Università di Catania
Dipartimento di Matematica e Informatica
Corso di Studio in Informatica, A.A. 2022-2023
Compito di Programmazione 1 e Laboratorio F-N
02 febbraio 2023

Descrizione del programma

Si scriva un programma C che:

A. Prenda in input da riga di comando un parametro stringa "input" che contenga il nome di un file con estensione "txt" (ad esempio "file_di_input.txt"), e un parametro stringa "output" che contenga il nome di un file con estensione "txt" (ad esempio "file_di_output.txt"). Il programma controlla che l'utente abbia specificato il numero corretto di parametri e che i nomi dei file abbiano effettivamente estensione "txt". Se i parametri passati non rispettano i requisiti richiesti, il programma stampa un messaggio di errore sullo standard error e termina la sua esecuzione con un appropriato codice di terminazione.

B. Legga il contenuto del file e salvi i dati relativi in una lista concatenata A. Il file di testo contiene diverse righe relative a clienti di una banca nel seguente formato:

```
nome cognome numero_conto anno_apertura saldo
```

Ciascun dato contenuto nella lista concatenata deve essere una struct. Si assuma che `nome` e `cognome` siano delle stringhe di lunghezza massima 255 caratteri, `numero_conto` e `anno_apertura` (anno di apertura del conto) siano due interi e `saldo` un numero in virgola mobile a singola precisione. I dati vanno inseriti nella lista in ordine di anno di apertura del conto.

C. Determini il cliente che presenta il più alto valore di x, con x calcolato come segue:

$$x = \min\left(\frac{2023 - anno\_apertura}{5}, 1\right) \cdot \frac{saldo}{m}$$

dove m è il saldo massimo contenuto nella lista concatenata. Una volta trovato il cliente, il programma lo rimuove dalla lista.

D. L'operazione specificata al punto C deve essere eseguita h volte, con $h = \left\lfloor \frac{n}{2} \right\rfloor$, dove n è il numero di elementi letti dal file e $\left\lfloor \frac{n}{2} \right\rfloor$ rappresenta la parte intera di $\frac{n}{2}$.

E. Salvi il contenuto della lista A sul file il cui nome è indicato dal parametro "output". Il file di output dovrà seguire lo stesso formato del file di input.

Nota: gestire opportunamente i casi in cui i file non possono essere correttamente aperti in lettura o scrittura stampando un errore sullo standard error e terminando l'esecuzione del programma.

Specifiche

Il programma potrà essere articolato in un unico file sorgente, ma dovrà contenere <u>almeno le seguenti funzioni con opportuni parametri formali:</u>

- **readInput**: funzione che prende in input il vettore argv e il numero argc della funzione main(), controlli la presenza ed i requisiti degli argomenti e li inserisca in un record (struct) da

restituire allo user code (funzione main). La funzione deve gestire correttamente gli errori relativi a input non corretti;

- **readFile**: funzione per la lettura del contenuto del file. La funzione prende in input il nome del file da leggere e restituisce un riferimento alla testa della lista concatenata.

- **getMax:** funzione che calcola il saldo massimo contenuto nella lista.

- **removeAccount:** funzione che esegue l'operazione di "rimozione" dell'account con valore massimo di x dalla lista come specificato nel punto C.

- **writeFile:** funzione per la scrittura del contenuto di una lista su file come specificato nel punto E.

Note

- **Durata della prova**: 120 minuti
- **È VIETATO** usare variabili globali.
- **Si inseriscano i file sorgenti** direttamente nella propria home directory.
- **Accesso alla documentazione** disponibile tramite il browser al link: https://devdocs.io/c/.

Output di controllo

Si consideri il seguente file "input.txt" (troverete il file nella vostra home directory):

```
Amber King 7263529 2012 10283.22
Anna Knight 8827263 2022 8973.76
Amy Kelly 92837102 2015 87654.11
Alexandra Kennedy 89201736 2021 9287.33
Abigail Kelley 298367 2008 8273.15
Jennifer Kim 1726355 1998 1293.88
Heather Knutson 2983620 2019 2736.86
Alicia Kiser 982736 2001 145.45
Eleanor Kerns 827563900 2002 8273.28
Amelia Kearns 98277620 1998 982.99
```

Eseguendo il programma con il comando "`./programma input.txt output.txt`", il programma scriverà il seguente file `output.txt`:

```
Jennifer Kim 1726355 1998 1293.88
Amelia Kearns 98277620 1998 982.99
Alicia Kiser 982736 2001 145.45
Heather Knutson 2983620 2019 2736.86
Anna Knight 8827263 2022 8973.76
```

Università di Catania
Dipartimento di Matematica e Informatica
Corso di Studio in Informatica, A.A. 2022-2023
Compito di Programmazione 1 e Laboratorio F-N
23 febbraio 2023

Descrizione del programma

Si scriva un programma C che:

A. Prenda in input da riga di comando un parametro stringa *alfabeto*, un intero *n* e un nome di file *output* (ad esempio "file_di_output.txt"). Il programma controlla che la stringa alfabeto abbia una lunghezza compresa tra 5 e 15 caratteri (inclusi) e che l'intero n sia compreso tra 5 e 20 (inclusi). Se i parametri passati non rispettano i requisiti richiesti, il programma stampa un messaggio di errore sullo standard error e termina la sua esecuzione con un appropriato codice di terminazione.

B. Chieda all'utente di inserire *n* interi *x* (separati da caratteri di invio) da standard input. Il programma aggiorna il valore di *x* usando la formula:

$$x = \min(\max(1, x), L)$$

Il programma inserisce dunque ciascun valore *x* (aggiornato con la formula sopra) all'interno di un array W di lunghezza *n* (gli elementi vanno inseriti nello stesso ordine in cui vengono letti da standard input).

C. Costruisca un array di stringhe *Q* di lunghezza *n* il cui i-esimo elemento *Q[i]* sia una stringa di lunghezza *W[i]* di caratteri casuali estratti dalla stringa *alfabeto*.

D. Inizializzi una pila vuota, scorra gli elementi di *Q* nell'ordine in cui essi appaiono e il inserisca nella pila. Prima di ciascuna operazione di inserimento (ad eccezione del primo inserimento), il programma controlla se la stringa *Q[i]* da inserire ha lunghezza dispari. In tal caso, il programma estrae la stringa che si trova in cima alla pila, la concatena con la stringa *Q[i]* e inserisce nella pila la stringa risultato della concatenazione.

E. Salvi il contenuto della pila sul file il cui nome è indicato dal parametro "output".

Nota: gestire opportunamente i casi in cui i file non possono essere correttamente aperti in lettura o scrittura stampando un errore sullo standard error e terminando l'esecuzione del programma.

Specifiche

Il programma potrà essere articolato in un unico file sorgente, ma dovrà contenere almeno le seguenti funzioni con opportuni parametri formali:

- **decodeParameters**: funzione che prende in input il numero argc e il vettore argv della funzione main(), controlli la presenza e i requisiti degli argomenti e li inserisca in un record (struct) da restituire allo user code (funzione main). La funzione deve gestire correttamente gli errori relativi a input non corretti;

- **readInput**: funzione che legge l'input da tastiera e restituisce l'array W come definito nel punto B del testo;
- **sampleString:** funzione che prende in input la stringa *alfabeto* e un intero *h* e restituisca una stringa di *h* caratteri casuali campionati dalla stringa *alfabeto*.

- **getStringArray:** funzione che prende in input l'array *W* e la stringa *alfabeto* e permette di ottenere l'array di stringhe *Q* come specificato nel punto C.

- **getStack:** funzione che prende in input l'array *Q* e permette di ottenere lo stack come specificato nel punto D.

- **writeStackToFile:** funzione per la scrittura del contenuto dello stack su file come specificato nel punto E.

Note

- **Durata della prova**: 120 minuti
- **È VIETATO** usare variabili globali.
- **Si inseriscano i file sorgenti** direttamente nella propria home directory **.**
- **Accesso alla documentazione** disponibile tramite il browser al link: https://devdocs.io/c/.
- **Per la generazione di numeri casuali si utilizzi la funzione get_random fornita.**

Output di controllo

Si consideri il seguente file "input" (troverete il file nella vostra home directory):

```
6
4
-1
3
10
6
4
5
10
2
```

Eseguendo il programma con il comando:

```
./soluzione xywldbsdkm 10 out.txt < input
```

il programma scriverà il seguente file `out.txt`:

```
mk
bdyydysdxw
byddbkbdm
wykwdd
mklslbdskk
llbxkbkx
klywmm
```

Università di Catania
Dipartimento di Matematica e Informatica
Corso di Studio in Informatica, A.A. 2022-2023
Compito di Programmazione 1 e Laboratorio F-N
04 aprile 2023

Descrizione del programma

Si scriva un programma C che:

A. Prenda in input da riga di comando due parametri interi *n* e *m* e un parametro stringa *filename* che contenga un nome di file di input (ad esempio "file_di_input.txt"). Il programma controlli che i parametri *n* e *m* siano interi compresi tra 3 e 7 inclusi e che il nome del file di input abbia estensione "txt". Se i parametri passati non rispettano i requisiti richiesti, il programma stampa un messaggio di errore sullo standard error e termina la sua esecuzione con un appropriato codice di terminazione.
Si stampino a schermo i valori dei parametri presi in input.

B. Si assuma che il file di testo contenuto al percorso indicato da *filename* contenga *n* righe, ciascuna contenente *m* double separati da virgole. Il programma legga il contenuto del file e lo inserisca all'interno di un matrice *X* di double di dimensione $n \times m$.
Si stampi a schermo la matrice X.

C. Definisca un array *Y* di *n* interi e inserisca in *Y[i]* il numero di elementi nella riga i-esima di *X* che hanno un valore compreso tra *a* e *b* (esclusi), dove *a=q – (max-min)*0.3* e *b=q+(max-min)*0.3,* e *q, min, e max* sono rispettivamente il valore medio, minimo e massimo della riga i-esima di *X*. (L'operazione va effettuata per ciascun valore di i)
Si stampi a schermo l'array Y.

D. Ordini l'array *Y* in senso ascendente mediante l'algoritmo insertion sort.
Si stampi a schermo l'array Y ordinato.

E. Crei un nuovo array *Z* a partire da *Y*, tale che $Z[i] = \sum_{j=0}^{i} Y[i]$ e lo stampi a schermo.

Nota: gestire opportunamente i casi in cui i file non possono essere correttamente aperti in lettura o scrittura stampando un errore sullo standard error e terminando l'esecuzione del programma.

Specifiche

Il programma potrà essere articolato in un unico file sorgente, ma dovrà contenere almeno le seguenti funzioni con opportuni parametri formali:

- **decodeParameters**: funzione che prende in input il numero argc e il vettore argv ricevuti in input dalla funzione main(), controlli la presenza e i requisiti degli argomenti e li inserisca in un record (struct) da restituire allo user code (funzione main). La funzione deve gestire correttamente gli errori relativi a input non corretti;
- **readFile**: funzione che legge il contenuto del file e restituisce la matrice X come definito nel punto B del testo;
- **getArray**: funzione che prende in input la matrice X e restituisce l'array Y, come definito nel testo;
- **insertionSort**: funzione che permette di ordinare l'array Y mediante insertion sort;
- **getCumulative**: funzione che prende in input la l'array Y e restituisce l'array Z, come definito nel testo;

Note

- **Durata della prova**: 120 minuti
- **È VIETATO** usare variabili globali.
- **Si inseriscano i file sorgenti** direttamente nella propria home directory **.**
- **Accesso alla documentazione** disponibile tramite il browser al link: https://devdocs.io/c/.

Output di controllo

Si consideri il seguente file "input.txt" (troverete il file nella vostra home directory):

```
0.236,-0.845,0.342,-0.513,0.666,0.039,-0.784
0.382,-0.953,0.453,0.312,-0.224,0.169,-0.738
0.223,0.076,-0.545,0.735,0.638,-0.105,0.522
-0.278,0.759,0.412,0.004,-0.204,0.747,-0.656
0.514,0.053,-0.865,0.604,-0.283,0.247,-0.478
-0.915,0.386,0.042,-0.432,0.211,-0.627,0.986
-0.738,-0.693,0.620,0.938,-0.180,0.354,-0.584
```

Eseguendo il programma con il comando: `./soluzione 7 7 input.txt`
il programma stamperà su standard output il seguente contenuto (lo trovate nel file "output.txt" nella home directory):

```
======Stampa Parametri======
n = 7
m = 7
filename = input.txt

======Stampa X======
 0.24 -0.84  0.34 -0.51  0.67  0.04 -0.78
 0.38 -0.95  0.45  0.31 -0.22  0.17 -0.74
 0.22  0.08 -0.55  0.73  0.64 -0.10  0.52
-0.28  0.76  0.41  0.00 -0.20  0.75 -0.66
 0.51  0.05 -0.86  0.60 -0.28  0.25 -0.48
-0.92  0.39  0.04 -0.43  0.21 -0.63  0.99
-0.74 -0.69  0.62  0.94 -0.18  0.35 -0.58

======Stampa Y======
3 3 4 4 3 4 2

======Stampa Y ordinato======
2 3 3 3 4 4 4

======Stampa Z======
2 5 8 11 15 19 23
```

Università di Catania
Dipartimento di Matematica e Informatica
Corso di Studio in Informatica, A.A. 2022-2023
Compito di Programmazione 1 e Laboratorio F-N
27 giugno 2023

Descrizione del programma

Si scriva un programma C che:

A. Prenda in input da riga di comando un parametro stringa *filename* che contenga un nome di file di output (ad esempio "file_di_output.bin") e un parametro intero *n*. Il programma controlli che il nome del file abbia estensione "bin" e che *n* sia compreso tra 3 e 5. Se i parametri passati non rispettano i requisiti richiesti, il programma stampa un messaggio di errore sullo standard error e termina la sua esecuzione con un appropriato codice di terminazione.
Si stampino a schermo i valori dei parametri presi in input.

B. Chieda all'utente di inserire da riga di comando un numero arbitrario di record bancari separati da caratteri di "invio". Ciascun record deve seguire il formato:

numeroConto nome cognome saldo

Ciascun record viene inserito in una apposita struct *Record* (si assumano stringhe lunghe al massimo 255 caratteri) e inserito in una lista concatenate *L*. L'inserimento termina quando l'utente dà il carattere di end of file (CTRL + D o CTRL + Z a seconda del sistema).
Si stampi a schermo il contenuto di L dopo le operazioni di inserimento.

C. Dichiari un array *X* di struct *Record* e copi i record dalla lista *L* all'interno di *X*. Ordini mediante selectionsort gli elementi di *X* in ordine ascendente rispetto al seguente valore:

(saldo[i] – m)* (saldo[i] – m)

Dove saldo[i] è il saldo dell'i-esimo elemento di *X*, mentre *m* è il saldo medio.
Si stampi a schermo il contenuto di X al termine dell'operazione.

D. Inizializzi un file binario in *filename* e inserisca al suo interno gli elementi di *X*. Legga dunque in ordine casuale e stampi a schermo i primi 5 elementi in *filename*.

Nota: gestire opportunamente i casi in cui i file non possono essere correttamente aperti in lettura o scrittura stampando un errore sullo standard error e terminando l'esecuzione del programma.

Specifiche

Il programma potrà essere articolato in un unico file sorgente, ma dovrà contenere almeno le seguenti funzioni con opportuni parametri formali:

- **decodeParameters**: funzione che prende in input il numero argc e il vettore argv ricevuti in input dalla funzione main(), controlli la presenza e i requisiti degli argomenti e li inserisca in un record (struct) da restituire allo user code (funzione main). La funzione deve gestire correttamente gli errori relativi a input non corretti;
- **readRecord**: legge un singolo record da riga di comando e restituisce una struct *Record*;
- **loadRecords**: legge tutti i record da riga di comando e li conserva nella lista *L*;

- **getArray**: funzione che prende in input la pila *S* e restituisce l'array *X*, come definito nel testo;
- **sort**: funzione che permette di ordinare l'array *X* mediante selectionsort;
- **saveToFile**: funzione che salva il contenuto di *X* su file;
- **showFileContent**: funzione che legge il contenuto del file e lo mostra a schermo.

Note

- **Durata della prova**: 120 minuti
- **È VIETATO** usare variabili globali.
- **Si inseriscano i file sorgenti** direttamente nella propria home directory.
- **Accesso alla documentazione** disponibile tramite il browser al link: https://devdocs.io/c/.

Output di controllo

Si consideri il seguente file "input" (troverete il file nella vostra home directory):

```
123456 John Doe 6253.50
987654 Alice Smith 166.75
567890 Michael Johnson 93826.20
234567 Sarah Williams 177625.00
876543 David Brown 8276.80
345678 Emily Miller 222763.25
654321 James Anderson 29376.50
432109 Olivia Thomas 166524.10
765432 Robert Wilson 888827.75
210987 Sophia Jackson 2736.60
```

Eseguendo il programma con il comando: `./soluzione out.bin 4 < input.txt`
il programma stamperà su standard output il seguente contenuto (lo trovate nel file "output.txt" nella home directory):

```
PUNTO A - valori dei parametri presi in input:
filename = s.bin
n = 4

PUNTO B - Contenuto di L:
123456 John Doe 6253.50
987654 Alice Smith 166.75
567890 Michael Johnson 93826.20
234567 Sarah Williams 177625.00
876543 David Brown 8276.80
345678 Emily Miller 222763.25
654321 James Anderson 29376.50
432109 Olivia Thomas 166524.10
765432 Robert Wilson 888827.75
210987 Sophia Jackson 2736.60
```

PUNTO C - Contenuto di X dopo l'ordinamento:
432109 Olivia Thomas 166524.10
234567 Sarah Williams 177625.00
345678 Emily Miller 222763.25
567890 Michael Johnson 93826.20
654321 James Anderson 29376.50
876543 David Brown 8276.80
123456 John Doe 6253.50
210987 Sophia Jackson 2736.60
987654 Alice Smith 166.75
765432 Robert Wilson 888827.75

PUNTO D - Contenuto del file binario in ordine casuale:
345678 Emily Miller 222763.25
567890 Michael Johnson 93826.20
432109 Olivia Thomas 166524.10
234567 Sarah Williams 177625.00

Università di Catania
Dipartimento di Matematica e Informatica
Corso di Studio in Informatica, A.A. 2022-2023
Compito di Programmazione 1 e Laboratorio F-N
20 luglio 2023

Descrizione del programma

Si scriva un programma in C che:

A. Prenda in input da riga di comando due parametri interi, *n* e *m* compresi tra 3 e 7 (inclusi). Il programma deve verificare che *n* e *m* siano entrambi numeri interi positivi. Se i parametri passati non rispettano i requisiti richiesti, il programma stampa un messaggio di errore sullo standard error e termina la sua esecuzione con un appropriato codice di terminazione. Si stampino a schermo i valori dei parametri presi in input.
Si stampino a schermo i valori dei parametri presi in input.

B. Chieda all'utente di inserire da riga di comando un valore intero *h* compreso tra 10 e 100 (inclusi). Il programma controlli che il valore inserito rispetti i requisiti richiesti. Il programma crei dunque una matrice *A* di dimensione $n \times m$ di puntatori a numeri interi generati casualmente nell'intervallo [0, h] (estremi inclusi) mediante la funzione *get_random* fornita.
Si stampi a schermo il contenuto della matrice A.

C. Elimini in ciascuna colonna *j* di *A* i tre valori più grandi della colonna. L'eliminazione di un elemento $A[i][j]$ della matrice va effettuata ponendo a *NULL* l'elemento $A[i][j]$ e liberando la memoria occupata dall'intero puntato da *A[i][j]*. SUGGERIMENTO: è possibile rimuovere i tre elementi più grandi effettuando tre volte la ricerca del massimo ed eliminando di volta in volta il valore trovato.
Si stampi a schermo il contenuto della matrice A dopo l'operazione. Si indichino i puntatori NULL con degli asterischi.

D. Si stampino su standard output i valori della riga contenente il numero minore di valori NULL. NOTA: qualora più righe dovessero avere lo stesso numero x di valori NULL, sarà sufficiente stampare una qualsiasi riga contenente x valori NULL.

Nota: gestire opportunamente i casi in cui i file non possono essere correttamente aperti in lettura o scrittura stampando un errore sullo standard error e terminando l'esecuzione del programma.

Specifiche

Il programma potrà essere articolato in un unico file sorgente, ma dovrà contenere almeno le seguenti funzioni con opportuni parametri formali:

- **decodeParameters:** funzione che prende in input il numero argc e il vettore argv ricevuti in input dalla funzione main(), controlli la presenza e i requisiti degli argomenti e li inserisca in un record (struct) da restituire allo user code (funzione main). La funzione deve gestire correttamente gli errori relativi a input non corretti;
- **getRandomMatrix:** funzione che prende in input le dimensioni *n* e *m* della matrice e il valore *h* e restituisce una matrice $n \times m$ di puntatori a numeri interi generati casualmente come indicato al punto B;
- **removeMax:** funzione che prende in input la matrice A, la dimensione *n* e un indice di colonna *col*, e rimuove il massimo valore nella colonna *col* come indicato nel punto C;

- **modifyMatrix:** funzione che prende in input la matrice A, le dimensioni *n* e *m*, e rimuova i tre numeri più grandi da ogni colonna di A come indicato nel punto C;
- **rowMinNull:** funzione che prende in input la matrice A, le dimensioni *n* e *m*, e trova la riga contenente il numero minore di valori NULL (puntatori NULL);

Note

- **Durata della prova**: 120 minuti
- **È VIETATO** usare variabili globali.
- **Si inseriscano i file sorgenti** direttamente nella propria home directory.
- **Per la generazione di numeri casuali**, si usi la funzione "get_random" definita nel file "get_random.c" (si copi e questa definizione e la si incolli nel main).
- **Accesso alla documentazione** disponibile tramite il browser al link: https://devdocs.io/c/.

Output di controllo

Si consideri il file `input.txt` contenente il numero "20" (è possibile trovare il file nella home directory).

Eseguendo il programma con il comando: `./soluzione 5 7 < input.txt` il programma stamperà su standard output il seguente contenuto (lo trovate nel file "output.txt" nella home directory):

```
Punto A - Parametri inseriti: n=5, m=7
Inserisci un numero compreso tra 10 e 100:

Punto B - Contenuto di A:
   18   16   19    2   18   11   10
    7    9   17   11    7   15    9
    1    7    7   16   11   17    9
   17    3    1   13   12    2    3
    3   12    5   17    5    8   16

Punto C - Contenuto di A dopo la modifica:
    *    *    *    2    *    *    *
    *    *    *   11    7    *    *
    1    7    *    *    *    *    9
    *    3    1    *    *    2    3
    3    *    5    *    5    8    *

Punto D - Riga con minor numero di NULL:
    *    3    1    *    *    2    3
```

Università di Catania
Dipartimento di Matematica e Informatica
Corso di Studio in Informatica, A.A. 2022-2023
Compito di Programmazione 1 e Laboratorio F-N
06 settembre 2023

Descrizione del programma

Si scriva un programma in C che:

A. Prenda in input da riga di comando un parametro intero, *n* compreso tra 5 e 20 (inclusi) e un parametro stringa *inputFileName* che indica il nome di un file con estensione ".dat". Il programma deve verificare che *n* sia un numero interi nel range indicato e che il nome di file specificato abbia l'estensione indicata. Se i parametri passati non rispettano i requisiti richiesti, il programma stampa un messaggio di errore sullo standard error e termina la sua esecuzione con un appropriato codice di terminazione.
Si stampino a schermo i valori dei parametri presi in input.

B. Legga i numeri interi contenuti nel file specificato dal parametro *inputFileName* e li inserisca all'interno di un array di interi *A* (si veda il contenuto del file `input.dat` per un esempio del formato del file). Il programma dovrà scorrere il contenuto del file due volte. La prima volta il programma determinerà il numero di elementi contenuti nel file in modo da inizializzare un array *A* della dimensione corretta. La seconda volta, il programma inserirà gli elementi del file nell'array *A* nell'ordine in cui essi appaiono nel file.
Si stampi a schermo il contenuto dell'array A.

C. Inizializzi una pila di float *P*. Scorra gli elementi di *A* dal primo all'ultimo. Dato l'elemento *A[i]*, il programma controlli che tale elemento sia un multiplo del parametro *n*. Se A[i] è un multiplo di *n*, il programma lo inserisce in cima alla pila, altrimenti, il programma estrae (con una operazione di pop) l'elemento in cima alla pila (sia esso *x*) e inserisce in pila (con una operazione di push), la media aritmetica tra *x* e *A[i]*.
Si stampi a schermo il contenuto della pila P al termine dell'operazione.

D. Si stampino su standard output il valore massimo, il valore minimo, e il valore medio contenuti nella pila.

Nota: gestire opportunamente i casi in cui i file non possono essere correttamente aperti in lettura o scrittura stampando un errore sullo standard error e terminando l'esecuzione del programma.

Specifiche

Il programma potrà essere articolato in un unico file sorgente, ma dovrà contenere almeno le seguenti funzioni con opportuni parametri formali:

- **decodeParameters:** funzione che prende in input il numero argc e il vettore argv ricevuti in input dalla funzione main(), controlli la presenza e i requisiti degli argomenti e li inserisca in un record (struct) da restituire allo user code (funzione main). La funzione deve gestire correttamente gli errori relativi a input non corretti;
- **readInput**: funzione che legge i numeri contenuti nel file di input e restituisce un array A contenente tali numeri;
- **createStack**: funzione che scorre gli elementi dell'array A e costruisce la pila come specificato nel punto C;

- **findMinMaxMean**: funzione che prende in input un riferimento alla pila e restituisce al chiamante il valore massimo, minimo e medio della pila.

Note

- **Durata della prova**: 120 minuti
- **È VIETATO** usare variabili globali.
- **Si inseriscano i file sorgenti** direttamente nella propria home directory **.**
- **Accesso alla documentazione** disponibile tramite il browser al link: https://devdocs.io/c/.

Output di controllo

Si consideri il file `input.dat` con il seguente contenuto (è possibile trovare il file nella home directory):

```
42
18
76
89
53
27
65
9
34
71
5
84
12
61
98
47
22
56
3
67
```

Eseguendo il programma con il comando: `./soluzione 6 input.dat`
il programma stamperà su standard output il seguente contenuto (lo trovate nel file `output.txt` nella home directory):

```
Punto A - Parametri inseriti: n=0,
inputFileName=input.dat

Punto B - Contenuto di A:
42
18
76
```

```
89
53
27
65
9
34
71
5
84
12
61
98
47
22
56
3
67

Punto C - Contenuto di S:
46.20
84.00
28.46
42.00

Punto D:
Valore minimo: 28.00
Valore massimo: 84.00
Valore medio: 50.05
```

Università di Catania
Dipartimento di Matematica e Informatica
Corso di Studio in Informatica, A.A. 2022-2023
Compito di Programmazione 1 e Laboratorio F-N
21 settembre 2023

Descrizione del programma

Si scriva un programma in C che:

A. Prenda in input come argomenti un parametro intero *min* compreso tra 7 e 12 (inclusi), un parametro intero *max* compreso tra 7 e 12 (inclusi), e un parametro intero *n* positivo e diverso da zero. Il programma deve verificare che *min*, *max* e *n* siano numeri interi con le caratteristiche specificate, che *min*<*max*. Se i parametri passati non rispettano i requisiti richiesti, il programma stampa un messaggio di errore sullo standard error e termina la sua esecuzione con un appropriato codice di terminazione.
Si stampino a schermo i valori dei parametri presi in input.

B. Inizializzi un array *A* di stringhe di dimensione *n* e inserisca all'interno di ciascuna posizione *A[i]* (dove *i* è l'i-esima posizione all'interno dell'array) una stringa di *x* vocali estratte in maniera pseudocasuale, dove *x* è un numero pseudocasuale compreso tra *min* e *max* (inclusi).
Si stampi a schermo il contenuto dell'array A.

C. Ordini l'array *A* mediante l'algoritmo "insertion sort" in maniera lessicografica. Una volta effettuato l'ordinamento, il programma concatena le stringhe contenute in *A* nell'ordine in cui appaiono, conservando il contenuto in una nuova stringa *B*.
Si stampi a schermo il contenuto della stringa B.

D. Il programma sostituisce dunque con un asterisco (*) tutte le occorrenze di "u" precedute da "e" nella stringa. Ad esempio, la stringa "aiueuieaa" diventerebbe "aiue*ieaa".
Si stampi a schermo il contenuto della stringa B dopo l'operazione di sostituzione.

Nota: gestire opportunamente i casi in cui i file non possono essere correttamente aperti in lettura o scrittura stampando un errore sullo standard error e terminando l'esecuzione del programma.

Specifiche

Il programma potrà essere articolato in un unico file sorgente, ma dovrà contenere almeno le seguenti funzioni con opportuni parametri formali:

- **decodeParameters:** funzione che prende in input il numero argc e il vettore argv ricevuti in input dalla funzione main(), controlli la presenza e i requisiti degli argomenti e li inserisca in un record (struct) da restituire allo user code (funzione main). La funzione deve gestire correttamente gli errori relativi a input non corretti;
- **generateString**: funzione che genera una stringa di vocali di lunghezza specificata, come descritto al punto B.
- **makeArray**: funzione che costruisce l'array *A* come specificato al punto B.
- **sortArray**: funzione che ordina l'array come specificato nel punto C.
- **concatString**: funzione che concatena le stringhe contenute nell'array *A* in una stringa, come specificato nel punto C.
- **replaceCharacters**: funzione che sostituisce le "u" precedute da "e" in una stringa specificata, come indicato nel punto D.

Note

- **Durata della prova**: 120 minuti
- **È VIETATO** usare variabili globali.
- **Si inseriscano i file sorgenti** direttamente nella propria home directory **.**
- **Per la generazione di numeri casuali**, si usi la funzione "get_random" definita nel file "get_random.c" (si copi e questa definizione e la si incolli nel main).
- **Accesso alla documentazione** disponibile tramite il browser al link: https://devdocs.io/c/.

Output di controllo

Eseguendo il programma con il comando: `./soluzione 8 10 12`
il programma stamperà su standard output il seguente contenuto (lo trovate nel file `output.txt` nella home directory):

```
Punto A - Parametri inseriti: min=8, max=10, n=12

Punto B:
oeiuuooa
oaoauooeoa
eooieoiu
aeiiaoau
aieeueeuai
ouaeeooa
uoiiiiei
iauueoeiii
oauoaeeie
eeuuiuiea
iouaauoo
iuaiieuee

Punto C:
oeiuuooaoaoauooeoaeooieoiuaeiiaoauaieeueeuaiouaeeooauoiii
ieiiauueoeiiioauoaeeieeeuuiuieaiouaauooiuaiieuee

Punto D:
oeiuuooaoaoauooeoaeooieoiuaeiiaoauaiee*ee*aiouaeeooauoiii
ieiiauueoeiiioauoaeeieee*uiuieaiouaauooiuaiie*ee
```

Università di Catania
Dipartimento di Matematica e Informatica
Corso di Studio in Informatica, A.A. 2022-2023
Compito di Programmazione 1 e Laboratorio F-N
05 dicembre 2023

Descrizione del programma

Si scriva un programma C che:

A. Prenda in input da riga di comando un parametro stringa *simboli*, un intero *n* e un nome di file *output* (ad esempio "file_di_output.txt"). Il programma controlla che la stringa *simboli* abbia una lunghezza *L* compresa tra 10 e 20 caratteri (inclusi) e che l'intero n sia compreso tra 8 e 18 (inclusi). Se i parametri passati non rispettano i requisiti richiesti, il programma stampa un messaggio di errore sullo standard error e termina la sua esecuzione con un appropriato codice di terminazione.
Si stampino i parametri presi in input.

B. Chieda all'utente di inserire *n* interi *x* (separati da caratteri di invio) da standard input. Il programma calcola il valore:
$$\mathrm{y} = \mathrm{x}! = x(x-1)(x-2)\ldots 1$$
Il programma inserisce dunque ciascun valore *y* all'interno di un array W di lunghezza *n* (gli elementi vanno inseriti nello stesso ordine in cui vengono letti da standard input).
Si stampi il contenuto di W.

C. Costruisca un array di stringhe *Q* di lunghezza *n* il cui i-esimo elemento *Q[i]* sia una stringa di lunghezza $W[i] \bmod L$ di caratteri casuali estratti dalla stringa *simboli*, dove L è la lunghezza della stringa *simboli* e "mod" rappresenta l'operazione modulo.
Si stampi il contenuto di Q.

D. Inizializzi una pila vuota, scorra gli elementi di *Q[i]* dell'array *Q* nell'ordine in cui essi appaiono. Per ciascuna stringa *Q[i]*, il programma controlla se la sua lunghezza *h* è pari. Se *h* è pari, o se si tratta del primo inserimento in pila, il programma inserisce regolarmente la stringa *Q[i]* in cima alla pila. Se *h* è dispari, il programma estrae la stringa che si trova in cima alla pila con una operazione di pop (sia tale stringa a) e inserisce nella pila la stringa b contenente i caratteri di *Q[i]* che appaiono in *a*, nell'ordine in cui appaiono in *Q[i]*.
Si stampi il contenuto della pila.

E. Salvi il contenuto della pila sul file il cui nome è indicato dal parametro *output*.

Nota: gestire opportunamente i casi in cui i file non possono essere correttamente aperti in lettura o scrittura stampando un errore sullo standard error e terminando l'esecuzione del programma.

Specifiche

Il programma potrà essere articolato in un unico file sorgente, ma dovrà contenere almeno le seguenti funzioni con opportuni parametri formali:

- **decodeParameters**: funzione che prende in input il numero argc e il vettore argv della funzione main(), controlli la presenza e i requisiti degli argomenti e li inserisca in un record (struct) da restituire allo user code (funzione main). La funzione deve gestire correttamente gli errori relativi a input non corretti;

- **readInput**: funzione che legge l'input da tastiera e restituisce l'array W come definito nel punto B del testo;

- **sampleString:** funzione che prende in input la stringa *simboli* e un intero *h* e restituisca una stringa di *h* caratteri casuali campionati dalla stringa *simboli*.

- **getStringArray:** funzione che prende in input l'array *W* e la stringa *simboli* e permette di ottenere l'array di stringhe *Q* come specificato nel punto C.

- **getStack:** funzione che prende in input l'array *Q* e permette di ottenere lo stack come specificato nel punto D.

- **writeStackToFile:** funzione per la scrittura del contenuto dello stack su file come specificato nel punto E.

Note

- **Durata della prova**: 120 minuti
- **È VIETATO** usare variabili globali.
- **Si inseriscano i file sorgenti** direttamente nella propria home directory **.**
- **Accesso alla documentazione** disponibile tramite il browser al link: https://devdocs.io/c/.
- **Per la generazione di numeri casuali si utilizzi la funzione get_random fornita.**

Output di controllo

Si consideri il seguente file "input" (troverete il file nella vostra home directory):

```
2
5
3
5
2
3
4
2
3
3
```

Eseguendo il programma con il comando:

```
./soluzione jshdywjdufhfghdywb 10 out.txt < input
```

il programma genererà il seguente output:

```
Punto A - Parametri: simboli = jshdywjdufhfghdywb,
output=out.txt, n = 10

Punto B - Contenuto di W:
    2  120    6  120    2    6   24    2    6    6
```

```
Punto C - Contenuto di Q:
gh
hufwssfddsjg
dhdhfw
gujwyydggyff
ww
dhsywf
dfdgsh
hd
wdhbdu
fjjhfu


Punto D - Contenuto della pila:
fjjhfu
wdhbdu
hd
dfdgsh
dhsywf
ww
gujwyydggyff
dhdhfw
hufwssfddsjg
gh
```

e scriverà il seguente file `out.txt`:

```
fjjhfu
wdhbdu
hd
dfdgsh
dhsywf
ww
gujwyydggyff
dhdhfw
hufwssfddsjg
gh
```