Esame 20220617

Esercizio 1

(1) **Esercizio 1 v1**

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

Sia `input.txt` un file di input contenente: un valore intero ed una serie di stringhe separate da spazi o caratteri nuova riga (`\n`). Queste stringhe possono essere ripetute molteplici volte all'interno del file. Il valore intero indica quante sono le stringhe presenti nel file. Sia `order.txt` un file in input contenente dei numeri interi separati da spazi. I numeri interi possono essere sia positivi che negativi e ci possono essere dei duplicati. Ogni numero **puó** corrispondere alla posizione di una stringa nel primo file di input.

Scrivere nel file `esercizio1.cc` un programma che, dati come argomenti da riga di comando il file di input, il file con gli interi e il nome di un file di output, scriva sul file in output le stringhe specificate dagli interi del secondo file. Nel caso uno degli interi non corrisponda a nessuna stringa del file, si dovrá scrivere `MISSING`. Il programma deve anche controllare l'accuratezza dell'invocazione (e.g., numero di argomenti corretti) e la corretta apertura degli stream di input e output.

Supponiamo che il primo file `input.txt` contenga

```
14
You make a ton of progress by making a ton of mistakes Astro Teller
```

e che il secondo file in input contenga

```
3 1 52 1 6 -3
```

Il programma dovrá poter essere chiamato nel seguente modo:

```
./esercizio1.out input.txt order.txt output.txt
```

e dovrá produrre un file chiamato `output.txt` che conterrá i seguenti valori:

```
ton make MISSING make by MISSING
```

Note:

- Si assuma che la numerazione degli interi del secondo file inizi da 0.
- Si assuma inoltre che ogni stringa sia formata da al massimo 100 caratteri. Non é concesso fare assunzioni sul numero **massimo** di stringhe presenti nel file.
- E' consentito l'utilizzo della funzioni `int atoi (const char * str)` e `char * strcpy (char * destination, const char * source)` della libreria `<cstring>`.
- Non é consentito l'utilizzo di altre funzioni di libreria "particolari" diverse da quelle specificate sopra o da quelle standard necessarie a risolvere l'esercizio.
- 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`.
- 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 1 v2**

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

Sia `input.txt` un file di input contenente: un valore intero ed una serie di stringhe separate da spazi o caratteri nuova riga (`\n`). Queste stringhe possono essere ripetute molteplici volte all'interno del file. Il valore intero indica quante sono le stringhe presenti nel file. Sia `order.txt` un file in input contenente dei numeri interi separati da spazi. I numeri interi possono essere sia positivi che negativi e ci possono essere dei duplicati. Ogni numero **puó** corrispondere alla posizione di una stringa nel primo file di input.

Scrivere nel file `esercizio1.cc` un programma che, dati come argomenti da riga di comando il file di input, il file con gli interi e il nome di un file di output, scriva sul file in output le stringhe specificate dagli interi del secondo file. Nel caso uno degli interi non corrisponda a nessuna stringa del file, si dovrá scrivere `MANCANTE`. Il programma deve anche controllare l'accuratezza dell'invocazione (e.g., numero di argomenti corretti) e la corretta apertura degli stream di input e output.

Supponiamo che il primo file `input.txt` contenga

```
14
You may think using Google's great,
but I still think it's terrible. Larry Page
```

e che il secondo file in input contenga

```
3 1 52 1 6 -3
```

Il programma dovrá poter essere chiamato nel seguente modo:

```
./esercizio1.out input.txt order.txt output.txt
```

e dovrá produrre un file chiamato `output.txt` che conterrá i seguenti valori:

```
using may MANCANTE may but MANCANTE
```

Note:

- Si assuma che la numerazione degli interi del secondo file inizi da 0.
- Si assuma inoltre che ogni stringa sia formata da al massimo 100 caratteri. Non é concesso fare assunzioni sul numero **massimo** di stringhe presenti nel file.
- E' consentito l'utilizzo della funzioni `int atoi (const char * str)` e `char * strcpy (char * destination, const char * source)` della libreria `<cstring>`.
- Non é consentito l'utilizzo di altre funzioni di libreria "particolari" diverse da quelle specificate sopra o da quelle standard necessarie a risolvere l'esercizio.
- 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`.

- 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 1 v3**

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

Sia `input.txt` un file di input contenente: un valore intero ed una serie di stringhe separate da spazi o caratteri nuova riga (`\n`). Queste stringhe possono essere ripetute molteplici volte all'interno del file. Il valore intero indica quante sono le stringhe presenti nel file. Sia `order.txt` un file in input contenente dei numeri interi separati da spazi. I numeri interi possono essere sia positivi che negativi e ci possono essere dei duplicati. Ogni numero **puó** corrispondere alla posizione di una stringa nel primo file di input.

Scrivere nel file `esercizio1.cc` un programma che, dati come argomenti da riga di comando il file di input, il file con gli interi e il nome di un file di output, scriva sul file in output le stringhe specificate dagli interi del secondo file. Nel caso uno degli interi non corrisponda a nessuna stringa del file, si dovrá scrivere `VUOTO`. Il programma deve anche controllare l'accuratezza dell'invocazione (e.g., numero di argomenti corretti) e la corretta apertura degli stream di input e output.

Supponiamo che il primo file `input.txt` contenga

```
19
Great things in business are never done by one person.
They're done by a team of people. Steve Jobs
```

e che il secondo file in input contenga

```
3 1 52 1 6 -3
```

Il programma dovrá poter essere chiamato nel seguente modo:

```
./esercizio1.out input.txt order.txt output.txt
```

e dovrá produrre un file chiamato `output.txt` che conterrá i seguenti valori:

```
business things VUOTO things done VUOTO
```

Note:

- Si assuma che la numerazione degli interi del secondo file inizi da 0.
- Si assuma inoltre che ogni stringa sia formata da al massimo 100 caratteri. Non é concesso fare assunzioni sul numero **massimo** di stringhe presenti nel file.
- E' consentito l'utilizzo della funzioni `int atoi (const char * str)` e `char * strcpy (char * destination, const char * source)` della libreria `<cstring>`.
- Non é consentito l'utilizzo di altre funzioni di libreria "particolari" diverse da quelle specificate sopra o da quelle standard necessarie a risolvere l'esercizio.
- 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`.

- 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 1 v4**

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

Sia `input.txt` un file di input contenente: un valore intero ed una serie di stringhe separate da spazi o caratteri nuova riga (`\n`). Queste stringhe possono essere ripetute molteplici volte all'interno del file. Il valore intero indica quante sono le stringhe presenti nel file. Sia `order.txt` un file in input contenente dei numeri interi separati da spazi. I numeri interi possono essere sia positivi che negativi e ci possono essere dei duplicati. Ogni numero **puó** corrispondere alla posizione di una stringa nel primo file di input.

Scrivere nel file `esercizio1.cc` un programma che, dati come argomenti da riga di comando il file di input, il file con gli interi e il nome di un file di output, scriva sul file in output le stringhe specificate dagli interi del secondo file. Nel caso uno degli interi non corrisponda a nessuna stringa del file, si dovrá scrivere `EMPTY`. Il programma deve anche controllare l'accuratezza dell'invocazione (e.g., numero di argomenti corretti) e la corretta apertura degli stream di input e output.

Supponiamo che il primo file `input.txt` contenga

```
14
If you can't make it good, at least make it look good. Bill Gates
```

e che il secondo file in input contenga

```
3 1 52 1 6 -3
```

Il programma dovrá poter essere chiamato nel seguente modo:

```
./esercizio1.out input.txt order.txt output.txt
```

e dovrá produrre un file chiamato `output.txt` che conterrá i seguenti valori:

```
make you EMPTY you at EMPTY
```

Note:

- Si assuma che la numerazione degli interi del secondo file inizi da 0.
- Si assuma inoltre che ogni stringa sia formata da al massimo 100 caratteri. Non é concesso fare assunzioni sul numero **massimo** di stringhe presenti nel file.
- E' consentito l'utilizzo della funzioni `int atoi (const char * str)` e `char * strcpy (char * destination, const char * source)` della libreria `<cstring>`.
- Non é consentito l'utilizzo di altre funzioni di libreria "particolari" diverse da quelle specificate sopra o da quelle standard necessarie a risolvere l'esercizio.
- 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`.

- 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