University of Salahaddin-Hawler
College of Engineering
Software and Informatics Engineering Department
Second Year Class

Data Structure and Algorithm Design

Lecture 5

Lecturer

Kanar Shukr Muhamad

2023-2024

Stack

- A stack is a linear data structure in which the insertion of a new element and removal of an existing element takes place at the same end represented as the top of the stack.
- To implement the stack, it is required to maintain the pointer to the top of the stack.
- A stack data structure could use an array, a linked list, or any thing that can hold data, stores arbitrary objects.

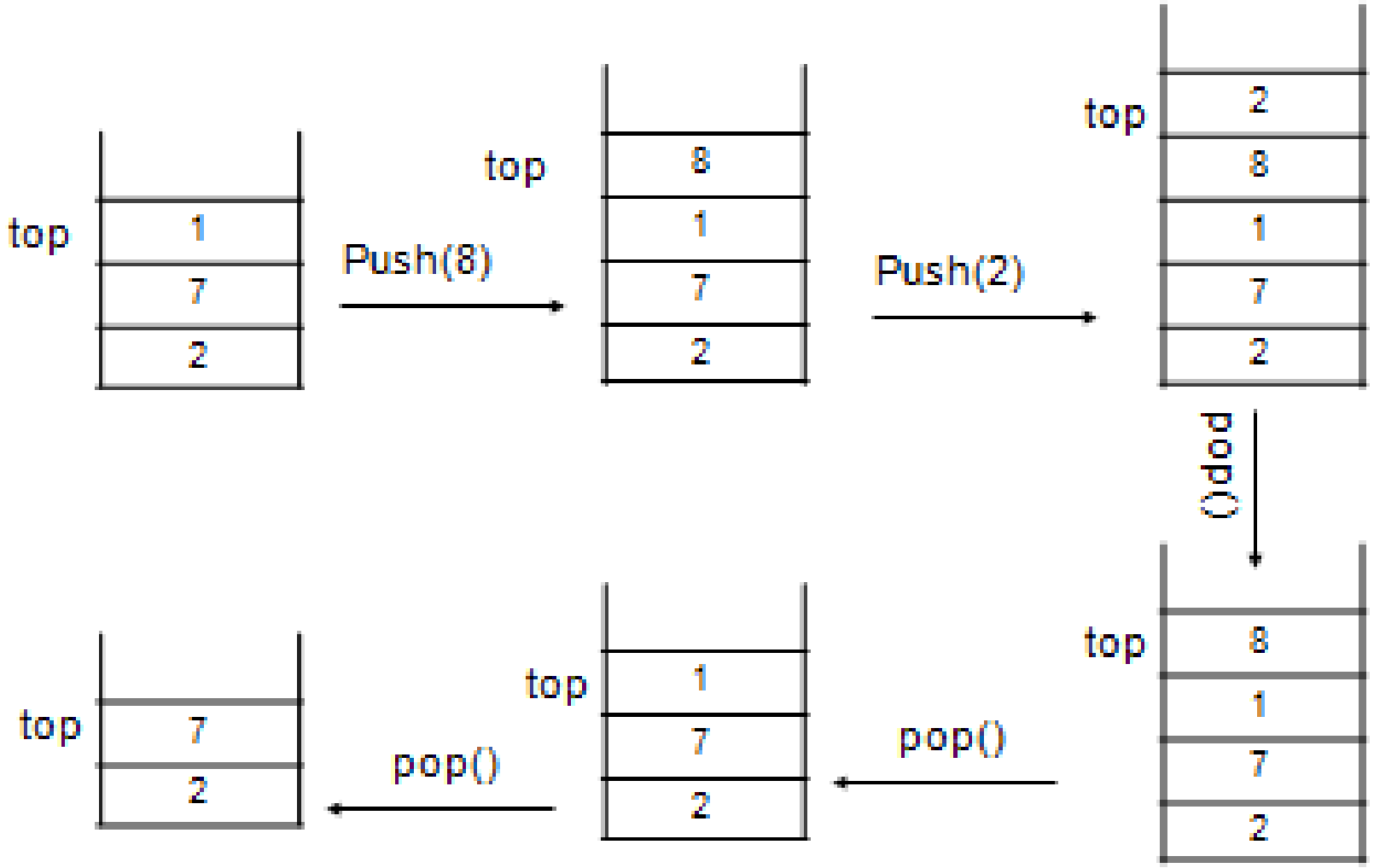
Stack Continue

- To add(push) an item to the stack, it must be placed on the top of the stack.
- To remove(pop) an item from the stack, it must be removed from the top of the stack too.
- Thus, the last element that is pushed in to the stack, is the first element to be popped out from the stack. i.e. ,Last In First Out(LIFO)

Stack

- It supports the following main operations:
- **push(object o):** inserts element o.
- **pop():** removes and returns the last inserted element.
- **isEmpty():** returns a Boolean value indicating whether no elements are stored.
- **isFull():** checks weather stack is full or not.

Stack Continue



Stack Applications

- Reversing a sequence of string.
- Checking palindrome sequences.
- Store the return address when use the CALL interrupt.
- Converting expression from infix to prefix or prefix to infix and evaluation the expression which found in any form above.
- Page-visited history in a Web browser
- Undo sequence in a text editor

Exception

- Attempting the execution of an operation of ADT may some times cause an error condition, called an exception.
- Exceptions are said to be “thrown” by an operation that can not be executed.
- In the Stack ADT, operations pop can not be performed if the stack is empty.
- Attempting the execution of pop on an empty stack throws an **EmptyStackException**

Stack Algorithm

- Stack is an order collection of items, it may be declared as array or structure containing two objects.
- As array we can define the stack as follows:

```
int stack[max_size_of_stack];
int top;
```

- Note that stack variable can be of any data type (integer, float, etc.) while top must be of type integer always.

Stack Algorithms Continue

- **1. PUSH algorithm:**

  Adds an item to the stack. If the stack is full, then it is said to be an overflow condition.

```
push(value)
begin
   if stack is full
        return
   end if
   else
        increment top
        assign value to stack[top]
   end else
end algorithm
```

Stack Algorithms Continue

2. POP algorithm: Return and remove an item from the stack. The items are popped in the reversed order in which they are pushed.

If the stack is empty, then it is said to be an underflow condition.

Stack Algorithms Continue

```
pop()
begin
      if stack is empty
            return
      end if
      else
            save a value of stack[top]
            decrement top
       end else
      return saved value
end algorithm
```

Stack Algorithms Continue

- **3. Stack empty algorithm:**

Returns true if the stack is empty, else false.

```
isEmpty()
begin
        if top < 0
                return true
        else
                return false
end algorithm
```

Stack Algorithms Continue

4.Stack full algorithm: This algorithm is use to check if stack is full or not, it is called from push algorithm.

```
isFull()
begin
      if top =maxsize-1
            return true
     else
           return false
 end algorithm
```

Limitations

- **Limitations**
  - The maximum size of the stack must be defined a priori, and can not be changed.

Convert Infix to Postfix

1. **Infix:** A+B Here arithmetic operator in the middle.
2. **Postfix:** AB+ Here arithmetic operator at the end.
3. **Prefix:** +AB Here arithmetic operator at the begin.

- In computer the compiler covert infix expression top postfix expression by using a stack.

Convert Infix to Postfix

- **Algorithm of converting infix expression to postfix expression by using single stack.**

1. We use single stack to store the operator signal.
2. Check arithmetic expression character by character from left to right.
3. Character may be:
    1. If character is operand (a to z, 1 to 9) then output it into output string.

Convert Infix to Postfix Continue

2. If character is left parenthesis “(“ then push in stack
3. if character is operator (+,-,*,/) then pop all operators from stack that have priority greater or equal than new operator to the output string then push new operator to the stack.
4. if character is right parenthesis “)“, pop all operators from stack until left bracket to the output string , ignore left and right parentheses do not add to the output string.`

Convert Infix to Postfix Continue

4. If you finished the arithmetic expression then popped all operators from stack to the output string. The final shape of output string is postfix expression.

Convert Infix to Postfix Continue

- **The order of precedence:**
  - Exponentiation ^
  - /, *
  - -, +
  - Equivalent(=,<,<=,>,>=.!=)
  - NOT
  - AND,%
  - OR

Convert Infix to Postfix Continue

- **E.x/** Convert infix to postfix: (6 – 2) * (5+4)

Step No.	I/P	Stack	O/P String
1	(	(	
2	6	(	6
3	-	(-	6
4	2	(-	6 2
5	)	null	6 2 -
6	*	*	6 2 -
7	(	* (	6 2 -
8	5	* (	6 2 - 5
9	+	* (+	6 2 - 5
10	4	* (+	6 2 – 5 4
11	)	*	6 2 – 5 4 +
12	null	null	6 2 – 5 4 + *