National Institute of Technology, Calicut
Department of Computer Science and Engineering
CS2094 – Data Structures Lab
Assignment-3

Submission deadline (on or before):

21st February 2016, 10:00:00 PM (for both Main and Advanced batches)

Policies for Submission and Evaluation

You must submit your assignment in the moodle (Eduserver) course page, on or before the submission deadline. Also, ensure that your programs in the assignment must compile and execute without errors in Athena server. During evaluation your uploaded programs will be checked in Athena server only. Failure to execute programs in the assignment without compilation errors may lead to zero marks for that program.

Your submission will also be tested for plagiarism, by automated tools. In case your code fails to pass the test, you will be straight away awarded zero marks for this assignment and considered by the examiner for awarding F grade in the course. Detection of ANY malpractice regarding the lab course will also lead to awarding an F grade.

Naming Conventions for Submission

Submit a single ZIP (.zip) file (do not submit in any other archived formats like .rar or .tar.gz). The name of this file must be ASSG<NUMBER>_<ROLLNO>_<FIRST-NAME>.zip (For example: ASSG3_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>.<extension> (For example: ASSG3_BxxyyyyCS_LAXMAN_1.c). If there is a part *a* and a part *b* for a particular question, then name the source files for each part separately as in ASSG3_BxxyyyyCS_LAXMAN_1b.c.

If you do not conform to the above naming conventions, your submission might not be recognized by some automated tools, and hence will lead to a score of 0 for the submission. So, make sure that you follow the naming conventions.

Standard of Conduct

Violations 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.pdf.

Assignment Questions

General Instructions for all the questions:

- Invalid input should be detected and suitable error messages should be generated.
- Sample inputs are just indicative.

1. Write a function **void swap(struct node* head, int x, int y)** with the following specification: It takes three arguments – **head** points to the first node of a singly linked list containing integers, **x** and **y** are the values of two nodes in the singly linked list. **swap** function swaps the node whose value is **x** with the node whose value is **y** (by rearranging the links). Also implement a **main()** function which creates the singly linked list by reading its contents. The function **main()** should also read the values of **x** and **y** from the console; then calls the function **swap** and finally prints the modified linked list.

Input format:

The first line of input contains a positive integer **n**, the size of the linked list.
The second line contains **n** space separated integers in the range -2^{30} to 2^{30}, the contents of the linked list.
The third line contains two integers **x** and **y** in the range -2^{30} to 2^{30}

Output format:

n space separated integers, representing the modified linked list.

Example:

a. Input:

```
10
76 29 46 7 91 836 47 92 724 512
46 836
```

Output:

```
76 29 836 7 91 46 47 92 724 512
```

b. Input:

```
10
76 29 46 7 91 836 47 92 724 512
48 836
```

Output:

```
48 not found
```

2. Write a function **int k_last(struct node* head, int k)** with the following specification: It takes two arguments – the **head** of a singly linked list containing integers in the range -2^{30} to 2^{30} and a positive integer **k**; and returns the content of the **kth** node from the end of the linked list in *linear time*. Also implement a **main()** function which creates the singly linked list by reading its contents. The function **main()** should also read the value of **k** from the console; then calls the function **k_last** and finally prints the value returned by this function.

Input format:

The first line of input contains a positive integer **n**, the size of the linked list.
The second line contains **n** space separated integers in the range -2^{30} to 2^{30}, the contents of the linked list.
The third line contains a single positive integer **k**, 1 <= **k** <= **n**.

Output format:

Single integer denoting the content of **k***th* node from the end of the linked list

Example:

a. Input:

```
10
76 29 46 7 -91 836 47 92 724 512
3
```

Output:

```
92
```

b. Input:

```
10
76 29 46 7 -91 836 47 92 724 512
12
```

Output:

```
Invalid Input: 12 > 10
```

3. Write a program to implement a queue using the following:

a) Array representation

b) Linked list representation

Your program must support the following functions:

- **enqueue(q, element) -** Puts the data specified by **element** at the rear end of the queue **q.**
- **dequeue(q)** - removes and returns the element at the front of the queue **q**. Return EMPTY, if the queue is empty.
- **print_queue(q)** - displays the values of all the elements present in the queue **q**.

Input format:

The input consists of multiple lines, each line of the input contains a character from {'e', 'd', 'p', 's'} followed by zero or one integer. The integer, if given, is in the range 0 to 2^{31}.

- The character 'e' means enqueue the next integer from the input into the queue. In this case, the next integer (>= 0) is given on the same line as the character 'e', separated by a space.
- The character 'd' means dequeue and output element from the front of the queue. Output “EMPTY”, if the queue was originally empty.
- The character 'p' means show all elements in the queue. In this case, output all elements of the queue on a single line, separated by space, starting with the element at the front. Output “EMPTY”, if the queue was originally empty.
- The character 's' means “stop the program”

Output format:

- The output (if any) of each command should be printed on a separate line.

Example:

a) Array Representation: For the array implementation, the very first line of input contains an integer value c, 0 < c < 100, which is the capacity of the queue. In this case, the enqueue operation must output “OVERFLOW” when an element is being enqueued into an already full queue.

Input:

```
4
e 2
e 5
e 7
d
p
e 10
e 11
d
p
d
d
```

```
d
e 3
p
s
```

Output:

```
2
5 7
OVERFLOW
5
7 10
7
10
EMPTY
3
```

b) Linked list representation

Input:

```
e 2
e 5
e 7
d
p
e 10
d
p
d
d
d
e 3
p
s
```

Output:

```
2
5 7
5
7 10
7
10
EMPTY
3
```

4. Write a program to implement a stack using the following:

a) Array representation

b) Linked list representation

Your program must support the following functions:

- **push(stk, element)** - puts the data specified by **element** on top of the stack **stk**.
- **pop(stk)** - removes and returns the topmost element of the stack **stk.** Return EMPTY, if the stack is empty.
- **print_stack(stk) -** displays all the data present in the stack **stk**.

Input format:

The input consists of multiple lines, each line in the input contains a character from {'a', 'r', 'p','s'} followed by zero or one integer. The integer, if given, is in the range 0 to 2^{31}.

- The character 'a' means push the next integer from the input on the stack. In this case, the next integer (>= 0) is given on the same line as the character 'a', separated by a space.

- The character 'r' means means pop and output the topmost element of the stack. Output "EMPTY", if the stack was originally empty.
- The character 'p' means show all elements in the stack. In this case, output all elements of the stack on a single line, separated by space, starting with the top most element. Output "EMPTY", if the stack was originally empty.
- The character 's' means "stop the program".

Output format:

- The output (if any) of each command should be printed on a separate line.

Example:

a) Array representation: For the array implementation, the first line of the input contains an integer value c, 0 < c < 1000, which is the capacity of the stack. In this case, the push operation must output "OVERFLOW" when an element is being pushed into an already full stack.

Input:

```
3
a 2
a 5
a 7
r
p
a 10
a 11
r
p
r
r
r
a 3
p
s
```

Output:

```
7
5 2
OVERFLOW
10
5 2
5
2
EMPTY
3
```

b) Linked list representation

Input:

```
a 2
a 5
a 7
r
p
a 10
r
p
r
r
r
a 3
```

```
                p
                s
Output:
                7
                5 2
                10
                5 2
                5
                2
                EMPTY
                3
```

5. Write a program to implement a priority queue using heap. Your program must support the following functions:

insert(pq, elem, pr) - inserts the data specified by element **elem** into the priority queue **pq** with priority **pr**.
extract_min(pq) - removes and prints the element with the lowest priority from the priority queue **pq**; prints “EMPTY” if the priority queue is empty.
get_min(pq) - prints the element with the lowest priority from the priority queue **pq**, without actually removing it from the priority queue; prints “EMPTY” if the priority queue is empty.
decrease_priority(pq, elem, newpr) - changes the priority of element **elem** in the priority queue specified by **pq**, by assigning it the new priority **newpr**. It is guaranteed that **elem** will be present in the priority queue, and that **newpr** will be lower (in the sense of priority) than the original priority of **elem**.

Input format:
The input consists of multiple lines, each line of the input contains a character from {'a', 'e', 'g', 'd', 's'} followed by zero or two integers. The integers, if given, are in the range -2^{30} to 2^{30}.

- The character 'a' means insert the next integer from the input into the priority queue. In this case, two more integers follow the character 'a', each separated by space. The first integer (>= 0) is the data to be inserted. The second integer (>= 1) is the priority of this data item (1 being the lowest priority). Assign the priority to the data, and then insert this data into the priority queue.
- The character 'e' means remove and output the element with the lowest priority from the priority queue (Output the priority of the element in parenthesis, separated by a space). Output “EMPTY”, if the priority queue was originally empty.

- The character 'g' means output the element with the lowest priority from the priority queue without actually removing it (Output the priority of the element in parenthesis, separated by a space). Output “EMPTY”, if the priority queue was originally empty.

- The character 'd' means decrease the priority of a datum in the priority queue. In this case, two more integers follow 'd', each separated by space. The first integer (>= 0) is the data (which will be present in the priority queue). The second integer (>= 1) is the new decreased priority. After this operation, the priority of this data should be the one specified in this input.

- The character 's' means “stop the program”.

Output format:

- The output (if any) of each command should be printed on a separate line.

Example:

Input:

```
a 2 100
a 5 110
a 7 95
e
e
a 10 120
e
a 6 30
g
d 10 25
g
e
e
e
a 1 30
a 2 15
e
s
```

Output:

```
7
2
5
6
10
10
6
EMPTY
2
```

****