National Institute of Technology Calicut
Department of Computer Science and Engineering
Third Semester B. Tech.(CSE)
CS2092D Programming Laboratory
Assignment #7

Submission deadline (on or before):

- 04.11.2020, 10:00 PM

Policies for Submission and Evaluation:

- You must submit your assignment in the Eduserver course page, on or before the submission deadline.
- Ensure that your programs will compile and execute without errors in the Linux platform.
- During the evaluation, failure to execute programs without compilation errors may lead to zero marks for that evaluation.
- Detection of ANY malpractice related to the lab course can lead to awarding an F grade in the course.

Naming Conventions for Submission

- Submit a single ZIP (.zip) file (do not submit in any other archived formats like .rar, .tar, .gz). The name of this file must be

  `ASSG<NUMBER>_<ROLLNO>_<FIRST-NAME>.zip`

  (Example: *ASSG*1_*BxxyyyyCS_LAXMAN.zip*). DO NOT add any other files (like temporary files, input files, etc.) except your source code, into the zip archive.

- The source codes must be named as

  `ASSG<NUMBER>_<ROLLNO>_<FIRST-NAME>_<PROGRAM-NUMBER>.c`

  (For example: *ASSG*1_*BxxyyyyCS_LAXMAN*_1.*c*). If you do not conform to the above naming conventions, your submission might not be recognized by our automated tools, and hence will lead to a score of 0 marks for the submission. So, make sure that you follow the naming conventions.

Standard of Conduct

- Violation of academic integrity will be severely penalized. Each student is expected to adhere to high standards of ethical conduct, especially those related to cheating and plagiarism. Any submitted work MUST BE an individual effort. Any academic dishonesty will result in zero marks in the corresponding exam or evaluation and will be reported to the department council for record keeping and for permission to assign F grade in the course. The department policy on academic integrity can be found at: http://cse.nitc.ac.in/sites/default/files/Academic-Integrity_new.pdf.

General Instructions

- Programs should be written in C language and compiled using C compiler in Linux platform. **Submit the programs for questions 1,2,3 and 4 through the submission link in Eduserver. You should complete the program for questions 5 and 6 before the next lab session. During the evaluation we may be asking you to modify any of these three programs**.

QUESTIONS

1. Write a menu driven program to implement a STACK S of at most n elements using an array $A[0..n-1]$. STACK is to be declared as a struct with an attribute *top* and an array $A[0..n-1]$ as members. The attribute *top* indexes the most recently inserted element in the stack. The stack consists of elements $A[0..S.top]$ where *A[0]* is the element at the bottom of the stack and *A[S.top]* is the element at the top. Your program must contain the following functions: (in function prototypes, S denotes a stack, and k denotes an integer. All operations should run in *O(1)* time.)

   - MAIN() - repeatedly reads a character 'i', 'd', 'e' or 't' from the terminal and calls the appropriate function until character 't' is entered.
   - STACK-EMPTY(S) - checks whether the Stack S is empty or not.
   - STACK-FULL(S) - checks whether the Stack S is full or not.
   - PUSH(S, k) - inserts k to the top of S (check STACK-FULL() inside this function).
   - POP(S) - deletes the most recently inserted element from S.

   Input format:

   - The first line of the input contains an integer $n \in [0, 10^5]$, the size of the array A.
   - Upcoming lines contain a character from 'i', 'd', 'e', or 't' followed by zero or one integer. The integer, if given, is in the range $[-10^6, 10^6]$.
   - Character 'i' is followed by an integer separated by space. In this operation, the integer is inserted to the *top* of S.
   - Character 'd' is to delete and print the most recently inserted element from S.
   - Character 'e' is to check whether the Stack is empty or not.
   - Character 't' is to 'terminate' the program.

   Output Format:

   - The output (if any) of each command should be printed on a separate line.
   - For option 'd', print the deleted element. If S is empty, then print -1.
   - For option 'e', if S is not empty, then print 1. If S is empty, then print -1.

   Sample Input :

   ```
   10
   i 8
   i 10
   d
   i 12
   d
   d
   d
   e
   i 18
   e
   t
   ```

   Sample Output:

   ```
   10
   12
   8
   -1
   -1
   1
   ```

2. Write a menu driven program to implement a STACK S using a singly linked list. Each node in the list has an integer attribute *data* and a pointer attribute *next*. Keep $S.top$ point to the first node (node at the front end) of the list. Your program must contain the following functions: (in function prototypes, S denotes a Stack, k an integer value, and x a node in the list. All operations should run in *O(1)* time.)

   - MAIN() - repeatedly reads a character 'i', 'd', 'e' or 't' from the terminal and calls the appropriate function until character 't' is entered.
   - CREATE-NODE(k) - creates a new node with *data* value k and returns a pointer to the created node. The attribute *next* of the new node should be set as NULL.
   - PUSH(S, x) - inserts x as the new top node of S.
   - POP(S) - deletes the most recently inserted node from S.
   - STACK-EMPTY(S) - checks whether the Stack is empty or not.

   Note:- For every PUSH() operation, the node x is to be created by calling the CREATE-NODE() function.

   Input format:

   - Each line contains a character from 'i', 'd', 'e', or 't' followed by zero or one integer. The integer, if given, is in the range $[-10^6, 10^6]$.
   - Character 'i' is followed by an integer separated by space. In this operation, the node with this integer as data is inserted to the front end of S.
   - Character 'd' is to delete the most recently inserted node and the deleted node's data value is printed.
   - Character 'e' is to check whether the Stack is empty or not.
   - Character 't' is to 'terminate' the program.

   Output Format:

   - The output (if any) of each command should be printed on a separate line.
   - For option 'd' print the deleted element. If S is empty, then print -1.
   - For option 's' if S is not empty, then print 1. If S is empty, then print -1.

   Sample Input :

   ```
   i 10
   i 15
   d
   i 12
   d
   d
   d
   e
   i 20
   e
   t
   ```

   Sample Output:

   ```
   15
   12
   10
   -1
   -1
   1
   ```

3. Write a menu driven program to implement a QUEUE Q using an array $A[0..n-1]$. The queue is to be declared as a struct with two attributes- *head* and *tail*. The attribute *Q.head* indexes its head and *Q.tail* indexes the next location at which a newly arriving element will be inserted into the queue. The elements in the queue reside in locations *Q.head, Q.head+1,...., Q.tail-1*, where the location *0* immediately follows location *n-1* in a circular order.Your program must contain the following functions: (in function prototypes, Q denotes a Queue and k denotes an integer. All operations should run in *O(1)* time.)

   - MAIN() - repeatedly reads a character 'i', 'd', 'f' or 'e' from terminal and calls the appropriate function until character 't' is entered.
   - QUEUEFULL(Q) - checks whether the Queue is full or not.
   - QUEUEEMPTY(Q) - checks whether the Queue is empty or not.
   - ENQUEUE(Q, k) - inserts the element k to the tail of Q (check QUEUEFULL() inside this function).
   - DEQUEUE(Q) - deletes the element from the head of Q (check QUEUEEMPTY() inside this function).

   Input format:

   - The first line of the input contains an integer $n \in [0, 10^5]$, the size of the array A.
   - Upcoming lines contain a character from 'i', 'd', 'f' or 'e' followed by zero or one integer. The integer, if given, is in the range $[-10^6, 10^6]$.
   - Character 'i' is followed by an integer separated by space. In this operation, this integer is inserted to the tail of Q.
   - Character 'd' is to delete first element from Q and print the deleted element.
   - Character 'f' is to check whether the Queue is full or not.
   - Character 'e' is to check whether the Queue is empty or not.
   - Character *'t'* is to 'terminate' the program.

   Output Format:

   - The output (if any) of each command should be printed on a separate line.
   - For option 'i' if Q is full, then print -1.
   - For option 'd' delete the first node and print the deleted node's key. If Q is empty, then print -1.
   - For option 'f', if Q is not full, then print 1. If Q is full, then print -1.
   - For option 'e', if Q is not empty, then print 1. If Q is empty, then print -1.

   Sample Input

   5
   i 20
   i 38
   d
   d
   d
   e
   i 40
   e
   i 35
   i 42
   f
   i 33

i 27
f
t

Sample Output

20
38
-1
-1
1
1
-1

4. Write a menu driven program to implement a QUEUE Q using a singly linked list. Queue is to be declared as a *struct* with two pointer attributes *head* and *tail*, pointing to the first node and the last node of the list respectively. Each node in the list is an object with an attribute *data* and a pointer attribute, *next*. Your program must contain the following functions: (in function prototypes, Q denotes a Queue, , k an integer (k need not be distinct) and , x a node. All operations should run in *O(1)* time.).

   - MAIN() - repeatedly reads a character 'i', 'd', 'f' or 'e' from the terminal and calls the appropriate function until character 't' is entered.
   - CREATE-NODE(k) - creates a new node with data value k and returns a pointer to the new node. The *next* field should be set as NULL.
   - QUEUEEMPTY(Q) - checks whether the Queue is empty or not.
   - ENQUEUE(Q, x) - inserts x to the tail of Q.
   - DEQUEUE(Q) - deletes the first node from Q (check QUEUEEMPTY() inside this function).

   Note:- For every ENQUEUE operation, the node x is to be created by calling CREATE-NODE() function.

   Input format:

   - Each line contains a character from 'i', 'd', 'e' or 't' followed by zero or one integer. The integer, if given, is in the range $[-10^6, 10^6]$.
   - Character 'i' is followed by an integer separated by space. In this operation, the node with this integer as data is inserted to the tail of Q.
   - Character 'd' is to delete the first node and the deleted node's key is printed.
   - Character 'e' is to check whether the Queue is empty or not.
   - Character 't' is to 'terminate' the program.

   Output Format:

   - The output (if any) of each command should be printed on a separate line.
   - For option 'd' delete the first node and print the deleted node's key. If Q is empty, then print -1.
   - For option 'e', if Q is not empty, then print 1. If Q is empty, then print -1.

   Sample Input

   i 30

i 48
d
d
d
e
i 50
e
t

Sample Output

30
48
-1
-1
1

5. A palindrome is a nonempty string that reads the same forward and backward. Write a program that reads a string and checks whether the given string is palindrome or not using a STACK. Use the implementation of Stack in Question 1 with the exception that elements are of `char` type. Write a function ISPALINDROME(*str*), that given a string *str*, returns 1 if *str* is a palindrome and 0 otherwise.

   Input format:

   - The first line of the input contains a string *s*.

   Output format:

   - If the string *s* is palindrome, print 1. Otherwise, print 0.

   Sample Input 1:
   malayalam
   Sample Output 1:
   1

   Sample Input 2:
   calicut
   Sample Output 2:
   0

6. In a computer science laboratory, all computer systems are connected to a single printer. All these systems can generate a print request for printing files. The printer can print only one file at a time. While the printer is printing a file, all other print requests need to be wait until the printer is free. Once the printer is free, the files are printed one by one in the order in which they come(i.e first in first out). The printer maintains a Queue, in which all these pending print requests are stored. For each print request, the *system_id*, *filename*, *numPages*, and *size* (in KB) are stored in the queue. Use singly linked list for implementing the print queue *Q*. Write functions ADDREQUEST() for adding a new request, EXTRACTNEXTREQUEST() for extracting the next request to be printed, and LISTREQUESTS() for listing the contents of the print queue (define the functions with appropriate arguments).

   Input format:

   - Each line contains a character from '*a*', '*e*' , '*l*' or '*t*' followed by the arguments required for the corresponding operation.

- Character 'a' is followed by two strings $s1$, $s2$, an integer n and a float value separated by space. In this operation, a new print request with the string $s1$ as *system_id*, $s2$ as *filename*, n as *numPages* and the float value as *size* (in KB) is inserted to the tail of Q.
- Character 'e' is to delete the first print request in Q and the deleted print request's *system_id* is printed.
- Character 'l' is to list the contents of print queue Q.
- Character 't' is to 'terminate' the program.

Output Format:

- The output (if any) of each command should be printed on a separate line.
- For option 'e' , delete the first print request in Q and print the *system_id* of deleted print request. If Q is empty, then print -1.
- For option 'l' , list the contents of print queue Q. If Q is empty, then print -1.

Sample Input

```
a f01898 abc 5 147.1
a f01897 xyz 7 190.2
e
l
e
e
l
t
```

Sample Output

```
f01898
f01897 xyz 7 190.2
f01897
-1
-1
```