First Semester Theory Solutions

1. **Write a program that takes a positive integer 'N' from the user and arrange it in ascending order using selection sorting and perform binary search.**
   Class Name : SortSearch
   Data members/ Instance Variables:
   - **ar[] :** Single dimension array of integer type
   - **N :** to store integer data

   Member functions
   - **void inputData() :** to input 'N' and store data into array
   - **void sorting() :** to arrange into ascending order using Selection sort
   - **int binSearch(int sd) :** Search 'sd' using Binary Search technique.

   Specify the class **SortSearch** giving all the details as specified. Also define the main function to create an object and call accordingly to enable the task.

```
import java.util.*;
class SortSearch
{

int ar[];
int N;

void inputData()
{
    Scanner input = new Scanner(System.in);
    System.out.print("Enter the size of the array : ");
    N = input.nextInt();
    ar = new int [N];
    for(int i = 0; i < N; i++)
    {
        System.out.print("Enter the "+i+" term : ");
        ar[i] = input.nextInt();
    }
}

void sorting()
{
    int N = ar.length;
    for (int i = 0; i < N - 1; i++)
    {
        int min = i;
        for (int j = i + 1; j < N; j++)
        {
            if (ar[j] < ar[min])
            {
                min = j;
            }
        }
        int temp = ar[min];
        ar[min] = ar[i];
        ar[i] = temp;
    }
}

int binSearch(int sd)
{
    int low = 0;
    int high = N - 1;
```

```
        while (low <= high)
        {
            int mid = low + (high - low) / 2;
            if (ar[mid] == sd)
            {
                return mid;
            }
            else if (ar[mid] < sd)
            {
                low = mid + 1;
            }
            else
            {
                high = mid - 1;
            }
        }
        return -1;
    }

    public static void main(String Args[])
    {
        Scanner input = new Scanner(System.in);
        SortSearch object = new SortSearch();
        object.inputData();
        object.sorting();
        System.out.print("Enter the number you want to search for  : ");
        int search = input.nextInt();
        int result = object.binSearch(search);
        if(result != -1)
        {
            System.out.println(search + " is present at "+result);
        }
        else
        {
            System.out.println(search + "is not present in the array.");
        }
    }
}
```

1. Write a program which will input data into a two dimensional array and find the sum of all the elements of the array except those which are on either left or right diagonals and to display all the elements of the array which are below right diagonals.

Class Name : AR01

Data Member

- **ar[][] :** a two dimensional array to store integer data into it
- **N :** size of the row and column (Square matrix)

Member functions

- **AR01(int n1) :** a constructor to initialize each element of the array 'ar' with 0 and size 'N' with n1
- **void getData() :** to input data into array using keyboard scanner input
- **int sumDiag() :** to add all the elements of the array except those which are on either left or right diagonals
- **void dispHalf() :** to display all the elements of the array which are below right diagonals

Specify the class **AR01** giving all the details as specified. Also define the main function to create an object and call accordingly to enable the task.

```
import java.util.Scanner;
class AR01
{
    int ar[][];
    int N;

    AR01(int n1)
    {
        N = n1;
        ar = new int[N][N];
        for (int i = 0; i < N; i++)
        {
            for (int j = 0; j < N; j++)
            {
                ar[i][j] = 0;
            }
        }
    }

    void getData()
    {
        Scanner input = new Scanner(System.in);
        for(int i = 0; i< N; i++)
        {
            for(int j = 0; j < N; j++)
            {
                System.out.print("[" + i + "][" + j + "]: ");
                ar[i][j] = input.nextInt();
            }
        }
    }

    int sumDiag()
    {
        int sum = 0;
        for (int i = 0; i < N; i++)
        {
            for (int j = 0; j < N; j++)
            {
                if (i != j && j != N - 1 - i)
                {
                    sum += ar[i][j];
                }
```

```
            }
        }
        return sum;
    }

    void dispHalf()
    {
        System.out.println("The elements below the right diagonals are:");
        for (int i = 1; i < N; i++)
        {
            for (int j = 0; j < N; j++)
            {
                if (j > N - 1 - i)
                {
                    System.out.print(ar[i][j] + " ");
                }
            }
            System.out.println();
        }
    }

    public static void main(String Args[])
    {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the size of the array: ");
        int size = input.nextInt();
        AR01 object = new AR01(size);
        object.getData();
        int sum = object.sumDiag();
        System.out.println("The sum of all the elements except the left and right diagonals is:
        object.dispHalf();
    }
}
```

3. Design a class **JubNum** to check if a given number is an Jubilee number or not.
   (A number is said to be a Jubilee number if sum of its digits raised to the power of the length of the number is equal to the number.)
   Example:
   - $371 = 3^3 + 7^3 + 1^3$
   - $1634 = 1^4 + 6^4 + 3^4 + 4^4$
   - $54748 = 5^5 + 4^5 + 7^5 + 4^5 + 8^5$

Thus, 371, 1634 and 54748 are all examples of Jubilee numbers.

Some of the members of the class are given below:

Class Name : JubNum

Data Members/ Instance Variables

- **n :** to store the number
- **l :** to store the length of the number

Methods/ Member Functions

- **JubNum(int nn) :** parameterized constructor to initialize the data member n = nn
- **int sum_pow(int i) :** returns the sum of each digit raised to the power of the length of the number using recursive technique eg., 34 will return $3^2 + 4^2$ (as the length of the number is 2)
- **void isJubilee() :** checks whether the given number is a Jubilee number by invoking the function sum_pow() and displays the result with an appropriate message.

Specify the class **JubNum** giving details of the **constructor()**, **int sum_pow(int)** and **void isJubilee()**. Define a **main()** function to create an object and call the function accordingly to enable the task.

```
import java.util.*;
class JubNum
{
    int n;
    int l;

    JubNum(int nn)
    {
        n = nn;
        l = String.valueOf(n).length();
    }

    int sum_pow(int i)
    {

        if (i == 0)
        {
            return 0;
        }
        int digit = i % 10;
        return (int) Math.pow(digit, l) + sum_pow(i / 10);
    }

    void isJubilee()
    {
        if(n == sum_pow(n))
        {
            System.out.println(n + " is a Jubilee Number.");
        }
        else
        {
            System.out.println(n + " is not a Jubilee Number.");
        }
    }

    public static void main(String Args[])
    {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int N = input.nextInt();
        JubNum object = new JubNum(N);
        object.isJubilee();
```

```
    }
}
```

4. Input a word in uppercase and check for the position of the first occurring vowel and perform the following operation.
   (i) Words that begin with a vowel are concatenated with "Y".
   For example, EUROPE becomes EUROPEY.
   (ii) Words that contain a vowel in-between should have the first part from the position of the vowel till the end, followed by the part of the string from beginning till the position of the vowel and is concatenated by "C".
   For example, PROJECT becomes OJECTPRC.
   (iii) Words which do not contain a vowel are concatenated with "N".
   For example, SKY becomes SKYN.
   Design a class **Rearrange** using the description of the data members and member function given below:
   Class Name : Rearrange
   Data members/ Instance variables
   - **void readword() :** to accept the word input in UPPER CASE
   - **void convert() :** converts the word into its changed form and stores it in string *Cxt*
   - **void display() :** displays the original and the changed word

   Specify the class **Rearrange** giving the details of the **constructor()**, **void readword()**, **void convert()** and **void display()**.
   Define a **main()** function to create an object and call the function accordingly to enable the task.

```
import java.util.*;
class Rearrange
{
    String word;
    String rearranged_word;
    void readword()
    {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter a word: ");
        word = input.next().toUpperCase();
    }

    void convert()
    {
        char vowels [] = {'A', 'E', 'I', 'O', 'U'};
        int length = word.length();
        int vowel_index = -1;
        for(int i = 0; i < length; i++ )
        {
            char ch = word.charAt(i);
            for (int j = 0; j < vowels.length; j++)
            {
                if (ch == vowels[j])
                {
                    vowel_index = i;
                    break;
                }
            }
            if(vowel_index != -1)
            {
                break;
            }
        }

        if(vowel_index == 0)
        {
            rearranged_word = word + 'Y';
        }
        else if (vowel_index <= length && vowel_index != -1)
        {
            rearranged_word = word.substring(vowel_index, length) + word.substring(0, vowel_inde
        }
        else
```

```
        {
            rearranged_word = word + 'N';
        }
    }

    void display()
    {
        System.out.println("Original Word: "+word);
        System.out.println("Changed Word: "+rearranged_word);
    }

    public static void main(String Args[])
    {
        Rearrange object = new Rearrange();
        object.readword();
        object.convert();
        object.display();
    }
}
```

5. An emirp number is a number which is prime backwards and forwards.
   Example 13 and 31 are both prime numbers. This, 13 us an emirp number
   Design a class **Emirp** to check if a given number is Emirp number or not. Some of the members of the class are given below:
   Class Name : Emirp
   Data members/ Instance Variables
   - **n :** stores the number
   - **rev :** stores the reverse of the number
   - **f :** stores the divisor

   Member functions :
   - **Emirp(int nn) :** to assign n = nn, rev = 0 and f = 2
   - **int isprime(int x) :** check if the number is prime using the recursive technique and return 1 if prime otherwise return 0
   - **void isEmirp() :** reverse the given number and check if both the original number and the reverse number are prime, by invoking the function isprim(int) and display the result with an appropriate message.

   Specify the class **Emirp** giving details of the **constructor(int)**, **int isprime(int)** and **void isEmirp()**. Define the main function to create an object and call the methods to check for Emirp number.

```
import java.util.*;
class Emirp
{
    int n;
    int rev;
    int f;

    Emirp(int nn)
    {
        n = nn;
        rev = 0;
        f = 2;
    }

    int isPrime(int x, int f)
    {
        if (f == 1)
        {
            return 1;
        }
        if (x % f == 0)
        {
            return 0;
        }
        return isPrime(x, f - 1);
    }

    void isEmirp()
    {
        int store = n;
        int temp=0,reverse=0;
        int number_digits=(String.valueOf(n).length());
        for(int i=1;i<=number_digits;i++)
        {
            temp=(store%10);
            reverse=((reverse*10)+temp);
            store=store/10;
        }
        if(isPrime(n, f) == 1 && isPrime(reverse, f) == 1)
        {
            System.out.println(n + " is an Emirp number.");
        }
        else
```

```
        {
            System.out.println(n + " is not an Emirp number.");
        }
    }

    public static void main(String Args[])
    {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter a number : ");
        int in = input.nextInt();
        Emirp object = new Emirp(in);
        object.isEmirp();
    }
}
```