Operative systems

University of Essex

Primitive data structure

- Integers
- Characters
- Floats
- Boolean
- Pointers - Holds memory address

Non-primitive data structures

- Arrays
- Lists
    - Stacks
    - Queues
- Files
- Trees

Data can be organized in two fashions:

- Linear - sequences and queues
- Non-linear - nodes and branches

Array

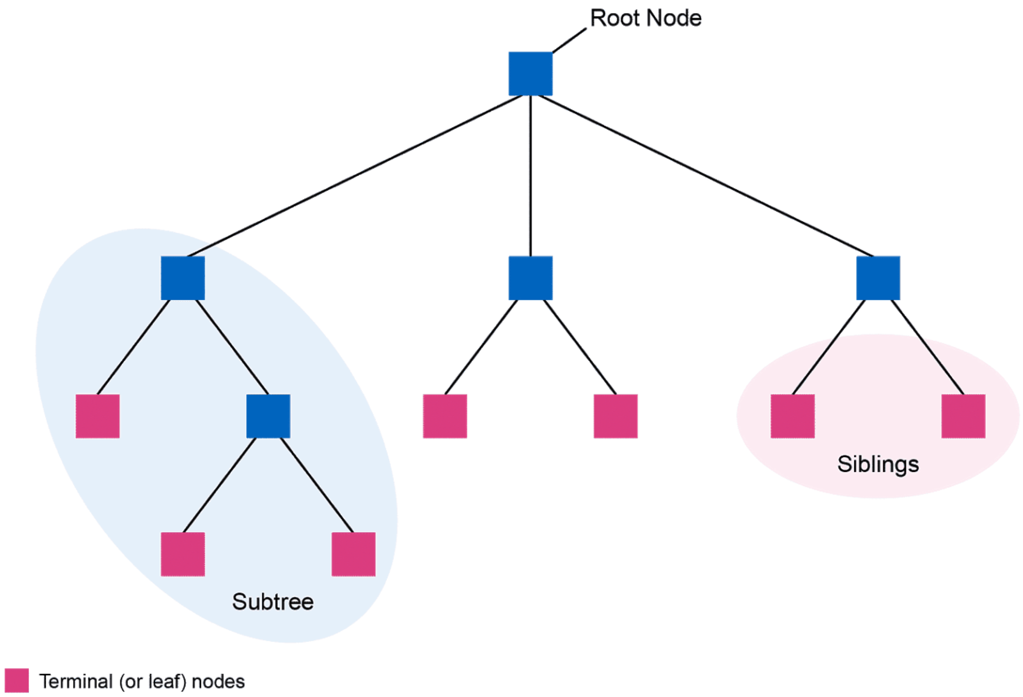
- Stores blocks of data which are of same type;
- Consists of rows and columns; and
- Indices are used to identify positions.

A two dimensional array is sometimes called a table, and other times it is referred to as a matrix.

Tree

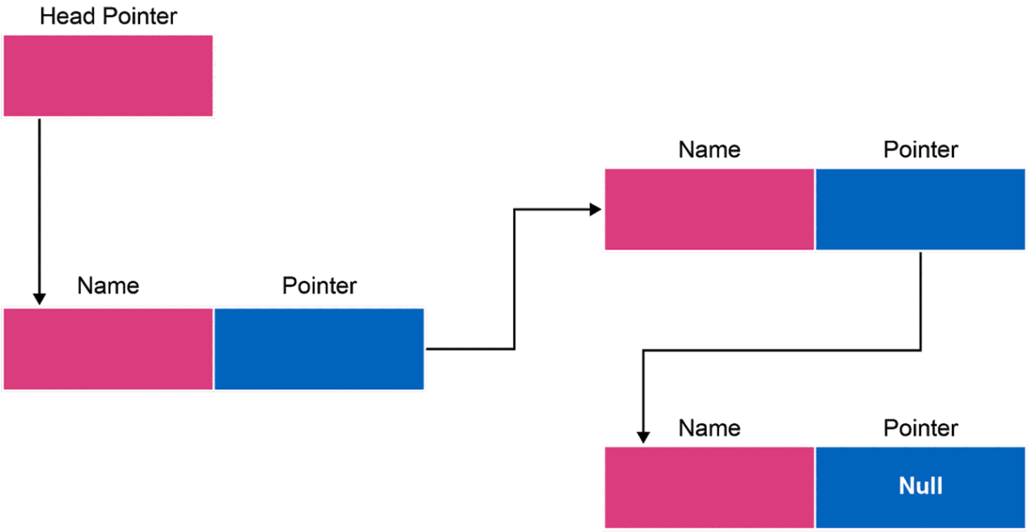
- A tree is non-linear data structure where data is not organised in a sequence.

- As this picture depicts, the data elements are arranged in nodes, branches and in sub-branches according to requirements. The structure is therefore in hierarchical order and these data elements are connected by edges.



Lists

- List
- Stack - is a data structure which performs insertion and deletion operations at one end only.
    - LIFO - Last in first out
- Queue - The Queue data structure allows for the insertion operation at one end and deletion operation on the other end.
    - FIFO - First in => First out
- Continuous list - A list in which entries are stored in an array.
- Linked list - A list in which entries are linked by pointers.
    - Head pointer - Head Pointer: Pointer to first entry in list.
    - Null pointer - A “non-pointer” value used to indicate end of list.

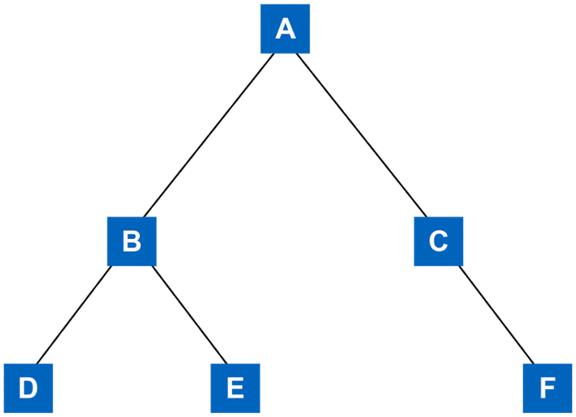


Brookshear, J. (2018)

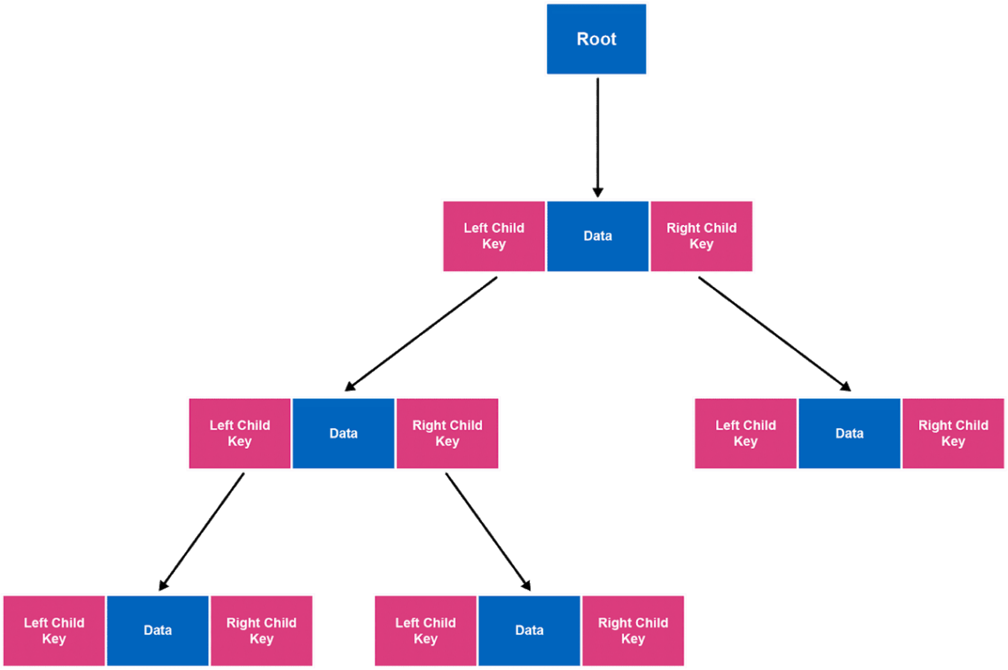
Binary Tree

A Binary Tree is a tree data structure with special conditions.

Each node in a binary tree contains a data cell, and a maximum of two child nodes (left and right). These child nodes are pointers to the left and right subtrees respectively.



Brookshear, J. (2018)



As you can see in this example, all the nodes to the left of G have keys (or values) smaller than G, while all the nodes to the right, have values greater than G. The same applies to the D and K parent nodes.

The left child of the parent will have keys of a smaller value than the parent, while the right child will have keys of a larger value than the parent.

Questions for knowledge practice

1. Why is the use of a constant considered better programming style than the use of a literal?
2. What is the difference between a declarative statement and an imperative statement?
3. List some common data types.
4. Identify some common control structures found in imperative and object-oriented programming languages.
5. What is the difference between an array and an aggregate type

Other simple data types

- Aggregate
- Lists

- Lists and list operations, such as add, remove, and search, are common in many programs.

Algorithms

The main task of an algorithm is to solve a well-specified problem by means of structured programming.

Programs & Algorithms

We often use the terms ‘program’ and ‘algorithm’ interchangeably, but there is a clear distinction between a program and an algorithm.

- Defines a problem and shows the steps as to how the problems can be solved by implementing an algorithm.
- Step by step approach of executing an algorithm.

Steps to solve a problem

1. Define the problem.
2. Outline an approach for solving the problem.
3. Execute the plan.
4. Examine the solution.
5. Evaluate the accuracy.
6. Continue the steps until you reach the most efficient solution.

Pseudocode

Pseudocode allows us to:

- Describe an algorithm using natural language;
- Write an algorithm in a formal way, following certain rules, which is close to a programming language;
- Write an algorithm without following strict syntax rules, unlike programming languages; and
- Increase readability of the logic used in an algorithm by using consistent notations.

Iterative Constructs

Iterative constructs run as follows:

- Instructions are repeated in a loop;
- The loop is initialised, also stating when the loop will be terminated;
- Test looping conditions and keep on changing the state to complete the loop; and
- Terminate the loop when the result satisfies the condition.

Types of iteration

- Pre-test iteration - the condition is checked first, then the code runs (While loop)
- Post-test iteration - the condition is checked after the code block has been processed (Until loop)

Algorithm complexity

An algorithm complexity function can measure efficiency of an algorithm by:

- Measuring how fast or slow a set of data is processed; and
- Measuring how much resources are used to process a set of data.

Algorithm Complexity Measurements

- Algorithm analysts often perform a comparison between two algorithms.
- Determine the amount of memory space the algorithm takes for a given input.
- Determine the amount of execution time for a given input.

Applying Algorithms

Algorithms can be implemented using:

- Programming languages which are equipped with predefined data types and data structure;
- Programming languages which offer control constructs to implement a series of instructions;

- Pseudocode, an informal way of representing an algorithm and it does not depend on a programming language; and
- An acceptable design approach where the steps are laid out clearly, showing how the problems will be solved.

High Level Programming Language

Third Generation Language

- Uses high-level primitives, similar to pseudocode;
- Mostly machine independent;
- Examples include: FORTRAN, COBOL;
- Each primitive corresponds to a sequence of machine language instructions; and
- Converted to machine language by a program called a compiler.

The Translation Process

- Algorithms are implemented by high-level languages.
    - Source code
    - Lexical Analyzer
    - Tokens
    - Parser
    - Parser trees
    - Code generator
    - Output program

Programming

High-level languages include:

- Constants, data types and variable declaration;
- Functions, statements, expressions, control; and
- Packages, procedures, modules and libraries.

In this module we will be covering Python. The Codio exercises on Python will enable you to have hands-on experience in using this programming language.

The Composition of a Program Unit

- Declaration statements to initiate the variables;
- Defining data types for storing and manipulating inputs;
- Defining functions and arguments;
- Applying control structures and iterations for arithmetic or logical operations;
- Checking conditions where necessary; and
- Returning the output value for printing or for further processing.

Data Types

Some examples of data types include the following:

- Integer - Whole numbers.
- Real (Float) - Numbers with fractions.
- Character - Symbols.
- Boolean - True/false.

Object-Oriented Programming

- Class - An abstract representation for data and its behavior they are often called methods, they are ‘templates’ on which objects are constructed
- Objects - An instance of a class that represents real world objects, it contains data and procedures.

Components of an object

The following lists different components of an object:

- Instance variable - Holds information within the object.
- Method - Describes the actions that the object can perform.
- Constructor - Special method used to initialise a new object when it is first constructed.

Object integrity

- Encapsulation - A way of restricting access to the internal components of an object.

- Inherits the attributes from the superclass.

Result Evaluation

The outputs of a computer program must have a purpose.

The output generated by an algorithm must be:

- Meaningful;
- Accurate;
- Timely; and
- Action-oriented.

Testing of Software

A computer program must be tested for its correctness and the validity of the program should be determined.

Testing can be done in the form of:

- Unit testing: to check individual module of software is working correctly;
- Integration testing: to check module is working correctly when is integrated to other modules; and
- Quality testing: a set of test cases are generated to check that the software is producing correct outputs

Quality assurance of software is usually conducted by developing a set of test cases. Apart from ensuring the correct outputs are produced, using test cases also ensures that the applied control structures and decision-making procedures are working as expected.