**C++**

**Introduzione:**

Il progetto C++ descrive la gestione di una formazione di una squadra di calcio.

Il C++ è un linguaggio di programmazione orientato agli oggetti, con tipizzazione statica, ovvero il tipo di ogni variabile viene stabilito direttamente nel codice sorgente. E’stato sviluppato come miglioramento del linguaggio C.

**Funzionamento:**

La gestione di ogni classe è realizzata in file separati infatti ognuna è composta da un header file, nel quale vengono definiti i campi, il costruttore e i metodi della classe, e da un source file, dove gli elementi presenti vengono implementati.

La classe Formazione possiede come campo una lista che viene interrogata sfruttando i vector messi a disposizione dalla standard library. Il costruttore Formazione() inizializza la lista. I metodi messi a disposizione sono:

1. ~Formazione() : distruttore di Formazione che serve per deallocare la memoria;
2. aggiungi(Calciatore \*p) : permette di inserire nella lista (in coda) un elemento della classe Calciatore;
3. stampa() : stampa la lista di calciatori presenti sfruttando il metodo toString() della classe Calciatore;
4. etàMedia() : restituisce l’età media dei calciatori presenti nella lista;
5. stampanaz(string naz) : data una stringa in ingresso (nome di una nazione) stampa tutti i calciatori appartenenti a quella nazione;
6. piùPresenze() : stampa a video il giocatore che vanta più presenze.

I metodi definiti virtual permettono l’override degli stessi nelle sottoclassi ed il tipo di ereditarietà (public, private, protected) viene specificato nella definizione della classe nell’header file. Per quanto riguarda il costruttore bisogna estenderlo al costruttore della superclasse all’interno del source file.

|  |
| --- |
|  |
| *Fig. 1 Gerarchia delle classi.* |

La superclasse Calciatore possiede come campi il nome e cognome del calciatore, la nazionalità di appartenenza, l’età e il numero di partite giocate. Il costruttore è definito in modo tale da permettere all’utente di inserire i dati relativi ad un calciatore ed associarli ai campi del nuovo elemento che viene creato. I metodi della classe Calciatore sono:

1. getNome(), getCognome(), getNazione() : restituiscono una stringa che contiene rispettivamente nome, cognome e nazionalità del calciatore;
2. getPartitegiocate() : restituisce un intero che contiene il numero di partite giocate dal calciatore;
3. getEtà() : restituisce un float che contiene l’età del giocatore;
4. toString() : restituisce una stringa che contiene nome, cognome e nazionalità del calciatore;
5. ~Calciatore() : distruttore di Calciatore che serve per deallocare la memoria.

Poiché ogni calciatore possiede un ruolo è necessario creare una sottoclasse per ogni ruolo possibile. Ogni sottoclasse estende pubblicamente la classe Calciatore.

Nelle sottoclassi, oltre a possedere i campi e i metodi della superclasse, vengono aggiunti dei campi e dei metodi propri. La classe portiere ha un campo intero parate, un metodo setParate(int p) per modificare le parate realizzate dal portiere e un metodo calcolaParate() che ritorna un float che contiene la media delle parate per partite giocate oltre a ridefinire il distruttore e il metodo toString().

Lo stesso vale per le sottoclassi Difensore, Centrocampista e Attaccante con la differenza che il campo e i metodi aggiunti riguardano rispettivamente i contrasti riusciti, i passaggi riusciti e i goal realizzati.

Nella classe Demo vengono testate le varie classi con i rispettivi metodi.

|  |
| --- |
|  |
| *Fig. 2 Output* |