# C-DAC Mumbai

**Date 25/09/2024**

## Subject: Algorithm and Data Structure
## Assignment 1

**Solve the assignment with following thing to be added in each question.**

-Program
-Flow chart
-Explanation
-Output
-Time and Space complexity

1. Armstrong Number
Problem: Write a Java program to check if a given number is an Armstrong number.

Test Cases:

Input: 153
Output: true
Input: 123
Output: false
Sol:-
```
import java.util.Scanner;
class HelloWorld {
	public static boolean armstrong(int n){
	int j=n;
	int temp=0;
	int sum=0;
	while(n!=0){// 121 , 12 , 1 ;  the while loop runs once per digit log(number);
	temp=n%10;
	sum=sum+(int)Math.pow(temp,3);
	n=n/10;
	}
	if(j==sum){
	return true;
	}
	else{
	return false;
	}

	}
	public static void main(String[] args) {
	Scanner sc=new Scanner(System.in);
	System.out.println(armstrong(sc.nextInt()));
	}
}
```
2. Prime Number
Problem: Write a Java program to check if a given number is prime.

Test Cases:

Input: 29
Output: true
Input: 15
Output: false

```
import java.util.Scanner;
class HelloWorld {
	public static boolean isPrime(int n){
	int flag=0;
	for(int i=2;i<n;++i){/// O(n)
	if(n%i==0){
	return false;
	}
	}

	return true;

	}
	public static void main(String[] args) {
	Scanner sc=new Scanner(System.in);
	System.out.println(isPrime(sc.nextInt()));
	}
}
```

3. Factorial
Problem: Write a Java program to compute the factorial of a given number.

Test Cases:

Input: 5
Output: 120
Input: 0
Output: 1

```
public class Program{
	public static int fact(int i){
	if(i==1)
	return 1;
	else
	return fact(i-1)*i;
	}
public static void main(String [] args){
int j=fact(3);
System.out.println(i+" "+j);
}
}
```

4. Fibonacci Series
Problem: Write a Java program to print the first n numbers in the Fibonacci series.

Test Cases:

Input: n = 5
Output: [0, 1, 1, 2, 3]
Input: n = 8
Output: [0, 1, 1, 2, 3, 5, 8, 13]

```
public class Program{
        public static int fib(int i){
        if(i==0)
        return 0;
        if(i==1)
        return 1;
        else
        return fib(i-1)+fib(i-2);
        }
public static void main(String [] args){
int i=fib(3);
System.out.println(i+" "+j);
}
}
```

5. Find GCD
Problem: Write a Java program to find the Greatest Common Divisor (GCD) of two numbers.

Test Cases:

Input: a = 54, b = 24
Output: 6
Input: a = 17, b = 13
Output: 1

```
class HelloWorld {
  public static double gcd(double a,double b){
        if(b==0)
        return a;
        else {
        return gcd(b,a%b);
        }
  }
        public static void main(String[] args) {
        System.out.println(gcd(5,6));
        }
}
```

6. Find Square Root
Problem: Write a Java program to find the square root of a given number (using integer approximation).

Test Cases.
Input: x = 16
Output: 4
Input: x = 27
Output: 5

```
class HelloWorld {
        public static double load(double a){
        if(a<0){
```

```
        System.out.println("this application only calculates the square root of a positive number ");
        return 0;
        }
        else if(a<=1)
        return a;
        else {
        return    sqrt(a,0,1);
        }
        }
        public static double sqrt(double a,double guess,double step){// O(log log n).
        double square =guess*guess;
        if(a==square) return guess;
        if(square > a){
        if(step > 0.000001) return sqrt(a,guess-step,step/10);
        else return guess-step;
        }
        else{
        return sqrt(a,guess+step,step);
        }
        }
        public static void main(String[] args) {
        System.out.println(load(25));
        }
}
```

7. Find Repeated Characters in a String
Problem: Write a Java program to find all repeated characters in a string.

Test Cases:

Input: "programming"
Output: ['r', 'g', 'm']
Input: "hello"
Output: ['l']

```
import java.util.Scanner;
class HelloWorld {
        public static void isDuplicate(char[] arr){
        char temp=arr[0];
        for(int i=0;i<arr.length-1;++i){
        temp=arr[i];
        if(temp==arr[i+1]){
        System.out.println(arr[i]);
        }
        }
        }
        public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        String s=sc.next();
        char[] arr=new char[s.length()];
        for(int i=0;i<arr.length;++i){
        arr[i]=s.charAt(i);
```

```
	}
	isDuplicate(arr);
	}
}
```

8. First Non-Repeated Character

Problem: Write a Java program to find the first non-repeated character in a string.

Test Cases:

Input: "stress"
Output: 't'
Input: "aabbcc"
Output: null

```
package recursion;

import java.util.Scanner;
class Hello {
	public static int isDuplicate(String s){
	if(s.length()<=1){
	return 0;
	}
	char c=s.charAt(1);
	String g=s.substring(1);
	boolean un=!g.contains(String.valueOf(c));
	if(un){
	return 1;
	}
	else {
	return 1 + isDuplicate(s.substring(1));//O(n)
	}
	}
	public static void main(String[] args) {
	Scanner sc=new Scanner(System.in);
	String s=sc.next();
	int i=isDuplicate(s);
	System.out.println(s.charAt(i));
	sc.close();
	}
}
```

9. Integer Palindrome

Problem: Write a Java program to check if a given integer is a palindrome.

Test Cases:

Input: 121
Output: true
Input: -121
Output: false

```
class HelloWorld {
```

```
 public static boolean ispal(int a){
        String s=Integer.toString(a);
        return pal(s);
 }
 public static boolean pal(String s){
        if(s.length()==1)
        return true;
        if(s.charAt(0)!=s.charAt(s.length()-1))
        return false;

        return pal(s.substring(1,s.length()-1));
 }
        public static void main(String[] args) {
        System.out.println(ispal(-121));
        }
}
```

10. Leap Year
Problem: Write a Java program to check if a given year is a leap year.

Test Cases:

Input: 2020
Output: true
Input: 1900
Output: false

```
class HelloWorld {

 public static boolean isleap(int year){

        if(year%4==0){

        if(year%100==0){

        return year%400==0;

        }

        return true;

        }

        return false;

 }

        public static void main(String[] args) {

        System.out.println(isleap(2011));

        }
```

```
}
```