Tipi di dato strutturato - Struct

mercoledì 28 ottobre 2020 21:15

il costruttore di tipo `struct`

- In C e C++ abbiamo un costruttore di tipo che ci permette di **aggregare dati non omogenei**

```
struct nomestruct
{
    tipo1 campo1;
    tipo2 campo2;
    …
    tipoN campoN;
};
```

Un esempio semplice

```
struct product{
      string description;
      string kind;
      double price;
 } ;
```

- Lo struct definisce un nuovo tipo

```
struct nomestruct
{
    tipo1 campo1;
    tipo2 campo2;
    …
    tipoN campoN;
};

nomestruct variabile;
```

Un esempio semplice

```
struct product{
      string description;
      string kind;
      double price;
 } ;

product P;
```

variabile P di tipo Product

- Come accedere ai campi di uno struct

```
struct nomestruct
{
    tipo1 campo1;
    tipo2 campo2;
    …
    tipoN campoN;
};

nomestruct variabile;
```

product P;

P. Kind è di tipo string

P. price è di tipo double

- variabile.campo1 è di tipo1

Esempio

```
struct studente {
    string cognome;
    string nome;
    unsigned int matricola;
};

int main ()
{
    studente S;

    cout << "Inserire il cognome senza spazi" << endl;
    cin >> S.cognome;

    cout << "Inserire il nome senza spazi" << endl;
    cin >> S.nome;

    cout << "Inserire la matricola" << endl;
    cin>>S.matricola;

    cout << "Cognome: " << S.cognome << "\t Nome: " << S.nome << "\t Matricola: " << S.matricola << endl;
    cout << endl;

    return(0);
}
```

lo struct definisce un nuovo tipo

notazione col punto:
`nomestruct.nomecampo`

Esempio

```cpp
#include <iostream>
#include <cmath>

using namespace std;

struct Point {
    double x;
    double y;
};

int main() {
    Point P;
    Point O={0.0,0.0};

    cout << "Insert point x y\n";
    cin >> P.x >>P.y;

    cout << "O: " << O.x << " " << O.y << endl;
    cout << "P: " << P.x << " " << P.y << endl;

    cout << "The distance of the point from the origin is: ";

    double d=sqrt((P.x-O.x)*(P.x-O.x)+(P.y-O.y)*(P.y-O.y));
    cout << d << endl;
    return 0;
}
```

N.B. Si pone prima del main per avere una visibilità all'interno del programma maggiore

Inizializzazione contestuale alla dichiarazione

posso dichiararla dentro il main, ma solo lì dentro posso richiamarla (= meno utilizzabile)

Struct annidati - esempio

```cpp
struct movies_t {
    string title;
    int year;
};

struct friends_t {
   string name;
   string email;
   movies_t favorite_movie;
} charlie, maria;
```

crea un nuovo "tipo" di variabile

posso dichiarare già qui le variabili di questo tipo (occhio alla scope!)

Attenzione!

- Per accedere ad una struct in lettura o scrittura occorre riferirsi esplicitamente ai suoi campi

  e.g. D.day=20;

- Per confrontare due variabili della stesso tipo strutturato occorre confrontare i campi corrispondenti (in corsi successivi impareremo a fare diversamente)

  Date D1, D2;
  non si può confrontare le due variabili direttamente (no D1>D2, D1==D2, …)

Esempio:

```cpp
#include <iostream>

using namespace std;

struct studente {
    string cognome;
    string nome;
    unsigned int matricola;
};

int main ()
{
    studente S;

    while (true)
    {
        cout << "Inserire il cognome" << endl;
        getline(cin,S.cognome);     // CON get line POSSIAMO INSERIRE STRING CON SPAZI
        if (S.cognome=="FINE")
            break;
        cout << "Inserire il nome" << endl;
        getline(cin,S.nome);
        cout << "Inserire la matricola" << endl;
        cin>>S.matricola;

        studente S1=S;
        cout << "Cognome1: " << S1.cognome << "\t Nome: " << S1.nome << "\t Matricola: " << S1.matricola << endl;
        cout << endl;
        cin.ignore(10000,'\n'); //pulisce il buffer dall'ultimo a capo rimasto
    }

    return(0);
}
```