funzioni, l'area e il perimetro:
Utilizzare inoltre una procedura che calcoli e stampi le caratteristiche.
Le funzioni e la procedura dovranno avere un'intestazione del tipo:

float Area(struct triangolo T);
int Perimetro (struct triangolo T);
void Stampa (struct triangolo V[], int n);

```
#include <stdio.h>

typedef struct { int base, altezza, lato1, lato2;
               } triangolo;

typedef triangolo vet[20];

float Area(triangolo T)
{return (T.base*T.altezza)/2.0;
}

int Perimetro(triangolo T)
{return T.base + T.lato1 + T.lato2;
}

void Stampa(vet V, int n)
{int i;
 for (i=0; i<n; i++)
   printf("Base %d\tAltezza %d\tLati %d,%d\tArea %f\tPerimetro %d\n",
          V[i].base, V[i].altezza, V[i].lato1, V[i].lato2, Area(V[i]), Perimetro(V[i]));
}

void main()
{int b, h, l1, l2, n=0, i;
 vet V;

 do
   {printf("Inserire i dati del triangolo (0 per terminare) \n Base : ");
    scanf("%d",&b);
    printf("Altezza : ");
    scanf("%d",&h);
    printf("Primo lato : ");
    scanf("%d",&l1);
    printf("Secondo lato : ");
    scanf("%d",&l2);
    if (b!=0)
      if ((b+l1>l2)&&(b+l2>l1)&&(l1+l2>b)&&(h>0))
        {V[n].base=b;
         V[n].altezza=h;
         V[n].lato1=l1;
         V[n++].lato2=l2;
        }
      else printf("Non e' un triangolo\n");
   }
 while (b!=0);

 Stampa(V,n);
}
```