Relazione finale

Struttura del codice

Il codice è suddiviso in quattro classi:

1° classe Calciatore:

- Attributi (ciascuno di essi è pubblico ed ha i metodi get e set):
- `string nome_e_cognome;`
- `string ruolo;`
- `string squadra;`
- `int costo;`
- `int PunteggioPartita;`
- `bool gia_assegnato;`
- Metodi:
- `Calciatore()` è il costruttore della classe Calciatore;
- `List<Calciatore> GetCalciatori(string[] array);`
- `Int ControlloCalciatoreAsta(ref List<Calciatore> calciatori, string risposta);`

2°classe Giocatori:

- Attributi (ciascuno di essi è pubblico ed ha i metodi get e set):
- `string nome;`
- `int punteggio;`
- `int punteggio_totale;`
- `int crediti;`
- Liste(ciascuna di esse è public affinché possano essere modificate anche all'esterno della classe):
- `List<Calciatore> Rosa;`
- `List<Calciatore> Formazione;`
- Metodi:
- `Giocatori()` è il costruttore della classe Giocatori;
- `bool ControlloNomeInserito(string risposta, string[] nomigiocatori);`
- `int ControlloPrezzoInserito();`
- `int Asta(int[] prezzi);`
- `bool ControlloRose(ref List<Giocatori> giocatoris);`
- `void AssegnazioneRosa(ref List<Calciatore> calciatores, int indice_c,int prezzo);`
- `int ControlloGiocatoreFormazione(ref List<Calciatore> calciatores, string risposta);`
- `void CalcolaPunteggio();`

3°classe Partita:

- Metodi:
- `int[] GenerazionePartita(int quantita);`
- `int AssegnazionePunteggioCalciatore();`
- `void ClassificaFinale(ref List<Giocatori> giocatori);`
- `int GestionePartita(ref List<Giocatori> giocatores, int indice_1, int indice_2);`

4°classe Program:

- Metodi:
- `void Main(string[] args);`
- `static bool ControlloQuantità(string risposta);`

Funzionalità

Appena viene avviato il programma, vengono inizializzati:

- gli array string calciatorifile e nomigiocatori in cui verranno rispettivamente nel primo vengono inseriti i calciatori del file Calciatori.txt;
- la stringa filecalciatoripath in cui viene inserito il percorso del file Calciatori.txt;

In seguito controlla se il file, specificato il percorso filecalciatoripath, non esiste, se ciò si verifica allora crea il file Calciatori.txt e chiede all'utente di inserire i giocatori in esso.

Sennò assegna il contenuto di Calciatori.txt all'array calciatorifile, inizializza:

- la lista calciatori di tipo Calciatore;
- l'istanza c della classe Calciatore;
- l'istanza g della classe Giocatori;
- l'istanza p della classe Partita;

In seguito invoca il metodo GetCalciatori () della classe Calaciatore che ritorna una lista che viene assegnata a calciatori.

Dopo, tramite un for che si ripete per la quantità di calciatori presenti nella lista calciatori, scrive tutti i calciatori a schermo.

Appena avrà finito, l'utente può decidere il numero di giocatori del campionato che sarà compreso tra 2 e 10 e dovrà essere un numero pari.

Inizializza la variabile int quantità e gli assegna la quantità di giocatori che l'utente ha inserito;

In seguito, tramite ciclo for che si ripete fino a quando raggiunge quantità, fa inserire a ciascun giocatore il suo nome e lo controlla tramite la funzione ControlloNomeInserito della classe Giocatori che serve affinché i nomi che i giocatori inseriscono siano tutti diversi e non contengano spazi o virgole.

Inizializzo:

- la lista giocatori di tipo Giocatori e con un for assegno un nuovo elemento alla lista giocatori e inizializzo il nome ed i crediti del nuovo elemento;
- l'array prezzo in cui verranno contenuti i prezzi delle offerte per l'asta;
- la variabile int indice in cui viene assegnato l'indice del giocatore che ottiene il calciatore;
- la variabile int posizione per l'indice del calciatore dell'asta.

Con un ciclo do while, viene fatta l'asta.

L'utente scrive il nome del calciatore da mettere all'asta e dopo aver fatto ciò invoca il metodo ControlloCalciatoreAsta della classe Calciatore che controlla che il calciatore da mettere all'asta esista nella lista e non sia già stato comprato.

Dopo aver inserito il calciatore dell'asta, con un for ogni giocatore inserisce l'offerta per l'asta che può andare da 0 fino a 110 e se mette valori superiori, negativi oppure che superino il limite dei crediti del giocatore o per l'offerta, viene fatta reinserire l'offerta.

Appena i giocatori inseriscono l'offerta viene invocato il metodo Asta della classe Giocatore che ritorna l'indice del giocatore che ha ottenuto il calciatore messo all'asta e dopo al giocatore posto nel valore di ritorno viene assegnato il calciatore nella sua rosa tramite il metodo AssegnazioneRosa().

Potranno terminare l'asta appena tutti i giocatori avranno almeno 11 giocatori nella propria rosa.

Dopo aver terminato l'asta, viene fatta l'assegnazione delle formazioni dei giocatori che si basa sul modulo 4-3-3.

Per l'assegnazione delle formazioni c'è un primo for che si ripete per tutti i giocatori presenti nella lista giocatori ed il secondo per l'assegnazione dei calciatori nella formazione.

Ad ogni ciclo del primo for viene mostrato a schermo la rosa del giocatore e quindi i calciatori tra cui può scegliere per la formazione.

Dopo aver mostrato la rosa, inizia il secondo for, il quale si ripete per 11 volte che corrisponde al numero di giocatori presenti in una formazione, con il quale i giocatori scelgono il giocatore da inserire in base al ruolo specificato, per ogni volta che inseriscono un calciatore viene richiamato il metodo ControlloGiocatoreFormazione della classe Giocatori e controlla che il calciatore che il giocatore vuole inserire nella formazione esista nella rosa e che non sia già stato assegnato in qualche altro ruolo.

Dopo aver selezionato il calciatore da assegnare nella formazione, viene invocato il metodo AssegnazioneFormazione che assegna il calciatore nella lista formazione del giocatore.

Appena tutti i giocatori avranno terminato l'inserimento delle formazioni, inizia il campionato che, tramite ciclo do while, si ripete fino a quando l'utente decide di terminare il campionato.

Nel ciclo do while viene invocato il metodo GenerazionePartita della classe Partita che, in base al numero di giocatori della lista giocatori, ritorna un array di interi che viene assegnato ad indici.

Dopo aver generato gli indici dei giocatori che si sfidano nelle partite, con un primo ciclo for vengono fatte le 7 giornate e con un secondo for, all'interno del primo, vengono effettuati le singole partite.

Ad ogni ciclo del secondo for vengono scritti a schermo i giocatori che si sfidano nella partita e, dopo aver fatto ciò, viene invocato il metodo GestionePartita della classe Partita che genera risultati random per ciascun giocatore e ritorna un intero che corrisponde a chi ha vinto:

- 2 indica la vittoria del primo giocatore;
- 1 indica la vittoria del secondo giocatore;
- 0 indica il pareggio.

Appena termina il primo ciclo for viene invocato il metodo ClassificaFinale della classe Partita che riordina la lista giocatori in base a chi ha il punteggio totale maggiore e dopo viene scritto a schermo la classifica.

Dopo verrà chiesto all'utente se vuole continuare o meno il campionato, se l'utente scrive "yes" o "y" allora il campionato termina, sennò esso continua nel modo già spiegato prima.

Alla fine di tutto viene richiamato di nuovo il metodo ClassificaFinale e dopo scrive a schermo la classifica dei giocatori.

Migliorie:

- Aggiungere la possibilità di salvare partite su file in modo da poterle continuare;
- Aggiungere un malus nel caso in cui i giocatori inserisco un calciatore, nella formazione, in un ruolo che non sia il loro esempio (attaccante assegnato come portiere);
- Aggiungere la funzionalità del mercato libero, in cui ogni giocatore può prendere qualunque calciatore che può essere assegnato più di una volta;