

# Relazione del Progetto di Raccomandazione Musicale

Giaconi Christian, Giacomo Rossi  
Matricola: 314045, 314671

6 gennaio 2025

# Indice

<b>1</b>	<b>Specifica del problema</b>	<b>3</b>
<b>2</b>	<b>Analisi del problema</b>	<b>4</b>
2.1	Dati in ingresso . . . . .	4
2.2	Dati in uscita . . . . .	4
2.3	Relazioni tra i dati . . . . .	4
<b>3</b>	<b>Progettazione dell'algoritmo</b>	<b>5</b>
3.1	Scelte di progetto . . . . .	5
3.2	Passi dell'algoritmo . . . . .	5
<b>4</b>	<b>Implementazione dell'algoritmo</b>	<b>6</b>
4.1	Implementazione in Haskell . . . . .	6
4.2	Implementazione in Prolog . . . . .	6
<b>5</b>	<b>Testing</b>	<b>7</b>
5.1	Esempi di input e output . . . . .	7
5.2	Conclusioni . . . . .	7

# 1 Specifica del problema

Il progetto consiste nell'implementazione di un sistema di raccomandazione musicale in Haskell e Prolog. Obiettivo principale è suggerire all'utente una classifica di canzoni in base alle sue preferenze di genere musicale e pesi assegnati. I passi principali includono:

- Caricamento delle canzoni da un file di testo strutturato.
- Specifica delle preferenze dell'utente per i generi musicali.
- Calcolo dei punteggi ponderati e ordinamento delle canzoni.
- Modifica dinamica dei pesi in Prolog.

## 2 Analisi del problema

### 2.1 Dati in ingresso

- Un file contenente le canzoni nel formato:

`Titolo,Artista,Genere,Punteggio`

- Una lista di generi preferiti.
- Pesi numerici assegnati ai generi preferiti.

### 2.2 Dati in uscita

- Una classifica ordinata di canzoni basata sui punteggi ponderati.
- Eventuali messaggi di errore o conferma nelle interazioni utente.

### 2.3 Relazioni tra i dati

Ogni canzone è caratterizzata da un titolo, un artista, un genere, e un punteggio. Il punteggio ponderato è calcolato come:

$$Punteggio\ Ponderato = Punteggio \times Peso\_Genere$$

se il genere della canzone appartiene ai generi preferiti, altrimenti è uguale al punteggio originale.

## **3 Progettazione dell'algoritmo**

### **3.1 Scelte di progetto**

- In Haskell, le canzoni sono modellate come tipi di dati strutturati per una manipolazione chiara e leggibile.
- In Prolog, si utilizzano predicati dinamici per rappresentare canzoni, generi preferiti e pesi.

### **3.2 Passi dell'algoritmo**

1. Caricare le canzoni da un file.
2. Inserire le preferenze dell'utente per i generi e i pesi.
3. Calcolare i punteggi ponderati.
4. Ordinare le canzoni per punteggio ponderato.
5. Stampare la classifica.

## 4 Implementazione dell'algoritmo

### 4.1 Implementazione in Haskell

Il file `raccomandazioni.hs` implementa l'algoritmo in Haskell. Un esempio di calcolo dei punteggi ponderati:

```
1 arricchisci :: [String] -> Double -> [Canzone] -> [(Double, Canzone)]
2 arricchisci _ _ [] = []
3 arricchisci generiPreferiti peso (c:cs) =
4     let genereMinuscolo = map toLower (genere c)
5         punteggioPonderato = if genereMinuscolo `elem` generiPreferiti
6                               then fromIntegral (punteggio c) * peso
7                               else fromIntegral (punteggio c)
8     in (punteggioPonderato, c) : arricchisci generiPreferiti peso cs
```

Listing 1: Calcolo dei punteggi ponderati

### 4.2 Implementazione in Prolog

Il file `raccomandazioni.pl` implementa l'algoritmo in Prolog. Esempio di ordinamento:

```
1 classifica_ordinata(Ordinata) :-
2     findall(Punteggio-Titolo, punteggio_ponderato(Titolo, Punteggio),
3             Punteggi),
4     sort(1, @>=, Punteggi, Ordinata).
```

Listing 2: Ordinamento delle canzoni

## 5 Testing

### 5.1 Esempi di input e output

Input di esempio:

Despacito,Luis Fonsi,Reggaeton,8

Shape of You,Ed Sheeran,Pop,9

Havana,Camila Cabello,Pop,10

Preferenze utente:

- Generi preferiti: Pop.
- Peso assegnato: 1.5.

Output:

1. Havana (Punteggio ponderato: 15.0)
2. Shape of You (Punteggio ponderato: 13.5)
3. Despacito (Punteggio ponderato: 8.0)

### 5.2 Conclusioni

Entrambi i programmi raggiungono l'obiettivo, ma differiscono per efficienza e flessibilità:

- Haskell offre un'elaborazione dati chiara e performante.
- Prolog consente una gestione dinamica dei pesi durante l'esecuzione.

Estensioni future potrebbero includere l'integrazione con database musicali e filtri avanzati.