VTC Academy
Think Ahead, Beyond Boundaries

DATA STRUCTURES AND ALGORITHMS

LESSON 1

Introduction to Data Structures and Algorithms

CONTENT

VTC Academy
Think Ahead, Beyond Boundaries

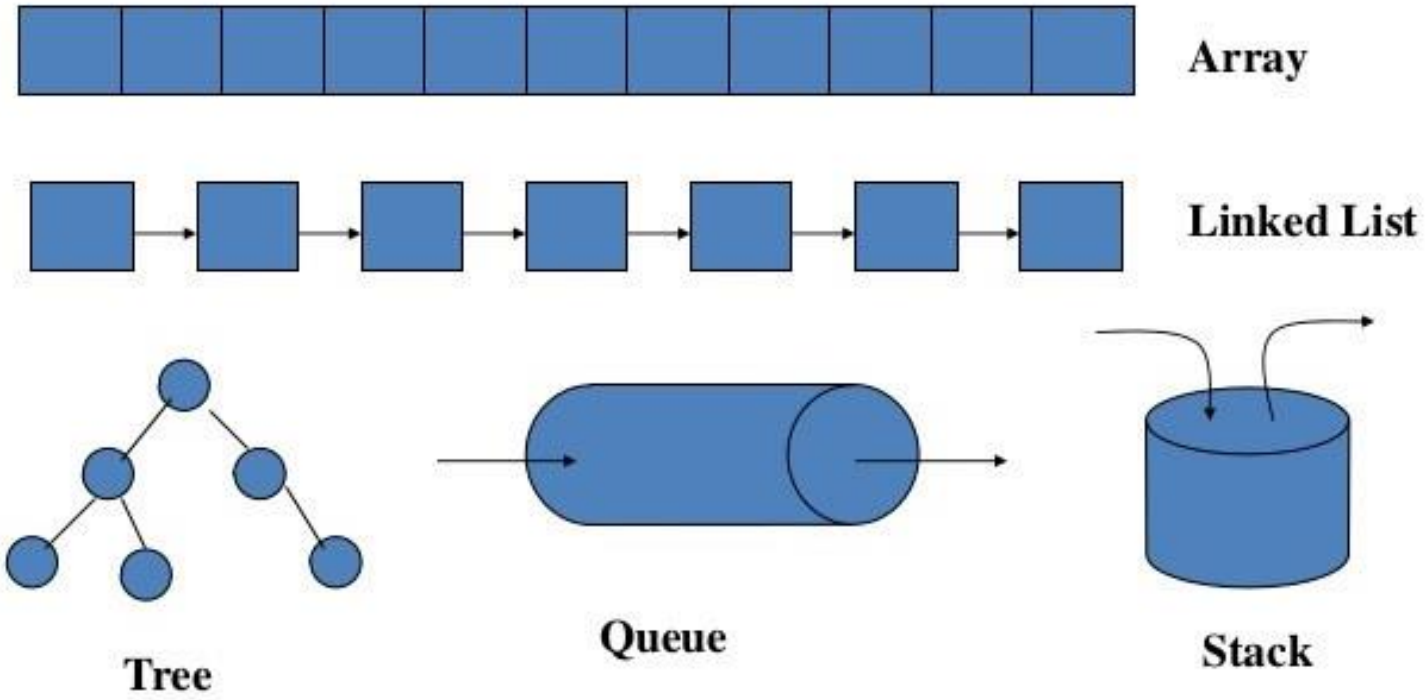
Introduction to Data Structures

VTC Academy
Think Ahead, Beyond Boundaries

- Data Structure is a way to store and organize data in a computer, so that it can be used efficiently.
- Almost every enterprise application uses various types of data structures in one or the other way.
- We talk about Data Structures as
  - Mathematical / Logic Model or Abstract Data Type
  - Implementation

Introduction to Data Structures

VTC Academy
Think Ahead, Beyond Boundaries



Abstract Data Type & Implementation

VTC Academy
Think Ahead, Beyond Boundaries

- Abstract Data Type
  - Each data structure has an Abstract Data Type.
  - Abstract Data Type represents the set of operations that a data structure supports.
  - An Abstract Data Type only provides the list of supported operations, type of parameters they can accept and return type of these operations.
- Implementation
  - Implementation provides the internal representation of a data structure.
  - Implementation also provides the definition of the algorithms used in the operations of the data structure.

Abstract Data Type & Implementation

VTC Academy
Think Ahead, Beyond Boundaries

Abstract Data Type	Implementation
List	Array
Store a given number of any type elements	int a[11]; //Declare an integer array
Read element by position	a[i]; //Read element by position i
Modify element at position	a[i] = 12; //Modify element at position

Characteristics of a Data Structure

VTC Academy
Think Ahead, Beyond Boundaries

- Correctness – Data structure implementation should implement its interface correctly.
- Time Complexity – Running time or the execution time of operations of data structure must be as small as possible.
- Space Complexity – Memory usage of a data structure operation should be as little as possible.

Need for Data Structure

VTC Academy
Think Ahead, Beyond Boundaries

- As applications are getting complex and data rich, there are three common problems that applications face now-a-days.
  - Data Search – If the application is to search an item, it has to search an item in 1 million(106) items every time slowing down the search.
  - Processor speed – Processor speed although being very high, falls limited if the data grows to billion records.
  - Multiple requests – As thousands of users can search data simultaneously on a web server, even the fast server fails while searching the data.
- Data can be organized in a data structure in such a way that all items may not be required to be searched, and the required data can be searched almost instantly

Execution Time Cases

VTC Academy
Think Ahead, Beyond Boundaries

- Worst Case – This is the scenario where a particular data structure operation takes maximum time it can take (operation time > $f(n)$).
- Average Case – This is the scenario depicting the average execution time of an operation of a data structure. If an operation takes $f(n)$ time in execution, then m operations will take $mf(n)$ time.
- Best Case – This is the scenario depicting the least possible execution time of an operation of a data structure. If an operation takes $f(n)$ time in execution, then the actual operation may take time as the random number which would be maximum as $f(n)$.

Basic Terminology

VTC Academy
Think Ahead, Beyond Boundaries

- Data are values or set of values.
- Data Item refers to single unit of values.
- Attribute and Entity – An entity is that which contains certain attributes or properties, which may be assigned values.
- Field is a single elementary unit of information representing an attribute of an entity.
- Record is a collection of field values of a given entity.
- File is a collection of records of the entities in a given entity set.

Types of Data Structure

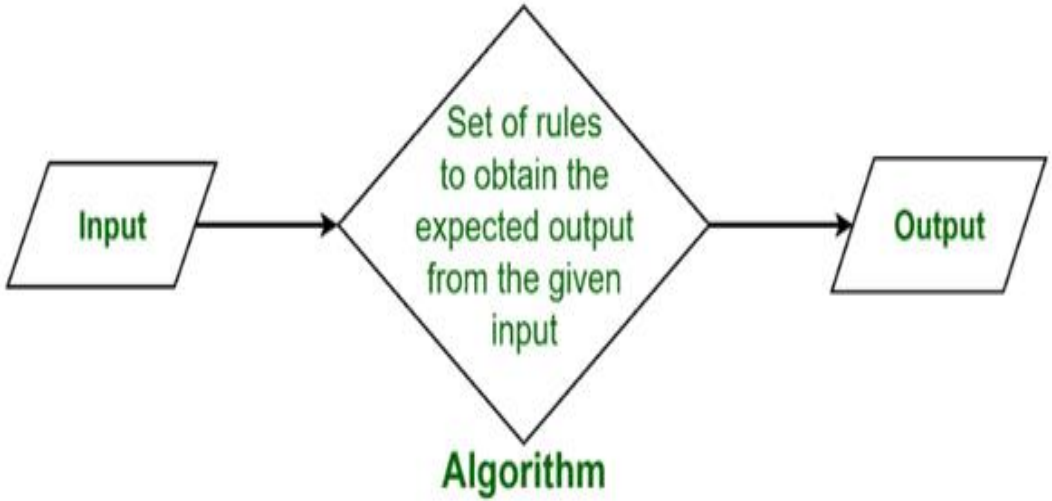
VTC Academy
Think Ahead, Beyond Boundaries

- Linear data structure
  - Array
  - Stack and Queue
  - Linked List
- Non-linear data structure
  - Tree
  - Graph

Introduction to Algorithms

VTC Academy
Think Ahead, Beyond Boundaries

- Algorithm is a step-by-step procedure, which defines a set of instructions to be executed in a certain order to get the desired output.
- Algorithms are generally created independent of underlying languages, i.e. an algorithm can be implemented in more than one programming language.



Introduction to Algorithms

VTC Academy
Think Ahead, Beyond Boundaries

- From the data structure point of view, following are some important categories of algorithms
  - Search – Algorithm to search an item in a data structure.
  - Sort – Algorithm to sort items in a certain order.
  - Insert – Algorithm to insert item in a data structure.
  - Update – Algorithm to update an existing item in a data structure.
  - Delete – Algorithm to delete an existing item from a data structure.

Characteristics of an Algorithm

VTC Academy
Think Ahead, Beyond Boundaries

- Unambiguous – Algorithm should be clear and unambiguous. Each of its steps (or phases), and their inputs/outputs should be clear and must lead to only one meaning.
- Input – An algorithm should have 0 or more well-defined inputs.
- Output – An algorithm should have 1 or more well-defined outputs, and should match the desired output.
- Finiteness – Algorithms must terminate after a finite number of steps.
- Feasibility – Should be feasible with the available resources.
- Independent – An algorithm should have step-by-step directions, which should be independent of any programming code.

How to Write an Algorithm?

VTC Academy
Think Ahead, Beyond Boundaries

- There are no well-defined standards for writing algorithms. Rather, it is problem and resource dependent. Algorithms are never written to support a particular programming code.
- As we know that all programming languages share basic code constructs like loops (do, for, while), flow-control (if-else), etc. These common constructs can be used to write an algorithm.
- We write algorithms in a step-by-step manner, but it is not always the case. Algorithm writing is a process and is executed after the problem domain is well-defined. That is, we should know the problem domain, for which we are designing a solution.

How to Write an Algorithm?

VTC Academy
Think Ahead, Beyond Boundaries

Use Flowchart Write an Algorithm – Sum 2 Numbers

Algorithm Analysis

VTC Academy
Think Ahead, Beyond Boundaries

- Efficiency of an algorithm can be analyzed at two different stages, before implementation and after implementation. They are the following:
  - A Priori Analysis – This is a theoretical analysis of an algorithm. Efficiency of an algorithm is measured by assuming that all other factors, for example, processor speed, are constant and have no effect on the implementation.
  - A Posterior Analysis – This is an empirical analysis of an algorithm. The selected algorithm is implemented using programming language. This is then executed on target computer machine. In this analysis, actual statistics like running time and space required, are collected.

Algorithm Complexity

VTC Academy
Think Ahead, Beyond Boundaries

- Suppose X is an algorithm and n is the size of input data, the time and space used by the algorithm X are the two main factors, which decide the efficiency of X.
  - Time Factor – Time is measured by counting the number of key operations such as comparisons in the sorting algorithm.
  - Space Factor – Space is measured by counting the maximum memory space required by the algorithm.
- The complexity of an algorithm f(n) gives the running time and/or the storage space required by the algorithm in terms of n as the size of input data.

Space Complexity

VTC Academy
Think Ahead, Beyond Boundaries

- Space complexity of an algorithm represents the amount of memory space required by the algorithm in its life cycle. The space required by an algorithm is equal to the sum of the following two components
  - A fixed part that is a space required to store certain data and variables, that are independent of the size of the problem. For example, simple variables and constants used, program size, ...
  - A variable part is a space required by variables, whose size depends on the size of the problem. For example, dynamic memory allocation, recursion stack space, ...
- Space complexity S(P) of any algorithm P is S(P) = C + SP(I), where C is the fixed part and S(I) is the variable part of the algorithm, which depends on instance characteristic I.

Time Complexity

VTC Academy
Think Ahead, Beyond Boundaries

- Time complexity of an algorithm represents the amount of time required by the algorithm to run to completion.
- Time requirements can be defined as a numerical function T(n), where T(n) can be measured as the number of steps, provided each step consumes constant time.

Recursive Algorithm

VTC Academy
Think Ahead, Beyond Boundaries

- A recursive algorithm is an algorithm which calls itself with "smaller (or simpler)" input values, and which obtains the result for the current input by applying simple operations to the returned value for the smaller (or simpler) input.
- More generally if a problem can be solved utilizing solutions to smaller versions of the same problem, and the smaller versions reduce to easily solvable cases, then one can use a recursive algorithm to solve that problem.

Applications of Recursive Algorithm

VTC Academy
Think Ahead, Beyond Boundaries

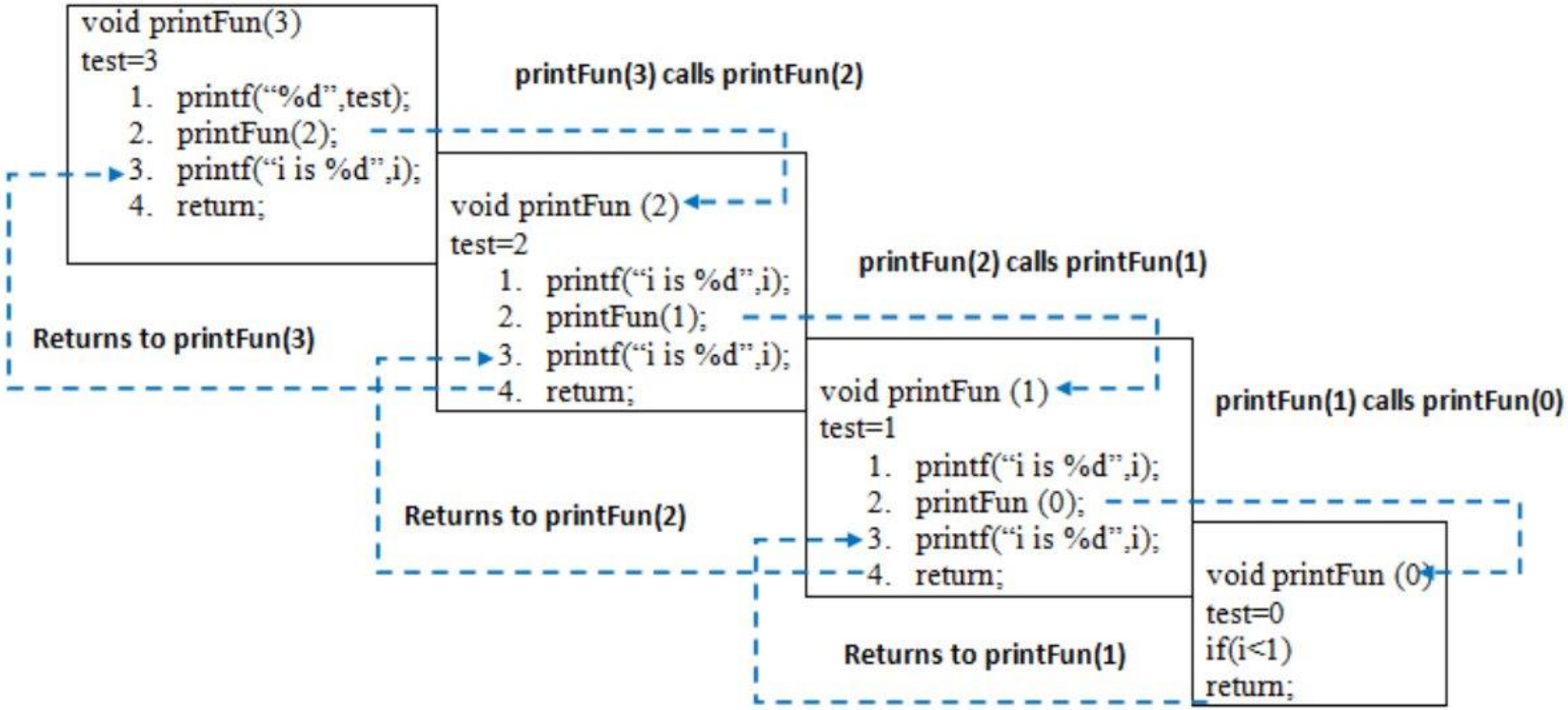
- Recursion can be used to replace loops in some problems such as printing list, calculating, or problem solving with divide-and-conquer methods.
- Recursion can be used in operations with binary tree.
- Recursive algorithms are used in popular searching and sorting algorithms such as:
  - Binary Search
  - Quick Sort
  - Heap Sort

Applications of Recursive Algorithm

VTC Academy
Think Ahead, Beyond Boundaries

- Take for example Candy Crush which uses them to generate combinations of tiles.

Recursive Algorithm



Fibonacci Series

VTC Academy
Think Ahead, Beyond Boundaries

- Fibonacci series generates the subsequent number by adding two previous numbers
- Fibonacci series starts from two numbers – F0 & F1. The initial values of F0 & F1 can be taken 0, 1 or 1, 1 respectively.
- Fibonacci series satisfies the following conditions
- Fn = Fn-1 + Fn-2

Fibonacci Series

VTC Academy
Think Ahead, Beyond Boundaries

```c
#include <stdio.h>

unsigned long fibonacci(unsigned int index);

int main(int count, char* args[]){
    unsigned int num;
    int i;
    printf("Input number of fibonacci series: ");
    scanf("%d", &num);
    printf("Fibonacci series: ");
    for(i=1; i<=num; i++){
        printf("%lu, ", fibonacci(i));
    }
    printf("\n");
    return 0;
}

unsigned long fibonacci(unsigned int index){
    if(index==1 || index==2){
        return 1;
    }
    return fibonacci(index-1) + fibonacci(index-2);
}
```

Summary

VTC Academy
Think Ahead, Beyond Boundaries

- Data Structures is a way to store and organize data in a computer, so that it can be used efficiently.
- Algorithm is a step-by-step procedure, which defines a set of instructions to be executed in a certain order to get the desired output.
- Algorithms are generally created independent of underlying languages (can be implemented in more than one programming language).
- A recursive algorithm is an algorithm which calls itself with "smaller (or simpler)" input values, and which obtains the result for the current input by applying simple operations to the returned value for the smaller (or simpler) input.

References

- Algorithms and Data Structures: An Approach in C - Charles F. Bowman
- Online courses:
  - https://www.cprogramming.com/algorithms-and-data-structures.html

VTC Academy
Think Ahead, Beyond Boundaries