## Programming Areas :

- Basic Logical Programming
- Object Oriented Programming
- Data Structure &  Algorithms

## Sample Practice Problems:

## 1.1: Basic Programming

Develop programs using built-in data types, conditions and loops, arrays and Input/Output Statements.

1. Write a program "***PrintFriendsNames***" that takes 3 friends' names as input arguments and prints out "Hi" to these friends in the reverse of the order given. It also adds "and " before the last name.

   For example:

   Input Arguments : Mahesh, Suresh, Devesh

   Output : "Hi Devesh, Suresh and  Mahesh"

2. Write a program "***PrintInitials***" that takes initials as input and prints the initials using nine rows of asterisks like the one below.

```
**        ***    **********      **             *             **
**      ***      **        **     **           ***           **
**    ***        **         **     **         ** **         **
**  ***          **          **     **       **   **       **
*****            **          **      **     **     **     **
**  ***          **          **       **   **       **   **
**    ***        **         **         ** **         ** **
**      ***      **        **           ***           ***
**        ***    **********              *             *
```

3. Write a program **CheckTheSeason** if the month number is entered. In this program check needs to be done if the month number is correct data type and numbers are between 1 to 12. If not, the program should throw an error and should continue till it is not correctly entered.

   Following table indicates Month numbers and seasons:

   2 & 3 : Vasanta

   4 & 5: Grishma / Summer

   6 & 7: Monsoon / Rainy

   8 & 9: Sharada

   10 & 11: Hemanta

   12 & 1 : Shishira / Winter

   12 & 1 : Winter

4. Write a program ***Quadratic*** to find the roots of the equation a*x*x + b*x + c. Since the equation is x*x, hence there are 2 roots. The 2 roots of the equation can be found using a formula

   delta = b*b - 4*a*c

   Root 1 of x = (-b + sqrt(delta))/(2*a)

   Root 2 of x = (-b - sqrt(delta))/(2*a)

   Take a, b, and c as input values to find the roots of x.

5. Write a ***TemperatureConversion*** program, given the temperature in Fahrenheit as input outputs the temperature in Celsius or vice versa using the formula

   Celsius to Fahrenheit: (°C × 9/5) + 32 = °F

   Fahrenheit to Celsius: (°F − 32) x 5/9 = °C.

6. Write a program which converts each character of an input string as an ASCII value and stores each of these numbers in an array. Print that array

## 1.2: Logical Programming:

1. Write a program that takes a range of numbers as input and outputs the Prime Numbers in that range.

2. Write a program ***PowerOf2*** that takes a command-line argument n and prints a table of the powers of 2 that are less than or equal to 2^n.

3. Write two programs ***Sin*** and ***Cos*** that compute sin x and cos x using the Taylor series expansions as shown below...

   Note - Convert angle x to an angle between -2 PI and 2 PI using the following logic

   x = x % (2 * Math.PI);

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots$$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots$$

4. Write a program to find common elements between two arrays (string values). Save the common values in another new array.

5. Write a program to find the second smallest element in an integer array.

6. Write a method to check if two strings are equal or not. If the second string's size is more than the first it returns -1. Else it compares character by character. If equals it returns 0. Else it returns the position wrt the first character where the difference is found.

7. Write a program ***SumOfTwoDice*** that prints the sum of two random integers between 1 and 6 (such as you might get when rolling dice).

## 1.3: Object Oriented Programming:

1. Write a program to compute the area of a rectangle by creating a class named 'Rectangle' having two methods. First method named as 'setDim' takes length and breadth of the rectangle as parameters and the second method named as 'getArea' returns the area of the rectangle. Length and breadth of rectangle are entered through the keyboard.

   Plan creating 2 rectangle objects, provide dimensions of them and then check which one has the higher area.

2. **Stock Report**
   a. Desc -> Write a program to read in Stock Names, Number of Share, Share Price. Print a Stock Report with total value of each Stock and the total value of Stock.
   b. I/P -> N number of Stocks, for Each Stock Read In the Share Name, Number of Share, and Share Price
   c. Logic -> Calculate the value of each stock and the total value
   d. O/P -> Print the Stock Report.
   e. Hint -> Create Stock and Stock Portfolio Class holding the list of Stocks read from the input file. Have functions in the Class to calculate the value of each stock and the value of total stocks

3. **Inventory Management Problem**

   a. Desc -> Extend the above program to Create InventoryManager to manage the Inventory. The Inventory Manager will use InventoryFactory to create Inventory Object from JSON. The InventoryManager will call each Inventory Object in its list to calculate the Inventory Price and then call the Inventory Object to return the JSON String. The main program will be with InventoryManager
   b. I/P -> read in JSON File
   c. Logic -> Get JSON Object in Java or NSDictionary in iOS. Create Inventory Object from JSON. Calculate the value for every Inventory.
   d. O/P -> Create the JSON from Inventory Object and output the JSON String.

# 1.4: Data Structures and Algorithms

1. Write a program to where it takes 10 integer numbers from a user, puts them in an array and then through Bubble sort, sorts this array. Input taking and forming an array should be one method and sorting should be other method.

2. Write a program to sort data using two algorithms and compare the performance.
   a. Generate 100,000 random floating point numbers 1.0 to 100.0 Keep these numbers in an array.
   b. Sort this array using Selection Sort and
   c. Sort with Quick Sort.
   d. Compute the time of execution for both the algorithms in mili-seconds.
   e. Check which one is faster and by what percentage?
   f. Number generation, Sorting algos and performance comparisons should be different methods.

3. Given a sorted array of n integers and a target value, determine if the target exists in the array using the binary search algorithm. If the target exists in the array, print the index of it.

   eg.

   Input array : 2, 3, 5, 7, 9

   Target : 7

   Output : Element found at Index 3

   Input array : 1, 4, 5, 8, 9

   Target : 7

   Output : Element Not Found

4. List Handling:

   a. Desc -> Read the Text from a file, split it into words and arrange it as Linked List. Take a user input to search a Word in the List. If the Word is not found then add it to the list, and if it found then remove the word from the List. In the end save the list into a file

b. I/P -> Read from file the list of Words and take user input to search a Text
c. Logic -> Create a Unordered Linked List. The Basic Building Block is the Node Object. Each node object must hold at least two pieces of information. One ref to the data field and  second the ref to the next node object.
d. O/P -> The List of Words to a File.