Programming Paradigms Tutorials
Abstract Data type

Abstract Data type

Abstract Data type (ADT) is a type (or class) for objects whose behavior is defined by a set of value and a set of operations.

The definition of ADT only mentions what operations are to be performed but not how these operations will be implemented. It does not specify how data will be organized in memory and what algorithms will be used for implementing the operations. It is called "abstract" because it gives an implementation-independent view. The process of providing only the essentials and hiding the details is known as abstraction.

The user of data type does not need to know how that data type is implemented, for example, we have been using Primitive values like int, float, char data types only with the knowledge that these data type can operate and be performed on without any idea of how they are implemented. So a user only needs to know what a data type can do, but not how it will be implemented. Think of ADT as a black box which hides the inner structure and design of the data type (check Fig 1).
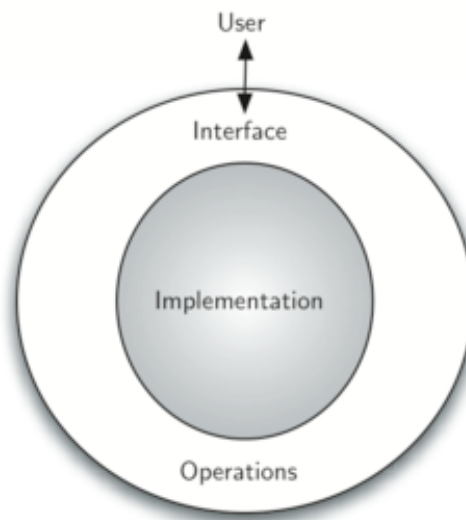

Fig 1. Overview of ADT concept.

Some examples of ADT are Stack, Queue, List etc. Let us see some operations of those mentioned ADT:

Stack:
- isFull(), This is used to check whether stack is full or not
- isEmpry(), This is used to check whether stack is empty or not
- push(x), This is used to push x into the stack
- pop(), This is used to delete one element from top of the stack
- peek(), This is used to get the top most element of the stack
- size(), this function is used to get number of elements present into the stack

Queue
- isFull(), This is used to check whether queue is full or not
- isEmpry(), This is used to check whether queue is empty or not
- insert(x), This is used to add x into the queue at the rear end
- delete(), This is used to delete one element from the front end of the queue
- size(), this function is used to get number of elements present into the queue

List
- size(), this function is used to get number of elements present into the list
- insert(x), this function is used to insert one element into the list
- remove(x), this function is used to remove given element from the list
- get(i), this function is used to get element at position i
- replace(x, y), this function is used to replace x with y value

Exercises

1. Define ADT for Map.

2. Define ADT for Set

3. Compare implementation of ArrayList and LinkedList in Java