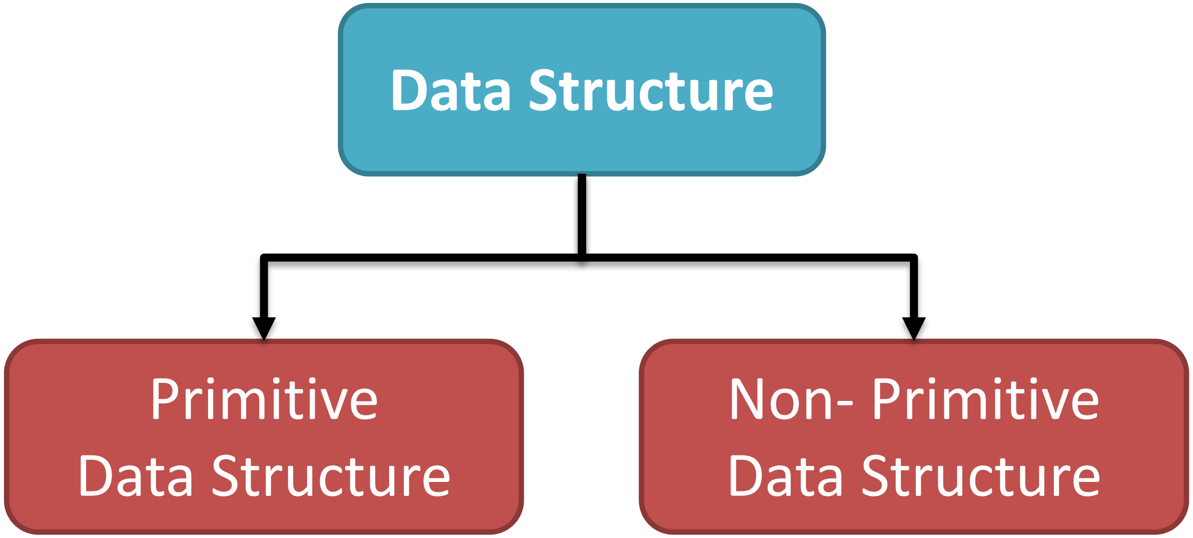
Introduction to Data Structures

- **Data Structure** is a way of storing and organizing data in computer memory so that it can be used efficiently.

- **Types of Data Structure:**



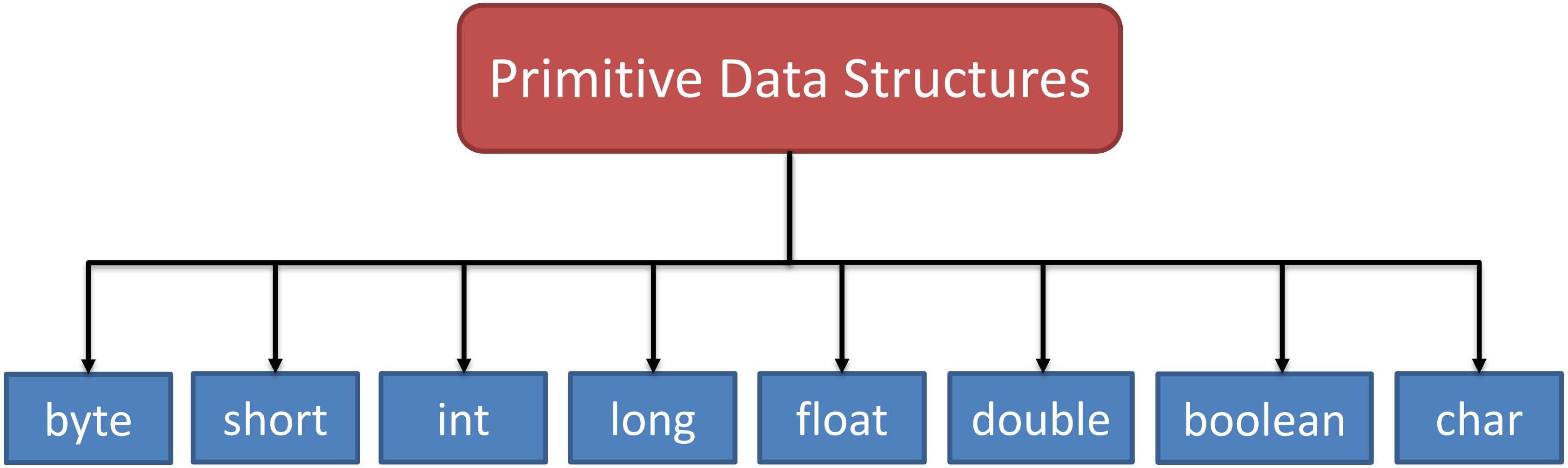
Data Structure

Primitive Data Structure

Non- Primitive Data Structure

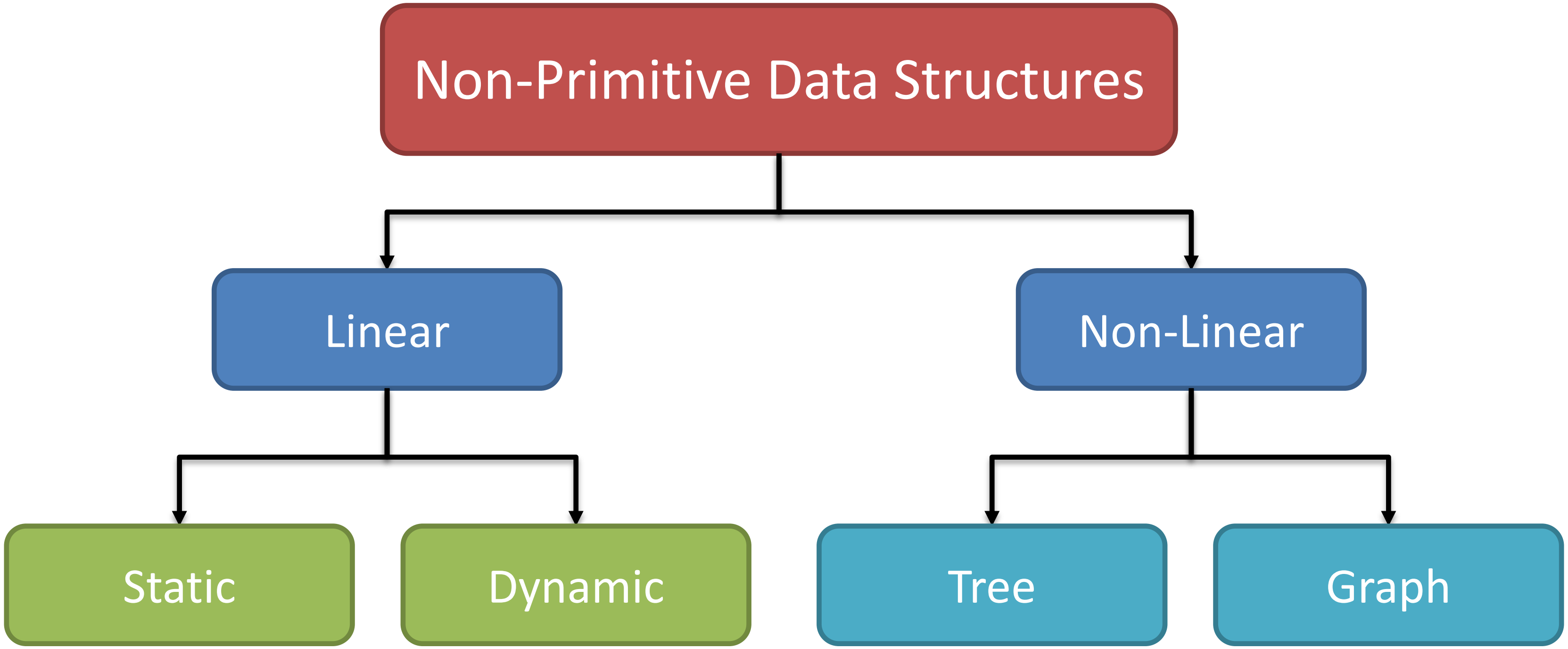
Primitive Data Structures

- **Primitive Data Structures** are those which can store only one value of one type.



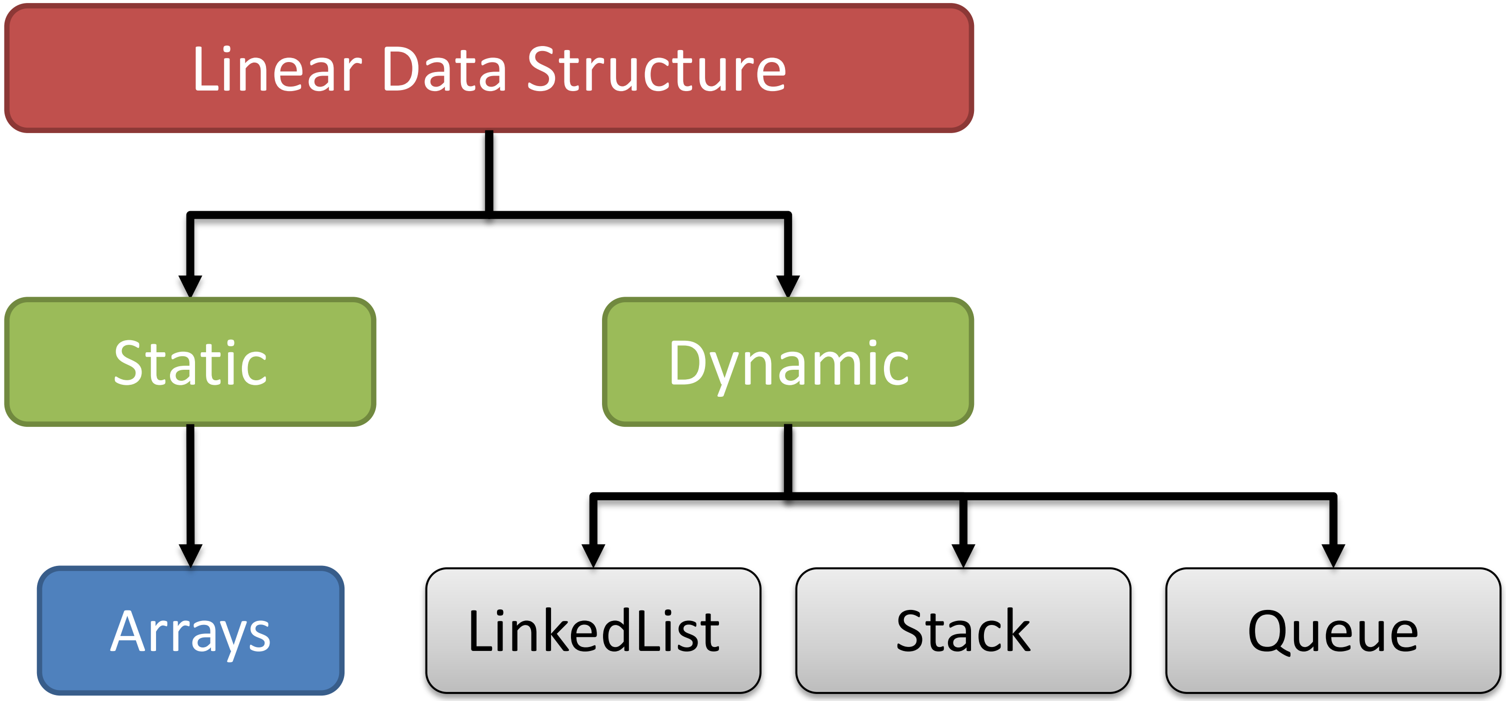
Non-Primitive Data Structures

- **Non-Primitive Data Structures** are those which can store multiple values of similar or dissimilar types.



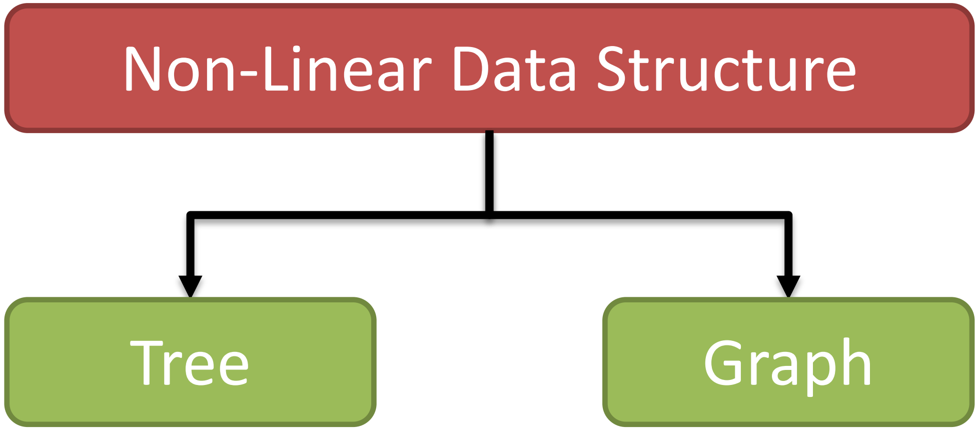
Linear Data Structures

- **Linear Data Structures** store the elements(values) in linear manner. Here, each element is connected to one other element.

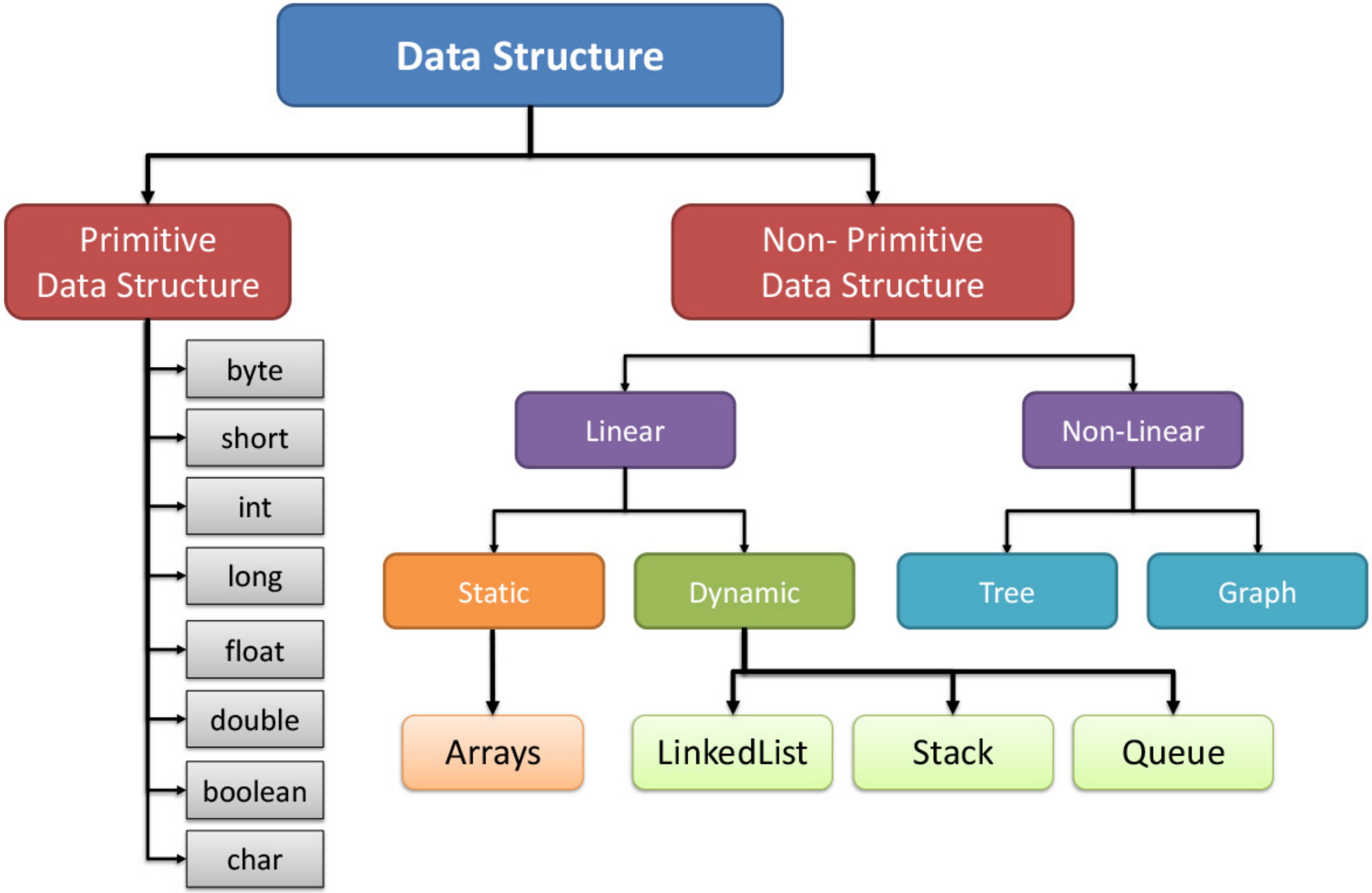


Non-Linear Data Structures

- **Non-Linear Data Structures** store the elements(values) in non-linear manner. Here, each element is connected to n- other elements.

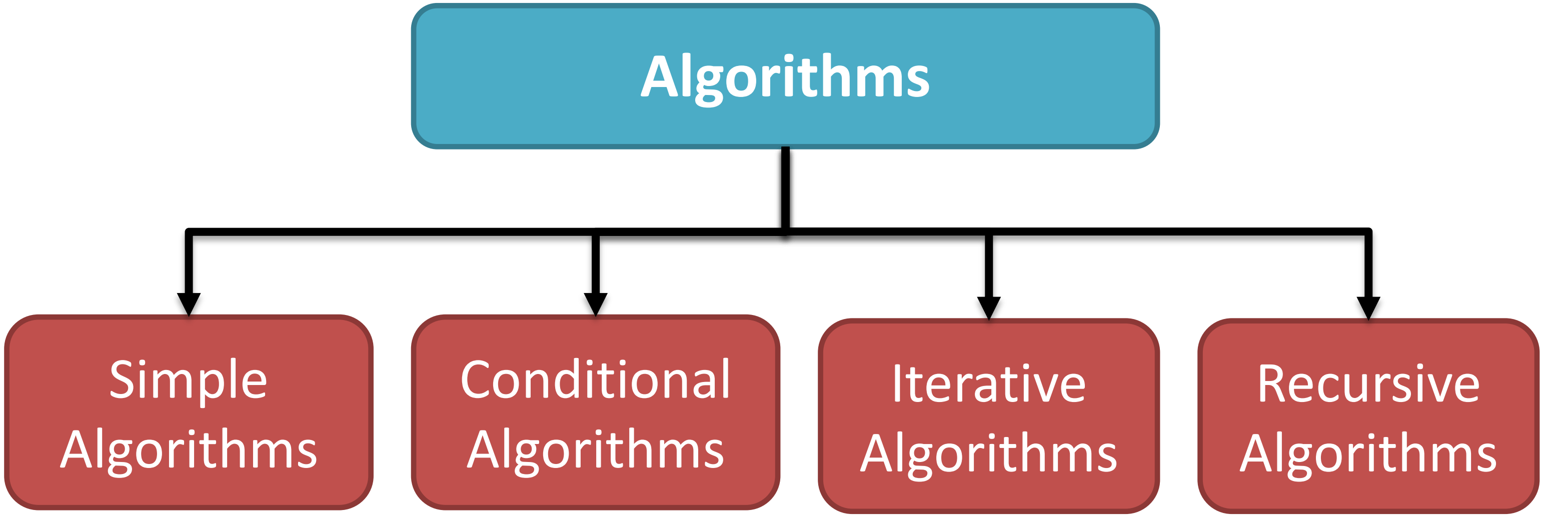


Data Structures Hierarchy



Introduction to Algorithms

- **Algorithm** is a collection or sequence of steps followed to performed a computational task. In simple words we can say it is a step-by-step process to solve a given computer problem.
- **Types of Algorithms:**



Algorithms

Simple Algorithms

Conditional Algorithms

Iterative Algorithms

Recursive Algorithms

Simple Algorithm

- An algorithm that contains only simple steps is called **simple algorithm**.

Simple Algorithm to Find Sum of Two Entered Numbers:

Let **n1** will store the first number entered through keyboard and **n2** will store the second number entered through keyboard and **s** will store their sum, then the algorithm is given below:

1. Read n1.
2. Read n2.
3. Set s = n1 + n2.
4. Print, sum of n1 and n2 is s.
5. Exit.

Conditional Algorithm

- An algorithm that contains one or more conditional step is called **conditional algorithm**.

Conditional Algorithm to Find Greatest of Two Entered Numbers:

Let **n1** will store the first number entered through keyboard and **n2** will store the second number entered through keyboard, then the algorithm is given below:

```
1. Read n1.
2. Read n2.
3. Check if n1 > n2, then:
     a) Print, n1 is greater than n2.
     b) Otherwise, check if n2 > n1, then:
          i.  Print, n2 is greater than n1.
     c) Otherwise,
          i.  Print, n1 is equal to n2.
4. Exit.
```

Iterative Algorithm

- An algorithm that contains one or more repetitional step is called **iterative algorithm**.
- Iterative algorithm are further of two types:
  - **Range Based Iteration Algorithm:** An algorithm that contains one or more ranged based repetitional step is called **range based** iterative algorithm.
  - **Condition Based Iteration Algorithm:** An algorithm that contains one or more condition based repetition step is called **condition based** iterative algorithm.

Iterative Algorithm

Range Based Iterative Algorithm to Print First n Natural Numbers:

Let **n** will store the last natural number entered through keyboard, and **i** will start with 1 for iteration then the algorithm is given below:

```
1. Read n.
2. Repeat for i = 1 to n
      a) Print, i
3. Exit.
```

Iterative Algorithm

Condition Based Iterative Algorithm to Count Digits in Entered Number:

Let **n** will store the number entered through keyboard, **c** will store the count of digits of entered number, **p** will store the copy of **n**, and then the algorithm is given below:

```
1.  Read n.
2.  Copy n to p
3.  Set c = 0.
4.  Repeat while n > 0. then
        a)  Set c = c + 1.
        b)  Set n = n / 10.
5.  Print, number of digits of given number p is c.
6.  Exit.
```

Recursive Algorithm

- An algorithm that contains one or more recursive step is called **recursive algorithm**.

Recursive Algorithm to Find Factorial of Entered Number:

Let **n** will store the number entered through keyboard, **f** will store the factorial value of entered number, and then the algorithm is given below:

```
1. Read n.
2. factorial(n)
     a) Check if n > 1, then:
          i.   return n * factorial(n - 1)
     b) Otherwise,
          i.   return 1
3. Set f = factorial(n)
4. Print, factorial of n is f.
5. Exit
```