**Analisi tecnica**

**PROGRAMMA GESTIONALE DEL FANTACALCIO**

**Premessa**  
Il programma avrà un’interfaccia grafica di tipo CLI (Command Line Interface), realizzata con .NET Core 3.1, attraverso l’uso del linguaggio di programmazione C#. Per realizzare il programma si userà l’ambiente di sviluppo Microsoft Visual Studio 2019. Di seguito verrà spiegato il funzionamento tecnico del programma, anche se questa non è una versione definitiva, quindi ci potranno essere dei cambiamenti più o meno importanti nel corso dello svolgimento del progetto.   
Per capire il funzionamento del Fantacalcio oppure i requisiti hardware e software del programma, fare riferimento all’analisi dei requisiti. Per vedere come funziona il software a livello utente, consultare l’analisi funzionale.

**Direttive *using***

Le principali direttive *using* che verranno utilizzate saranno molto probabilmente le seguenti:

* *using* **System**, direttiva che permette l’uso dei metodi più comuni;
* *using* **System.Threading.Tasks**, utilizzata per l’uso di metodi che posticipano le istruzioni;
* *using* **System.Linq**, che permette l’uso di specifici metodi per il confronto di stringhe;
* *using* **System.IO**, fondamentale per le operazioni sui file;
* *using* **System.Collections.Generic**, necessaria per la creazione di liste.

**Paradigma di programmazione**

Il codice verrà redatto seguendo la **OOP**, ovvero programmazione orientata agli oggetti, che prevede la presenza di:

* ***classi***, che hanno il compito di definire le caratteristiche di un insieme di oggetti che hanno proprietà e compiono azioni uguali;
* ***attributi***, cioè delle variabili e/o costanti che definiscono le caratteristiche o proprietà degli oggetti instanziabili invocando la classe; i valori inizializzati degli attributi sono ottenuti attraverso il cosiddetto costruttore;
* ***metodi***, ossia procedure che operano sugli attributi.

Gli oggetti, quindi, sono istanze delle classi.

**Classi**

Le classi che compongono il codice del programma, insieme ai loro metodi e attributi, sono le seguenti.

* Classe **Programma**: questa classe viene eseguita automaticamente all’avvio del programma.

Il principale attributo della classe è:

* + **comandoInserito** (string), che contiene il comando inserito dall’utente in quei casi in cui non si stiano inserendo dei particolari dati necessari per l’impostazione del torneo di Fantacalcio.

I metodi della classe, invece, sono:

* **Main**, il metodo a cui il programma fa per primo riferimento in seguito al suo avvio, che contiene fondamentalmente la visualizzazione a schermo dei due casi possibili della schermata iniziale;
* **SchermataInserimentoRose**, che contiene la parte di visualizzazione a schermo della schermata di inserimento delle rose di fanta-giocatori;
* **SchermataVisualizzazioneFantaCalciatori**, che contiene la parte di visualizzazione a schermo della schermata di ricerca dei fanta-giocatori;
* **SchermataSchieramentoCampoFantaCalciatori**, che contiene la parte di visualizzazione a schermo di schieramento in campo dei fanta-giocatori;
* **SchermataAggiornamentoStatisticheFantaCalciatori**, che contiene la parte di visualizzazione a schermo della schermata di aggiornamento delle statistiche dei fanta-giocatori;
* **SchermataVisualizzazioneClassifiche**, che contiene la parte di visualizzazione a schermo della schermata delle classifiche parziali dei fanta-giocatori;
* **SchermataCancellazioneDati**, che contiene la parte di visualizzazione a schermo della schermata di cancellazione dei dati;
* **SchermataComandi**, che contiene la parte di visualizzazione a schermo della schermata dei comandi;
* **UscitaProgramma**, che contiene i metodi per uscire dal programma, tra cui **Environment.Exit()**;
* Classe **Files**: questa classe contiene tutti i metodi necessari per eseguire la lettura e scrittura dei files.   
  Gli attributi della classe sono:
  + **inputOutput** (string), che contiene informazioni su ciò che deve essere letto (input) oppure stampato (output);
  + **percorsoIO** (string), che contiene il percorso assoluto di un file che deve essere letto (input) oppure scritto (output).

I metodi della classe, invece, sono:

* Metodo **costruttore**, che permette l’effettivo passaggio del valore degli attributi di altre classi agli attributi della classe corrente;
* **VerificaEsistenzaFile**, metodo che permette di verificare se esiste o meno il percorso del file che gli è stato passato attraverso **File.Exists()**;
* **CreaCartella**, metodo che viene richiamato quando né i file né la cartella di salvataggio sono presenti; contiene il metodo **Directory.CreateDirectory()**, le cui eccezioni verranno gestite attraverso un **blocco try-catch**;
* **CreaFile**, metodo richiamato successivamente a **CreaCartella**, che permette di creare i file attraverso il metodo **File.Create()**, le cui eccezioni verranno gestite attraverso un **blocco try-catch**;
* **LeggiFile**, metodo che permette, attraverso **File.ReadAllLines()**, la lettura e l’inserimento all’interno delle liste di fanta-calciatori, così come delle rose e degli schieramenti, di ciò che si trova nei files di salvataggio;
* **ScriviFile**, metodo che permette di scrivere su di un file di testo attraverso **File.WriteAllLines()**, le cui eccezioni verranno gestite attraverso un **blocco try-catch**;
* **GestisciErrori**, metodo richiamato nel caso di eccezioni verificatasi nei metodi precedentemente spiegati, che contiene i codici di errore e quale tipo di messaggi da visualizzare nelle varie situazioni.
* Classe **FantaAllenatore**: questa classe permette di accedere ai metodi che riguardano i fanta-allenatori.   
  Gli attributi della classe sono:
  + **torneo** (string), ossia il tipo di torneo reale di riferimento per il Fantacalcio scelto dai fanta-allenatori;
  + **nome** (string), ovvero il nome del fanta-allenatore;
  + **codiceRosa** (string), che indica il numero di rosa che appartiene al fanta-allenatore;
  + **budgetDisponibile** (integer), che indica il numero di Fantamilioni che sono rimasti al fanta-allenatore;
  + **FantaCalciatori** (lista string), che contiene le caratteristiche di ogni fanta-calciatore;
  + **RoseFantaCalciatori** (vettore integer), che contiene gli indici di posizione dei giocatori che si trovano all’interno della lista **FantaCalciatori** e che fanno parte della rosa del fanta-allenatore;
  + **SchieramentiFantaCalciatori** (vettore integer), che contiene gli indici di posizione dei giocatori che si trovano all’interno della lista **FantaCalciatori** e che fanno parte dello schieramento del fanta-allenatore.

I metodi della classe, invece, sono:

* Metodo **costruttore**, che permette l’effettivo passaggio del valore degli attributi di altre classi agli attributi della classe corrente;
* **AggiungiFantaAllenatore**, che permette di aggiungere un fanta-allenatore;
* **AggiungiRosa**, che permette ad ogni fanta-allenatore di creare la propria rosa di fanta-giocatori;
* **AggiungiSchieramento**, che permette ad ogni fanta-allenatore di creare il proprio schieramento giornaliero di fanta-giocatori;
* Classe **FantaCalciatore**: questa classe permette di accedere ai metodi che riguardano i fanta-calciatori. Gli attributi della classe sono:
  + **nome** (string), ovvero il nome del fanta-calciatore;
  + **cognome** (string), ovvero il cognome del fanta-calciatore;
  + **squadra** (string), ovvero la squadra di appartenenza del fanta-calciatore;
  + **ruolo** (string) , ovvero il ruolo fanta-calciatore all’interno della squadra;
  + **numeroMaglia** (integer), ovvero il numero di maglia del fanta-calciatore;
  + **quotazioneIniziale** (integer), ovvero la quotazione iniziale del fanta-calciatore;
  + quotazioneAttuale (integer), ovvero la quotazione attuale del fanta-calciatore;
  + **punteggioClassifica** (integer), ovvero il punteggio del fanta-calciatore nella classifica parziale;
  + **FantaCalciatori** (lista string), che contiene le caratteristiche di ogni fanta-calciatore.

I metodi della classe, invece, sono:

* Metodo **costruttore**, che permette l’effettivo passaggio del valore degli attributi di altre classi agli attributi della classe corrente;
* **AggiungiFantaCalciatore**, metodo che permette di aggiungere un giocatore alle liste di fanta-calciatori, utilizzando specifici metodi di inserimento degli oggetti **List**;
* **RicercaFantaCalciatore**, che permette di ricercare uno o più fanta-giocatori in base ai filtri inseriti dall’utente, utilizzando specifici metodi di ricerca degli oggetti **List**;
* **AggiornaStatisticheFantaCalciatore**, che aggiorna le quotazioni e i punteggi di ogni fanta-calciatore, utilizzando specifici metodi di inserimento degli oggetti **List**;
* **OrdinaFantaCalciatori**, che ordina in modo decrescente i giocatori in base al loro punteggio, utilizzando specifici metodi di ordinamento degli oggetti **List**.

Per concludere, in generale:

* ogni classe può avere degli attributi in più, o in meno, oppure possono essere leggermente differenti rispetto a quelli elencati fino ad ora;
* ogni classe, tranne Programma, contiene il metodo di tipo ovverride (un metodo cioè che ne sovrascrive un altro già esistente) **ToString**, e che ritorna il valore di una stringa costruita da più stringhe, pronta per essere visualizzata a schermo.
* ogni metodo, presenta delle variabili interne necessarie ai fini del funzionamento del programma, per esempio indici o contatori, che non possono essere completamente previsti e, di conseguenza, elencati all’interno di questa analisi.