University of Zurich^{UZH}

Department of Informatics
Database Technology Group
Prof. Dr. M. Böhlen

Informatics II
Exercise 7

March 30, 2020

Abstract Data Types (ADT)

Task 1. Create a new datatype `twoStack` of positive integers that represents two stacks. This datatype should use only one array i.e., both stacks should use the same array for storing elements. A stack is of the following type:

```
typedef struct stackADT {
    int *arr;
    int size;
    int top1, top2;
} twoStack;
```

Your implementation should start two stacks from two extreme corners of arr[]. The `stack1` starts from the leftmost element, the first element in `stack1` is pushed at index 0. The `stack2` starts from the rightmost corner, the first element in `stack2` is pushed at index (size-1). Both stacks grow and shrink in opposite direction.

Along with the above datatype create the following functions:

- `void initialize(twoStack *s, int n)` which initializes stack `s`.
- `void push1(twoStack *s, int value)` which inserts element `value` into first stack.
- `void push2(twoStack *s, int value)` which inserts element `value` into second stack.
- `int pop1(twoStack *s)` which removes an element from first stack and returns its value. In case of an error, -1 should be returned.
- `int pop2(twoStack *s)` which removes an element from second stack and returns its value. In case of an error, -1 should be returned.
- `void printStack(twoStack *s)` that prints the values of the elements of the twoStack to the console enclosed in brackets, e.g. `[ 3 5 7 2 ]`.

Test your implementation by performing the following operations:

1. Create and initialize stack s with array size of 5.
2. Insert 5 into first stack
3. Insert 10, 15 into second stack
4. Insert 11 into first stack
5. Insert 7 into second stack
6. Print the elements on the console
7. Remove one element of first stack and print the value of the removed element.
8. Remove one element of second stack and print the value of the removed element.
9. Insert 27 into first stack
10. Insert 92 into the second stack
11. Print the elements on the console

Task 2. Consider a datatype `queue` corresponding to a circular queue of positive integers which be implemented using Circular linked list. A queue is of the following structure:

```
struct queue {
  struct node* head;
  struct node* tail;
};
```

```
struct node {
  int data;
  struct node* next;
};
```

Considering the `queue` datatype write the C code for the following functions:

- `struct queue* initialize()` which initializes the `head` and the `tail` nodes of `q`.
- `void enQueue(queue *q, int value)` which inserts the element `value` in `q`.
- `int deQueue(queue *q)` which removes an element from `q` and returns its value. In case of an error, -1 should be returned.
- `void displayQueue(struct queue *q)` that prints the values of the elements of the cicular queue to the console enclosed in brackets, e.g. `[ 3 5 7 2 ]`.

After you have implemented and tested these functions with lists of your choice, include the following sequence in your *main()* method:

1. Insert 14, 22, 6 in the queue `q`.

2. Print all the elemets of queue `q`.
3. Dequeue two elements and print their values.
4. Print `q`.
5. Enqueue 9, 20 in the queue `q`
6. Print `q`