Simulazione Esame di Programmazione

Roberto Borelli

23 Gennaio 2024

Descrizione generale

Si richiede di implementare un programma per agevolare la correzzione di verifiche di matematica. Una verifica di matematica è composta da 8 esercizi, ognuno dei quali può essere di tre tipi, un'**eq**uazione, una **pr**oporzione oppure un **es**pressione da risolvere. Per semplicità si assuma che:

- Le equazioni siano del tipo `eq_a eq_x = eq_b`, dove `eq_a` e `eq_b` sono dei numeri e `eq_x` è un nome di variabile (una sola lettera). Ad esempio `4 y = 2` è una valida equazione e la sua soluzione è `y = 0.5`.
- Le proporzioni siano del tipo `pr_a1 : pr_a2 = pr_x : pr_a3` dove `pr_a1, pr_a2` e `pr_a3` sono numeri e `pr_x` è un nome di variabile (una sola lettera). Ad esempio `10 : 20 = c : 200` è una valida proporzione con soluzione `c = 100`.
- Le espressioni siano del tipo `es_a1 es_op es_a2` dove `es_a1` e `es_a2` sono due numeri, mentre `es_op` è un simbolo tra `+, -, /, *`. Ad esempio `4 * 4` è una valida espressione e il suo risultato è `16`.

La verifica è memorizzata nel file `exam.txt` che contiene ad esempio il seguente testo.

```
eq [3] 4 y = 40
eq [3] 5 z = 10
pr [1] 10 : 20 = c : 200
es [0.5] 30 * 20
es [0.5] 4 * 2
es [0.5] 6 / 6
pr [1] 15 : 150 = c : 150
es [0.5] 5 + 2
```

Ogni riga inizia con il tipo di esercizio (`eq`, `es` o `pr`). Tra parentesi quadre viene specificato il numero di punti dell'esercizio. Segue poi il testo del rispettivo esercizio.

Per memorizzare un esercizio bisognerà definire il tipo di dato `esercizio` che avrà i campi `tipo` (che potrà avere valori `eq`, `es` o `pr`) `punteggio`, e i campi `eq_a`, `eq_x`, `eq_b`, `pr_a1`, `pr_a2`, `pr_x`, `pr_a3`, `es_a1`, `es_op`, `es_a2` e `risultato`. I campi che hanno prefisso `eq` saranno utilizzati solo per le equazioni e analogamente i campi che iniziano con `pr` e `es` saranno utilizzati rispettivamente solo per le proporzioni e per le espressioni.

1 Leggere il file (10 pt)

Implementare una funzione `leggi_esame` che riceve in input il puntatore al file `exam.txt`. La funzione legge il file e riempie opportunamente un array (lungo 8) di tipo `esercizio` (che sarà un parametro di uscita). Se nel file, il numero di esercizi è diverso da 8, verrà stampato un opportuno errore. Si osservi che questa funzione non riempie il campo `risultato`.

2 Calcolare i risultati (5 pt)

Implementare una funzione `calcola_risultati` che prende in input un array (lungo 8) di tipo `esercizio` e per ciascun esercizio riempia opportunamente il campo `risultato`.

3 Stampare i risultati (5 pt)

Implementare una funzione `stampa_risultati` che prende in input un array (lungo 8) di tipo `esercizio` in cui si suppone che il campo `risultato` sia stato già calcolato. La funzione crea il file `sol.txt` in cui stampa i risultati opportunamente. Con il file di esame mostrato in precedenza, il file di risultati dovrà essere:

```
y = 10
z = 2
c = 100
600
8
1
c = 15
7
```

4 Calcolare punteggio (7 pt)

Creare una funzione `judge` che prende in input un array lungo 8 di tipo `esercizio` e il puntatore ad un file contenente delle risposte. La funzione controlla quali righe del file sono uguali alle righe del file `sol.txt` prodotto in precedenza, e in base a questo stampa il voto. Ad esempio se alla funzione viene passato in input il puntatore al file `sol_studente1.txt` che contiene le seguenti righe

```
z = 10
z = 2
c = 100
700
8
1
c = 14
7
```

Il voto sarà $10 - (3 + 0.5 + 1) = 5.5$.

5 Main (7 pt)

Creare un main che permetta di leggere il file `exam.txt` e crerae il file `sol.txt` tramite le funzioni degli esercizi precedenti. In seguito il main chiederà all'utente di inserire il nome di un file di soluzione. Chiamando l'opportuna funzione, il file verrà a aperto analizzato e sarà stamapato il punteggio. Dopo la chiusura del file il main continuerà chiedendo un altro nome di file e il processo si ripete.

6 Facoltativo

Modificare la funzione `judge` in modo che stampi quali esercizi sono stati sbagliati in un compito, le relative correzioni, e qual'è stato il punteggio rimosso per un esercizio sbagliato. Ad esempio, considerando sempre il file `sol_studente1.txt`, l'output dovrebbe essere il seguente.

```
[-3]   ERR z = 10 (doveva essere y = 10)
       OK  z = 2
       OK  c = 100
[-0.5] ERR 700    (doveva essere 600)
       OK  8
       OK  1
[-1]   ERR c = 14 (doveva essere c = 15)
       OK  7
---------------------------------------
voto = 5.5 su 10
```