Informatica

C++

Prof. Bortolai Gabriele

C++

Funzioni

Fino ad adesso abbiamo visto le basi della programmazione e i programmi scritti fino ad ora potrebbero risultare non ottimizzati e poco ordinati.
Al fine di ovviare questo problema é possibile usare le funzioni, consideriamo uno dei vecchi programmi.

C++

Funzioni

Il programma in questione é uno dei primi visti a lezione, ovvero quello in cui viene calcolata la media dei voti.

All’interno del main() sono scritte tutte le operazioni che deve svolgere il programma e ad un lettore poco attento potrebbe essere difficile capire, quindi bisogna trovare un modo per snellire il programma.

Il codice ha il compito di calcolare il valor medio di numeri interi, quindi dobbiamo scrivere una funzione che svolga le medesime operazioni.

```cpp
#include <iostream>

using namespace std;

int main(){

  int i,k,voto;
  double sum,media;

  cout<<"Inserire il numero di vot"<<endl;
  cin>>k;

  for(i=0;i<k;i++){

    cout<<"inserisci il voto"<<endl;
    cin>>voto;

    sum=sum+voto;

  }

  media=sum/k;
  if(media<3){
    cout<<"La media vale :"<<media<<" la tua media fa un po' schifo forse é megio che studi"<<endl;
  }
  else if(media>3 && media<5){
     cout<<"La media vale :"<<media<<" la tua media non fa troppo schifo forse é megio che studi"<<endl
  }

  else if(media>6 && media<8){
    cout<<"La tua media vale :"<<media<<" bravo/a sei sufficente"<<endl;
  }

  else if(media>8){
    cout<<"La tua media vale :"<<media<<" bravo/a il tuo premio é la bocciatura"<<endl;
  }

  return 0;
}
```

C++

Funzioni

Utilizzando le funzioni il main() risulta molto più ordinato e sopratutto molto snello.

Per definire una funzione si procede nel modo seguente:

1. Nel preambolo definire che tipo di risultato restituisce nel nostro caso un double e definire il suo argomento, nel nostro caso un intero.

2. Nel main() scrivere il codice di programma e per usare la funzione basterà passargli l'argomento nel nostro caso il numero di voti.

3. Dopo aver chiuso il main() bisogna scrivere le operazioni che deve svolgere la funzione nel nostro caso deve calcolare la media dei voti.

Vediamo un altro esempio più interessante.

```
using namespace std;

//Function
double  media(int);

int main(){

  int n;

  cout<<"Inserisci il nuemro di voti"<<endl;
  cin>>n;

  media(n);

  cout<<media(n)<<endl;

  return 0;
}
```

Prototipo della funzione

Codice

```
//Define function
double media(int n){

  double somma,voto;

  for(int i=1;i<n;i++){

    cout<<"Insrisci i voti"<<endl;
    cin>>voto;

    somma=somma+voto;

  }

  return somma/n;

}
```

Definizione e operazione che svolgerà la funzione

C++

Funzioni

Ottimizziamo il programma che aveva per compito un vostro compagno.

L’esercizio consisteva nel leggere un file composto da un numero sconosciuto di elementi suddivisi in blocchi da 5 per un totale di 10 blocchi e calcolare la media dei valori in ogni blocco e scrivere i risultati su un file.

Segue il programma che scrissi io.

```cpp
#include <iostream>
#include <fstream>
#include <string>

using namespace std;

int main(){

  int n,i,k;

  string filename="dati.dat";
  double d,x,sumx,sumy,Medx,Medy;

  ifstream ifile;
  ofstream ofile;
  cout<<"inserisci il nome del file"<<endl;

  ifile.open(filename.c_str());
  if(!ifile){
    cout<<"errore"<<endl;
    return 1;
  }

  n=0;
  while(ifile>>d>>x){

    n++;
  }
    cout<<n<<endl;
  double* X=new double [n];
  double* Y=new double [n];

   ifile.clear();
   ifile.seekg(ios::beg);
   for(i=0;i<n;i++){
     ifile>>X[i]>>Y[i];
   }

   ofile.open("risultati.dat");

   for(i=0;i<10;i++){
     sumx=0;
     sumy=0;
     for(k=(5*i);k<(5*i)+5;k++){
       sumx=sumx+X[k];
       sumy=sumy+Y[k];
   }

     Medx=sumx/5;
     Medy=sumy/5;
     ofile<<Medx<<" "<<Medy<<endl;

   }

     ifile.close();
     ofile.close();

  return 0;
}
```

C++

Funzioni

Ció che dobbiamo fare é riscrivere il programma in modo che venga più ordinato. Per farlo ci serviranno le seguenti funzioni:

1. Una funzione che ci calcoli la dimensione del file che chiameremo **size**.
2. Una funzione che ci permetta di calcolare i valori medi che chiameremo **media**.

C++

Funzioni

La funzione **size** dovrà leggere una stringa e restituiti un numero intero, quindi il suo prototipo sarà della seguente forma:

Int size(string)

E dovrá contare quanto é grande per eseguire questa operazione é possibile operare in maniera analoga a come é stato fatto per lo stream da file, segue il codice che svolge tale operazione.

C++

Funzioni

Nella immagine adiacente é possibile vedere come é stata implementate la funzione, da notare che rispetto al preambolo la definizione della funzione **size** é più precisa.

Infatti si chiede che la funzione restituisca un valore int e che deve ricevere un valore di tipo string che si chiama filename.

In fine la funzione restituisce tramite il comando **return** un interno **n** ovvero la dimensione del file composto da due colonne, infatti nel main () se dessimo il comando cout<<size(“dati.dat”)<<endl stamperà sul terminale la dimensione del file.

```
//Define functions
int size (string filename){

  double d,f;
  int n=0;

  ifstream ifile;
  ifile.open(filename.c_str());

  while (ifile>>d>>f){

    n++;

}

 ifile.clear();
 ifile.seekg(ios::beg);

return n;
  }
```

C++

Funzioni

La funzione **media** é molto più difficile da definire sia teoricamente che tramite codice.

Il calcolo della media é stato svolto nel modo classico che si é sempre usato e i risultati delle varie medie sono stati scritti su un array, così facendo si sono scritti tutti i risultati su un unico operatore.

Quindi teoricamente la funzione dovrebbe leggere un elenco di dati che sono stati scritti precedentemente su un vettore e dovrebbe restituire un array con all’interno tutti i risultati.

Peccato che le funzioni non restituiscano array.

C++

Funzioni

Per ovviare a questo problema é possibile usare i puntatori, nel modo seguente.

Possiamo definire una funzione che legga in ingresso un vettore e la sua dimensione e che restituisca un puntatore che “ Punterà “ alle componenti dell’array su cui sono scritti i risultati.

```
//Functions
int size (string);
double* media (int n, double [n]);
```

C++

Funzioni

Nella definizione della funzione gli diremo che la dimensione del vettore che deve leggere vale **n** e che il vettore si chiama **v**.

Poi definiamo il puntatore **mid** che successivamente punterà alle componenti del vettore risultante.

Svolgiamo i calcoli e scriviamoli sul vettore **mid** che successivamente con il comando **return** verrà convertito in un puntatore e potremmo usarlo nel main ().

```
double* media (int n, double v[n]){

  double sum1,sum2;
  double* mid = new double [n];

  for(int i=0;i<n;i++){
      sum1=0;

      for(int k=5*i;k<5*i+5;k++){

        sum1=sum1+v[k];

      }

    mid[i]=sum1/5;
    }

    return mid;


}
```

C++

Funzioni

Segue il programma ottimizzato, da notare quando é più snello il mai() rispetto a prima.

```cpp
#include <iostream>
#include <fstream>

using namespace std;

//Functions
int size (string);
double* media (int n, double [n]);


int main(){

  string filename= "dati.dat";
  int n;

  n=size(filename);

  ifstream ifile;
  ifile.open(filename.c_str());

  ofstream ofile("risultati.dat");

  double* x= new double [n];
  double* y= new double [n];

  for(int i=0;i<n;i++){

    ifile>>x[i]>>y[i];

  }

 double* med1=media(10,x);
 double* med2=media(10,y);

 for(int i=0;i<n/5;i++){

    ofile<<med1[i]<<" "<<med2[i]<<endl;

 }

  ifile.close();
  ofile.close();
  return 0;

}

//Define functions
int size (string filename){

  double d,f;
  int n=0;

  ifstream ifile;
  ifile.open(filename.c_str());

  while (ifile>>d>>f){

    n++;
```

Prima parte

```
  while (ifile>>d>>f){

    n++;

  }

   ifile.clear();
   ifile.seekg(ios::beg);

  return n;
    }

double* media (int n, double v[n]){

  double sum1,sum2;
  double* mid = new double [n];

  for(int i=0;i<n;i++){
     sum1=0;

     for(int k=5*i;k<5*i+5;k++){

       sum1=sum1+v[k];

     }

    mid[i]=sum1/5;
    }

    return mid;


}
```

Seconda parte

C++

Funzioni

Inoltre esistono molti altri tipi di funzione, ad esempio:

1. Int valori(double){}
2. Double voto(int){}
3. Double* gatto (int n, double[n]){}
4. Double* cane (double [3][4]){}

Nell’esempio uno si é definito una funzione che legge un valore double e restituisce un interno, nella seconda invece la funzione legge un intero e restituisce un double.

Nella terza invece legge un intero, non che la lunghezza del vettore, e un vettore double e restituisce un puntatore double.

Nella quarta invece legge una matrice double e restituire un puntatore double.

Esistono molti altri tipi di funzioni e in base alle proprie esigenze vanno definite.

C++

Funzioni

Infine una definizione di funzione molto importante é il **void**, questo tipo di funzioni non hanno un return e svolgono solo delle operazioni.

Ad esempio scambiare le componenti di un vettore, tali funzioni si definiscono in maniera analoga:

Void funzione (int){}

In questo caso la funzione legge un valore intero e svolge delle operazioni.

C++

Funzioni

Sorge immediatamente una domanda: é possibile scrivere le definizione delle funzioni **size** e **media** da “ Un’altra parte “ e far si che il programma diventi ancora più ordinato e pulito ?

La risposta é ovviamente si e lo vedremo più avanti.

C++

Funzioni

Per chi volesse provare ad usare le funzioni può riscrivere il programma che usa il metodo di Cramer per risolvere i sistemi usando le funzioni.

Allegherò la soluzione del compito nei file di Class Room.