

Esame 20250221

Esercizio 3

(1) Esercizio 3 v1

In molti contesti è necessario verificare che ad ogni parentesi aperta (sia essa il carattere '(', '[' o '{') corrisponda una parentesi chiusa dello stesso tipo (sia essa il carattere ')', ']' o '}') rispettivamente). A tale scopo il programma di questo esercizio prende come argomento un nome di un file, lo apre e chiama una funzione `check` che prende come argomento uno stream di input, legge il contenuto del file e restituisce un booleano.

La funzione `check` deve usare per calcolare il valore di ritorno o uno stack o una coda entrambi di caratteri (`char`).

In particolare, la funzione deve restituire `true` se e solo se il file contiene una sequenza di parentesi ben bilanciate, altrimenti deve restituire `false`.

Sono già fornite le implementazione dello stack e della coda, che possono essere utilizzate per implementare la funzione `check`. (Si vedano i file `stack.h`, `stack.cpp`, `queue.h` e `queue.cpp`).

Il file `esercizio3.cpp` contiene tutto quanto necessario tranne la dichiarazione e la definizione della funzione `check`.

Di seguito sono riportati alcuni esempi di esecuzione del programma.

```
computer > ./a.out input1.txt
The function check returns true
computer > ./a.out input2.txt
The function check returns false
computer > ./a.out input3.txt
The function check returns false
computer > ./a.out input4.txt
The function check returns true
```

Note:

- Scaricare il file `esercizio3.cpp`, modificarlo per inserire il codice necessario per rispondere a questo esercizio. **Caricare il file sorgente risultato delle vostre modifiche a soluzione di questo esercizio** nello spazio apposito.
- All'interno di questo programma **non è ammesso** l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `iostream`, `fstream`, `cstdlib`.
- Si ricorda che, gli esempi di esecuzione sono puramente indicativi, e la soluzione proposta NON deve funzionare solo per l'input fornito, ma deve essere robusta a variazioni compatibili con la specifica riportata in questo testo.
- Si ricorda di inserire solo nuovo codice e di **NON MODIFICARE** il resto del programma (pena annullamento dell'esercizio).
- Si ricorda che tutta la memoria allocata dinamicamente deve essere deallocata.

`esercizio3.cpp`

`stack.cpp` `stack.h`

`queue.cpp` `queue.h`

`input1.txt` `input2.txt` `input3.txt` `input4.txt`

Information for graders:

(2) Esercizio 3 v2

ESSAY marked out of 10 penalty 0 File picker

In molti contesti è necessario verificare che ad ogni parentesi aperta (sia essa il carattere '(', '[' o '{') corrisponda una parentesi chiusa dello stesso tipo (sia essa il carattere ')', ']' o '}') rispettivamente). A tale scopo il programma di questo esercizio prende come argomento un nome di un file, lo apre e chiama una funzione `check` che prende come argomento uno stream di input, legge il contenuto del file e restituisce un booleano.

La funzione `check` deve usare per calcolare il valore di ritorno o uno stack o una coda entrambi di caratteri (`char`).

In particolare, la funzione deve restituire `false` se e solo se il file contiene una sequenza di parentesi ben bilanciate, altrimenti deve restituire `true`.

Sono già fornite le implementazione dello stack e della coda, che possono essere utilizzate per implementare la funzione `check`. (Si vedano i file `stack.h`, `stack.cpp`, `queue.h` e `queue.cpp`).

Il file `esercizio3.cpp` contiene tutto quanto necessario tranne la dichiarazione e la definizione della funzione `check`.

Di seguito sono riportati alcuni esempi di esecuzione del programma.

```
computer > ./a.out input1.txt
The function check returns false
computer > ./a.out input2.txt
The function check returns true
computer > ./a.out input3.txt
The function check returns true
computer > ./a.out input4.txt
The function check returns false
```

Note:

- Scaricare il file `esercizio3.cpp`, modificarlo per inserire il codice necessario per rispondere a questo esercizio. **Caricare il file sorgente risultato delle vostre modifiche a soluzione di questo esercizio** nello spazio apposito.
- All'interno di questo programma **non è ammesso** l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `iostream`, `fstream`, `cstdlib`.
- Si ricorda che, gli esempi di esecuzione sono puramente indicativi, e la soluzione proposta NON deve funzionare solo per l'input fornito, ma deve essere robusta a variazioni compatibili con la specifica riportata in questo testo.
- Si ricorda di inserire solo nuovo codice e di **NON MODIFICARE** il resto del programma (pena annullamento dell'esercizio).
- Si ricorda che tutta la memoria allocata dinamicamente deve essere deallocata.

`esercizio3.cpp`

`stack.cpp` `stack.h`

`queue.cpp` `queue.h`

`input1.txt` `input2.txt` `input3.txt` `input4.txt`

Information for graders:

Total of marks: 20