Università di Catania
Dipartimento di Matematica e Informatica
Corso di Studio in Informatica, A.A. 2022-2023
Prova di Laboratorio
24 Febbraio 2023

Descrizione del programma
Scrivere un programma in C che:

- **A.** Prenda un input da tastiera (argomenti della funzione main) costituito da un intero positivo **N in** [5,10] e due caratteri **a** e **b** in [‘a’,’z’]. Il programma deve verificare che i parametri N, a e b rispettino le specifiche e restituire in caso contrario un messaggio di errore.
- **B.** Iteri la seguente procedura N volte, facendo uso di una struttura dati LIFO dinamica (pila o stack) da implementare mediante lista concatenata semplice (top==testa della lista):
    - per ognuno degli N passi, si generi innanzitutto un carattere x in [‘1’-’9’];
        - se x rappresenta un numero in [‘1’-’4’], allora si proceda ad x operazioni di inserimento (push) di caratteri pseudo-casuali che siano vocali, seguite dall’inserimento del carattere x;
        - se x rappresenta un numero in [‘5’-’9’], allora si proceda ad x operazioni di inserimento (push) di caratteri pseudo-casuali che siano consonanti, seguite dall’inserimento del carattere x;
        - in particolare, sia c il generico carattere da inserire sullo stack:
            - se c==a, si inserisca sullo stack al posto di esso il carattere '*';
            - se c==b, si inserisca sullo stack al posto di esso il carattere '?';
        - si inserisca infine il carattere x sullo stack
- **C** Stampi, sullo standard output, l’intero stack.

Specifiche

Il programma potra’ essere strutturato in un unico file sorgente, ma dovra’ contenere almeno le seguenti funzioni:

- **readInput:** funzione che prende in input l'array di puntatori a carattere argv della funzione main, controlli che gli argomenti richiesti siano nei limiti specificati, e restituisca il record (struct) che contiene tali parametri; se il controllo non va a buon fine, stampa un messaggio sullo standard error e termina il programma.
- **genVowel:** funzione che produca un carattere vocale pseudo-casuale;

- **genConsonant:** funzione che produca un carattere consonante pseudo-casuale;
- **push:** funzione che consente di inserire un elemento sullo stack;
- **fillStack:** funzione con opportuni parametri formali che rappresenti l'implementazione della procedura descritta nel punto B;
- **printStack:** funzione per la stampa dello stack.

Note
Durata della prova: 120 minuti
Generazione di numeri pseudocasuali:

- Si consideri la seguente funzione get_random() per la generazione di numeri pseudo-casuali interi positivi (qualora necessaria):

  https://pastebin.com/f6eAKNQy

```
unsigned int get_random() {
      static unsigned int m_w = 123456;
      static unsigned int m_z = 789123;
      m_z = 36969 * (m_z & 65535) + (m_z >> 16);
      m_w = 18000 * (m_w & 65535) + (m_w >> 16);
      return (m_z << 16) + m_w;
}
```

NB: Ai fini della eventuale generazione di numeri in virgola mobile, si faccia uso della costante UINT_MAX (<limits.h>) unitamente alla funzione get_random().

È VIETATO usare variabili globali.

OUTPUT DI CONTROLLO/ESEMPIO:

```
./a.out 5 t i

 ** TOP->  4 ? ? ? e 9 z j f q p c * b v 8 m n k * x k d j 7 * v b v h c z 8 p g f p s x q b <- BOTTOM **
```