Esame 20210907

Esercizio 2

(1) **Esercizio 2 v1**

ESSAY | marked out of 10 | penalty 0 | File picker

Scrivere la dichiarazione e la definizione di una funzione **ricorsiva** `extract`, che prende come argomento una stringa (letta da tastiera) di lunghezza non nota e che non può essere modificata (i.e. `const`), ma ben formata (i.e. con carattere di terminatore `'\0'`), ed estrae tutti i caratteri `'@'` e restituisce una **nuova** stringa di dimensione uguale al numero di caratteri estratti (ovviamente più uno per il carattere di terminazione). La nuova stringa deve essere allocata dinamicamente, e deve avere appunto la minima dimensione necessaria per memorizzare i caratteri estratti. Se non vi sono caratteri da estrarre ritorna la stringa vuota!

La funzione è inserita in un semplice programma che legge una stringa da tastiera, stampa la stringa estratta, e chiede se procedere con una nuova stringa o se terminare (il `main` non deve essere modificato). Alcuni esempi di esecuzione sono i seguenti:

```
computer > ./a.out
Inserire la stringa da controllare: caro@prof@che@esercizio@difficile
La stringa estratta e': @@@@
Si vuole inserire un'altra stringa? [*/n]s
Inserire la stringa da controllare: caroprofcheeserciziodifficile
La stringa estratta e':
Si vuole inserire un'altra stringa? [*/n]n
```

Note:

- Scaricare il file `esercizio2.cc`, modificarlo per inserire la dichiarazione e la definizione della funzione `extract`, e caricare il file sorgente risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito.
- La funzione `extract` deve essere ricorsiva ed al suo interno **NON ci possono essere cicli o chiamate a funzioni contenenti cicli**. Si può però fare uso di funzioni ausiliarie da chiamare all'interno di questa funzione che NON contengano cicli o che siano ricorsive.
- Le uniche assunzioni che si possono fare sull'input e su dimensioni di eventuali strutture/array utilizzate nel file di partenza fornito sono **solo quelle espressamente specificate in questo testo** (e NON quelle riportate nel file fornito, che sono SOLO indicative per consentire di svolgere l'esame).
- 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 `cstddef`.
- Si ricorda che, l'esempio di esecuzione è puramente indicativo, 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.

Information for graders:

(2) **Esercizio 2 v2**

ESSAY | marked out of 10 | penalty 0 | File picker

Scrivere la dichiarazione e la definizione di una funzione **ricorsiva** `extract`, che prende come argomento una stringa (letta da tastiera) di lunghezza non nota e che non può essere modificata (i.e. `const`), ma ben formata (i.e. con carattere di terminatore `'\0'`), ed estrae tutti i caratteri diversi da `'@'` e restituisce una **nuova** stringa di dimensione uguale al numero di caratteri estratti (ovviamente più uno per il carattere di terminazione). La nuova stringa deve essere allocata dinamicamente, e deve avere appunto la minima dimensione necessaria per memorizzare i caratteri estratti. Se non vi sono caratteri da estrarre ritorna la stringa vuota!

La funzione è inserita in un semplice programma che legge una stringa da tastiera, stampa la stringa estratta, e chiede se procedere con una nuova stringa o se terminare (il `main` non deve essere modificato). Alcuni esempi di esecuzione sono i seguenti:

```
computer > ./a.out
Inserire la stringa da controllare: caro@prof@che@esercizio@difficile
La stringa estratta e': caroprofcheesercizziodifficile
Si vuole inserire un'altra stringa? [*/n]s
Inserire la stringa da controllare: @@@@@@@
La stringa estratta e':
Si vuole inserire un'altra stringa? [*/n]n
```

Note:

- Scaricare il file `esercizio2.cc`, modificarlo per inserire la dichiarazione e la definizione della funzione `extract`, e caricare il file sorgente risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito.
- La funzione `extract` deve essere ricorsiva ed al suo interno **NON ci possono essere cicli o chiamate a funzioni contenenti cicli**. Si può però fare uso di funzioni ausiliarie da chiamare all'interno di questa funzione che NON contengano cicli o che siano ricorsive.
- Le uniche assunzioni che si possono fare sull'input e su dimensioni di eventuali strutture/array utilizzate nel file di partenza fornito sono **solo quelle espressamente specificate in questo testo** (e NON quelle riportate nel file fornito, che sono SOLO indicative per consentire di svolgere l'esame).
- 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 `cstddef`.
- Si ricorda che, l'esempio di esecuzione è puramente indicativo, 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.

Information for graders:

(3) **Esercizio 2 v3**

ESSAY | marked out of 10 | penalty 0 | File picker

Scrivere la dichiarazione e la definizione di una funzione **ricorsiva** `extract`, che prende come argomento una stringa (letta da tastiera) di lunghezza non nota e che non può essere modificata (i.e. `const`), ma ben formata (i.e. con carattere di terminatore `'\0'`), ed estrae tutti i caratteri diversi da `'$'` e restituisce una **nuova** stringa di dimensione uguale al numero di caratteri estratti (ovviamente più uno per il carattere di terminazione). La nuova stringa deve essere allocata dinamicamente, e deve avere appunto la minima dimensione necessaria per memorizzare i caratteri estratti. Se non vi sono caratteri da estrarre ritorna la stringa vuota!

La funzione è inserita in un semplice programma che legge una stringa da tastiera, stampa la stringa estratta, e chiede se procedere con una nuova stringa o se terminare (il `main` non deve essere modificato). Alcuni esempi di esecuzione sono i seguenti:

```
computer > ./a.out
Inserire la stringa da controllare: caro$prof$che$esercizio$difficile
La stringa estratta e': caroprofcheesercizиodifficile
Si vuole inserire un'altra stringa? [*/n]s
Inserire la stringa da controllare: $$$$$
La stringa estratta e':
Si vuole inserire un'altra stringa? [*/n]n
```

Note:

- Scaricare il file `esercizio2.cc`, modificarlo per inserire la dichiarazione e la definizione della funzione `extract`, e caricare il file sorgente risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito.
- La funzione `extract` deve essere ricorsiva ed al suo interno **NON ci possono essere cicli o chiamate a funzioni contenenti cicli**. Si può però fare uso di funzioni ausiliarie da chiamare all'interno di questa funzione che NON contengano cicli o che siano ricorsive.
- Le uniche assunzioni che si possono fare sull'input e su dimensioni di eventuali strutture/array utilizzate nel file di partenza fornito sono **solo quelle espressamente specificate in questo testo** (e NON quelle riportate nel file fornito, che sono SOLO indicative per consentire di svolgere l'esame).
- 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 `cstddef`.
- Si ricorda che, l'esempio di esecuzione è puramente indicativo, 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.

Information for graders:

(4) **Esercizio 2 v4**

ESSAY | marked out of 10 | penalty 0 | File picker

Scrivere la dichiarazione e la definizione di una funzione **ricorsiva** `extract`, che prende come argomento una stringa (letta da tastiera) di lunghezza non nota e che non può essere modificata (i.e. `const`), ma ben formata (i.e. con carattere di terminatore `'\0'`), ed estrae tutti i caratteri `'$'` e restituisce una **nuova** stringa di dimensione uguale al numero di caratteri estratti (ovviamente più uno per il carattere di terminazione). La nuova stringa deve essere allocata dinamicamente, e deve avere appunto la minima dimensione necessaria per memorizzare i caratteri estratti. Se non vi sono caratteri da estrarre ritorna la stringa vuota!

La funzione è inserita in un semplice programma che legge una stringa da tastiera, stampa la stringa estratta, e chiede se procedere con una nuova stringa o se terminare (il `main` non deve essere modificato). Alcuni esempi di esecuzione sono i seguenti:

```
computer > ./a.out
Inserire la stringa da controllare: caro$prof$che$esercizio$difficile
La stringa estratta e': $$$$
Si vuole inserire un'altra stringa? [*/n]s
Inserire la stringa da controllare: caroprofcheesercizиodifficile
La stringa estratta e':
Si vuole inserire un'altra stringa? [*/n]n
```

Note:

- Scaricare il file `esercizio2.cc`, modificarlo per inserire la dichiarazione e la definizione della funzione `extract`, e caricare il file sorgente risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito.
- La funzione `extract` deve essere ricorsiva ed al suo interno **NON ci possono essere cicli o chiamate a funzioni contenenti cicli**. Si può però fare uso di funzioni ausiliarie da chiamare all'interno di questa funzione che NON contengano cicli o che siano ricorsive.
- Le uniche assunzioni che si possono fare sull'input e su dimensioni di eventuali strutture/array utilizzate nel file di partenza fornito sono **solo quelle espressamente specificate in questo testo** (e NON quelle riportate nel file fornito, che sono SOLO indicative per consentire di svolgere l'esame).
- 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 `cstddef`.
- Si ricorda che, l'esempio di esecuzione è puramente indicativo, 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.

Information for graders:

Total of marks: 40