Stack

Lesson 5.1

Learning Objectives

LO 5.1.1 **Enumerate** and **define** the basic operations of a stack ADT

LO 5.1.2 **Compute** the asymptotic complexities of stack operations

LO 5.1.3 **Assert** solutions on computing problems involving stack ADT

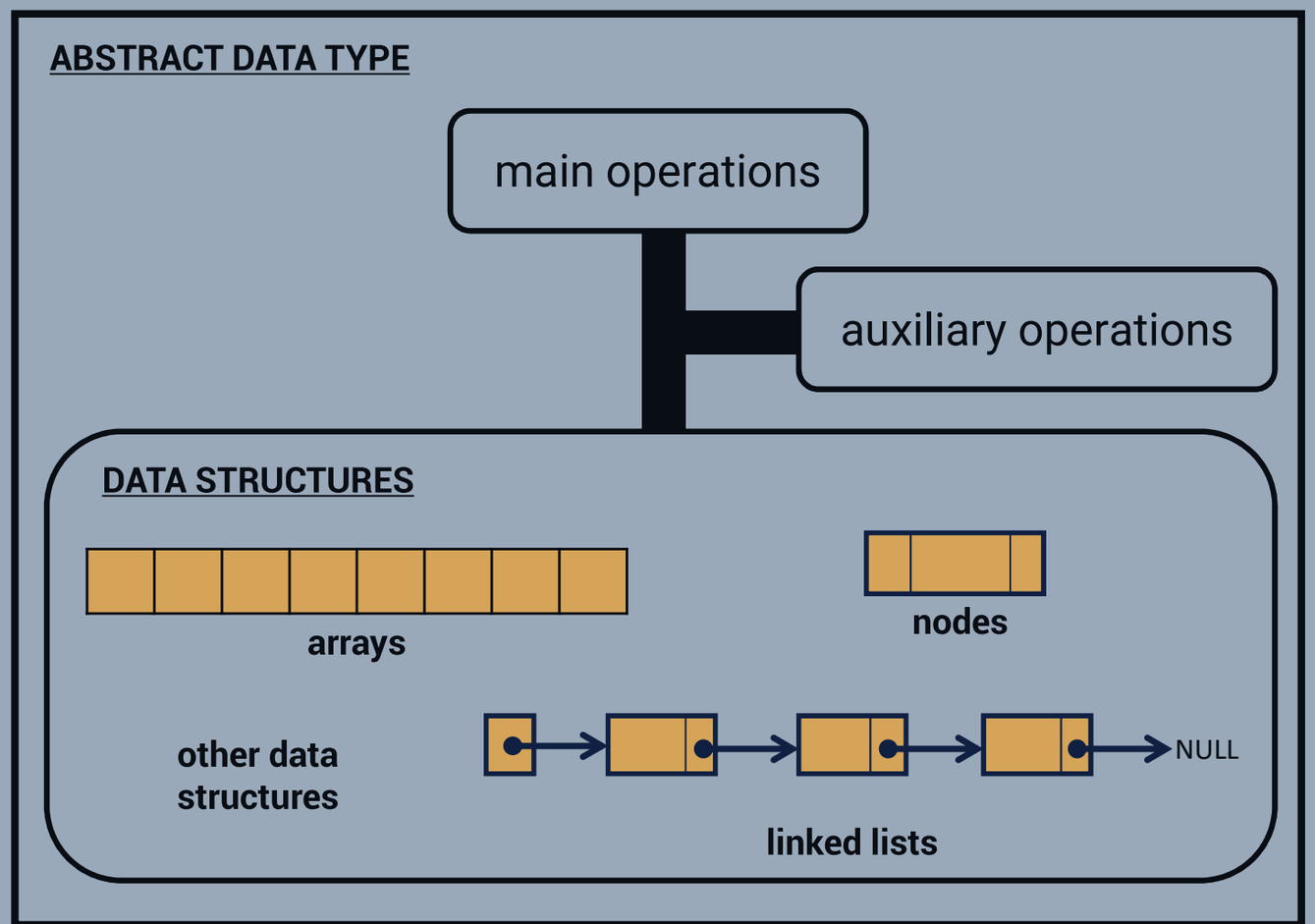
Abstract Data Types

- An **abstract data type** (ADT) is a mathematical model for data types and structures.
- It is defined by its functions and the type of data it is used for.

Data Structures vs. ADTs

- *Data structure is* the actual representation of the data during implementation, while, *abstract data type* is the logical description of data that is managed through its operations.
- In simple terms, **data structure** represents how each data is stored in the memory while **ADTs** uses *data structures* to manage the data through a defined set of operations.

Data Structures vs. ADTs

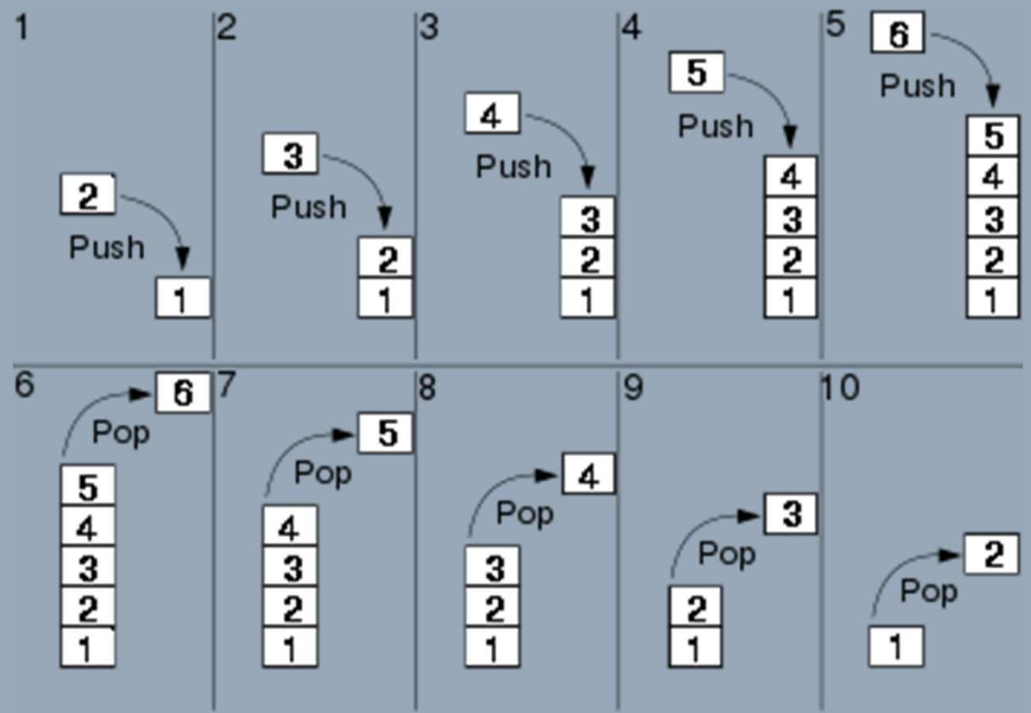


Stack

- Stack is an ADT that follows the principle **Last In, First Out** (*LIFO*).
- It is *linear* in form thus it can either be implemented using either an array or any type of linked list.
- A stack *may* be implemented to have a *bounded capacity*. If the stack is full and does not contain enough space to accept another element, the stack is in a state of **stack overflow**.

Stack Operations

- The main operations on a stack are the *push* and *pop*
- **Push** *inserts* the element at the *top* of the stack
- **Pop** *removes/deletes* the element at the *top* of the stack



Stack Operations

- Some of the auxiliary operations of the stack include; but not limited to,
  - **Peek** return the element at the top of the stack
  - **IsEmpty** returns 1 if the stack is empty and 0 otherwise
  - **IsFull** returns 1 if the stack is full and 0 otherwise

Applications of Stack

- Expression evaluation (*mathematical, logical, etc.*)
- Syntax parsing (*language*)
- Backtracking algorithms
- Memory management
- Undo/Redo features of desktop applications

Strengthening the Learning Objectives

LO 5.1.1 Enumerate and define the basic operations of a stack ADT

What are the basic operations of a stack ADT and what do they do?

LO 5.1.2 Compute the asymptotic complexities of stack operations

Construct a stack ADT structure in 2 implementations: using arrays and circular linked lists. Compute and compare the asymptotic complexity of all five (5) operations based on their most efficient implementations.

LO 5.1.3 Assert solutions on computing problems involving stack ADT

Create a program that will evaluate a given postfix expression using stack.

Example:

Input	8 2 3 ^ / 2 3 * + 5 1 * -
Output	2