UNIVERSITÀ
DI TRENTO

Dipartimento di Ingegneria e
Scienza dell'Informazione
DISI - Trento

Programmazione 1

17 - Esercitazione

Matteo Franzil
matteo.franzil@unitn.it

Anno Accademico 2023/2024

Nelle puntate precedenti

- **Strutture** (`struct`)
  - Sono collezioni ordinate di elementi non omogenei (ad esempio, stringhe, interi, float, etc.). Permettono di definire **nuovi tipi.**

```
struct punto2D {
  int x;
  int y;
};
```

```
struct libro  {
  char titolo[20];
  char autore[20];
  float prezzo;
};
```

```
struct lista  {
  float elemento;
  lista * precedente;
};
```

Nelle puntate precedenti

- **Strutture** (`struct`)
  - Sono collezioni ordinate di elementi non omogenei (ad esempio, stringhe, interi, float, etc.). Permettono di definire **nuovi tipi.**

```
struct punto2D {
  int x;
  int y;
};
```

```
struct libro  {
  char titolo[20];
  char autore[20];
  float prezzo;
};
```

```
struct lista  {
  float elemento;
  lista * precedente;
};
```

Nelle puntate precedenti

- **Strutture** (`struct`)
  - Una struttura può essere inizializzata con una lista ordinata dei valori dei campi rispettivi (in caso si ometta qualcosa, quel campo verrà inizializzato al valore zero di quel tipo);
  - Se **s** è una struttura e **campo** è un elemento della struttura. **s.campo** ci permette di accedere al valore di quell'elemento. Se **s** è un puntatore ad una struttura, allora **s->campo** è l'istruzione equivalente;

Nelle puntate precedenti

- **Strutture** (`struct`)

```
struct macchina  {
  char modello[20];
  char proprietario[20];
  float prezzo;
};

macchina volvo = {"XC60", "Rossi Luca", 40000};
cout << volvo.modello << endl;
cin >> volvo.prezzo;
strcpy(volvo.modello, "XC40");
```

1 - Esse3++ (Studenti)

Scrivere un programma che definisca una struttura chiamata **Studente** formata dai campi **nome (array di 30 caratteri)**, **cognome (array di 30 caratteri)**, **matricola (int)** e **media (float)** ponderata dei voti. Implementare poi le funzioni definite in basso: la funzione `stampa_studente` stampa a video il contenuto di una struttura Studente, mentre la funzione `genera_studente` genera una struttura Studente dati in input il valore dei suoi campi.
Si può utilizzare la funzione `strcpy` della libreria `cstring`.

```
void stampa_studente(Studente * S);

Studente* genera_studente(char nome[], char cognome[], int
matricola, float media_ponderata);
```

2 - Esse3++ (Database)

Utilizzando le funzioni scritte precedentemente, scrivere un programma che, dato in input un file contenente i dati di alcuni studenti e il numero di righe del file, generi un array dinamico che contenga tutti questi studenti. E' consigliato raggruppare questa logica all'interno di una **funzione** separata. Utilizzare la libreria `fstream`.

```
Studente * database = new Studente[N];
```

Link al file con gli studenti:
https://pastebin.com/AGXNyhK6

3 - Esse3++ (Ricerca)

Modificare il programma scritto precedentemente implementando le funzioni definite sotto: la funzione `cerca_matricola` permette di cercare uno studente tramite il numero di matricola, la funzione `cerca_nome_e_cognome` permette di cercare uno studente per nome e cognome, mentre la funzione `studente_top_media` ritorna lo studente con la più alta media ponderata. Tutte le funzioni ritornano **l'indice** del database corrispondente.
Si può utilizzare la funzione `strcmp` della libreria `cstring`.

- `int cerca_matricola(Studente ** database, int matricola, int DIM);`
- `int cerca nome e cognome(Studente ** database, char * nome, char * cognome, int DIM);`
- `int studente_top_media(Studente ** database, int DIM);`

4 - Esse3++ (Completo)

Scrivere un programma che utilizzi tutte le funzioni definite nei precedenti esercizi per fornire delle funzionalità molto semplici di database per memorizzare i dati degli studenti. La dimensione del database è fissata a priori a 100. Il programma dovrà permettere all'utente di eseguire le seguenti azioni:

- Aggiungere un utente al sistema (se si raggiunge la dimensione massima notificare l'utente);
- Cercare un utente per matricola, nome/cognome e stamparlo a video (se l'utente non esiste, notificare l'utente);
- Ritornare l'utente con la media ponderata più alta;
- Caricare gli studenti da un file esterno.