**UCS 2312 Data Structures Lab**

**Assignment 3: StackADT and its application**

**Date of Assignment: 21.10.2022**

Create an ADT for the stack data structure with the following functions. stack*ADT* will have the integer array, top and size. [CO1, K3]

1. createStack(top) – initialize size and top with -1
2. push(top,data) – push data into the stack if stack is not full. Print a message when stack is

full

1. pop(top) – decrements the top by 1
2. getTop(top)– returns the element at top, if stack is not empty, otherwise returns -1
3. isEmpty(top) – returns 1 if stack empty, otherwise returns 0
4. isFull(top) – returns 1 if stack full, otherwise returns 0

Test the operations of stackADT with the following test cases

|  |  |
| --- | --- |
| **Operation** | **Expected Output** |
| getTop(top) | Empty |
| push(top,2) | 2 |
| push(top,4) | 4, 2 |
| push(top,6) | 6, 4, 2 |
| push(top,8) | Full |
| pop(top) |  |
| getTop(top) | 4 |
| getTop(top) | 4 |
| pop(top) |  |
| pop(top) |  |
| getTop(top) | Empty |
| pop(top) |  |
| pop(top) |  |
| push(top,11) | 11 |
| getTop(top) | 11 |

Best practices to be followed:

* Design before coding
* Usage of algorithm notation
* Use of multi-file C program
* Versioning of code

Application using Stack

1. Evaluate the postfix expression using Stack

Example: 23+45+\*

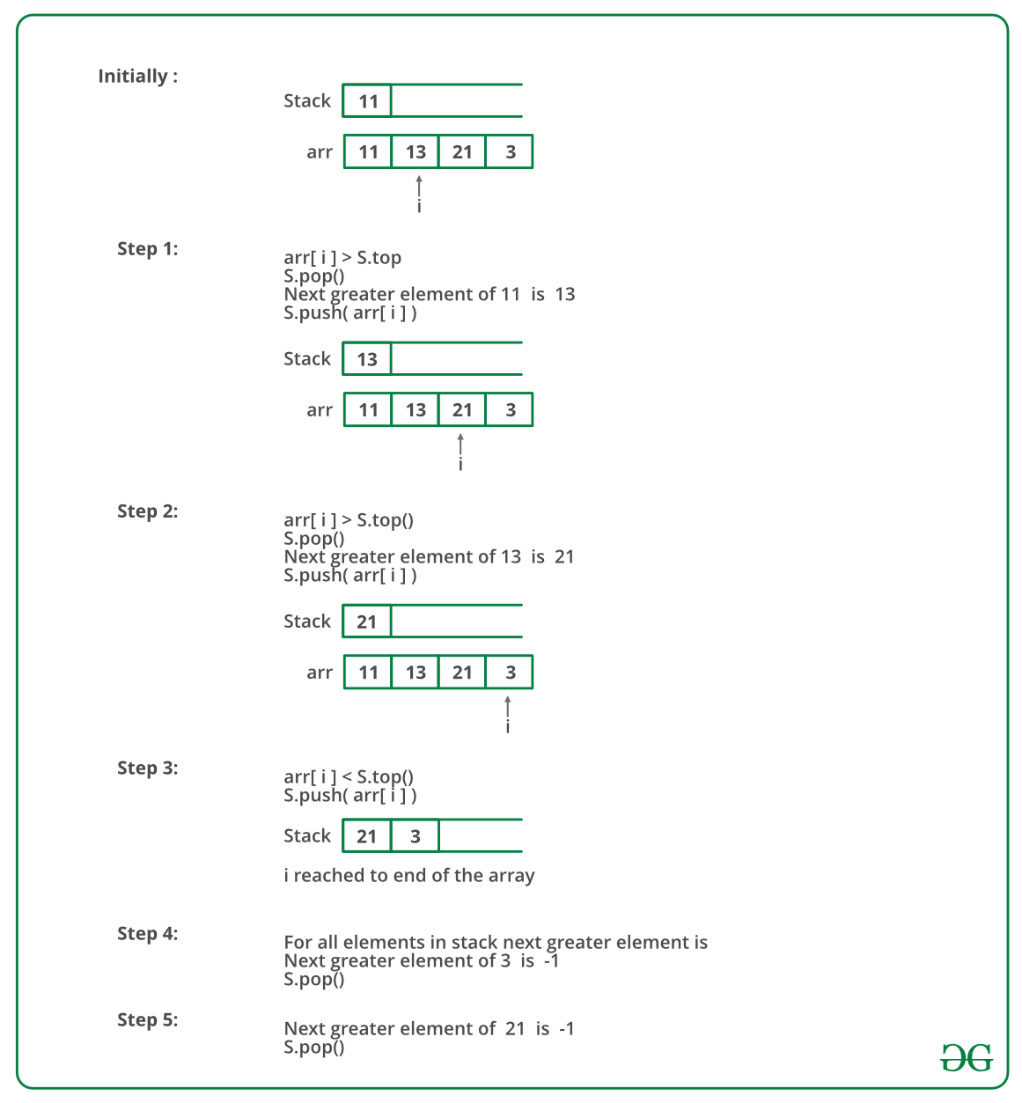
Ans: 45

1. Convert the given decimal number into binary using stack

Example: 14

Ans: 1110

1. Find next greater element using stack



Follow the steps mentioned below to implement the idea:

* Push the first element to stack.
* Pick the rest of the elements one by one and follow the following steps in the loop.
  + Mark the current element as next.
  + If the stack is not empty, compare top most element of stack with next.
  + If next is greater than the top element, Pop element from the stack. next is the next greater element for the popped element.
  + Keep popping from the stack while the popped element is smaller than next. next becomes the next greater element for all such popped elements.
* Finally, push the next in the stack.
* After the loop in step 2 is over, pop all the elements from the stack and print -1 as the next element for them.