**********MixTechSolutions**********

**********A New Way to Change your Life**********

**********Core Java Programs**********

Fibonacci Series Program 1:

```
1.          class FibonacciExample1{
2.          public static void main(String args[])
3.          {
4.           int n1=0,n2=1,n3,i,count=10;
5.           System.out.print(n1+" "+n2);//printing 0 and 1
6.
7.           for(i=2;i<count;++i)//loop starts from 2 because 0 and 1 are already printed
8.           {
9.            n3=n1+n2;
10.           System.out.print(" "+n3);
11.           n1=n2;
12.           n2=n3;
13.          }
14.
15.         }}
```


Prime Number Program 2:

Prime number in Java: **Prime number** is a number that is greater than 1 and divided by 1 or itself only. In other words, prime numbers can't be divided by other numbers than itself or 1. For example 2, 3, 5, 7, 11, 13, 17.... are the prime numbers.

```
1.          public class PrimeExample{
2.           public static void main(String args[]){
3.            int i,m=0,flag=0;
4.            int n=3;//it is the number to be checked
5.            m=n/2;
6.            if(n==0||n==1){
7.             System.out.println(n+" is not prime number");
8.            }else{
9.             for(i=2;i<=m;i++){
10.             if(n%i==0){
11.              System.out.println(n+" is not prime number");
12.              flag=1;
13.              break;
14.             }
15.            }
16.            if(flag==0)  { System.out.println(n+" is prime number"); }
17.           }//end of else
18.         }
19.         }
```


Palindrome Program 3:

Palindrome number in java: A **palindrome number** is *a number that is same after reverse*. For example 545, 151, 34543, 343, 171, 48984 are the palindrome numbers.

```
1.  class PalindromeExample{
2.   public static void main(String args[]){
3.    int r,sum=0,temp;
4.    int n=454;//It is the number variable to be checked for palindrome
5.
6.    temp=n;
7.    while(n>0){
8.     r=n%10;  //getting remainder
9.     sum=(sum*10)+r;
10.    n=n/10;
11.   }
12.   if(temp==sum)
13.    System.out.println("palindrome number ");
14.   else
15.    System.out.println("not palindrome");
16. }
17. }
```


Factorial Program in Java

Factorial Program in Java: Factorial of n is the *product of all positive descending integers*. Factorial of *n* is denoted by n!. For example:

```
1.  4! = 4*3*2*1 = 24
2.  5! = 5*4*3*2*1 = 120
```

```
1.          class FactorialExample{
2.           public static void main(String args[]){
3.            int i,fact=1;
4.            int number=5;//It is the number to calculate factorial
5.            for(i=1;i<=number;i++){
6.                fact=fact*i;
7.            }
8.            System.out.println("Factorial of "+number+" is: "+fact);
9.           }
10.         }
```

Armstrong Number Program 5:

An **Armstrong** number is a positive m-digit number that is equal to the sum of the m^{th} powers of their digits.

153: $1^3 + 5^3 + 3^3 = 1 + 125 + 27 = \mathbf{153}$

```
1.  import java.util.Scanner;
2.  import java.lang.Math;
3.  public class ArmstsrongNumberExample
4.  {
5.  //function to check if the number is Armstrong or not
6.  static boolean isArmstrong(int n)
7.  {
8.  int temp, digits=0, last=0, sum=0;
9.  //assigning n into a temp variable
10. temp=n;
11. //loop execute until the condition becomes false
12. while(temp>0)
13. {
14. temp = temp/10;
15. digits++;
16. }
17. temp = n;
18. while(temp>0)
19. {
20. //determines the last digit from the number
21. last = temp % 10;
22. //calculates the power of a number up to digit times and add the resultant to the
    sum variable
23. sum +=  (Math.pow(last, digits));
24. //removes the last digit
25. temp = temp/10;
26. }
27. //compares the sum with n
28. if(n==sum)
29. //returns if sum and n are equal
30. return true;
```

```
31.         //returns false if sum and n are not equal
32.         else return false;
33.         }
34.         //driver code
35.         public static void main(String args[])
36.         {
37.         int num;
38.         Scanner sc= new Scanner(System.in);
39.         System.out.print("Enter the limit: ");
40.         //reads the limit from the user
41.         num=sc.nextInt();
42.         System.out.println("Armstrong Number up to "+ num + " are: ");
43.         for(int i=0; i<=num; i++)
44.         //function calling
45.         if(isArmstrong(i))
46.         //prints the armstrong numbers
47.         System.out.print(i+ ", ");
48.         }
49.         }
```


Reverse a number Program 6:

ReverseNumberExample1.java

```
1.  public class ReverseNumberExample1
2.  {
3.  public static void main(String[] args)
4.  {
5.  int number = 987654, reverse = 0;
6.  while(number != 0)
7.  {
8.  int remainder = number % 10;
9.  reverse = reverse * 10 + remainder;
10. number = number/10;
11. }
12. System.out.println("The reverse of the given number is: " + reverse);
13. }
14. }
```


Program 7 to print the duplicate elements of an array

```
1.  public class DuplicateElement {
2.  public static void main(String[] args) {
```

```
3.              //Initialize array
4.              int [] arr = new int [] {1, 2, 3, 4, 2, 7, 8, 8, 3};
5.              System.out.println("Duplicate elements in given array: ");
6.              //Searches for duplicate element
7.              for(int i = 0; i < arr.length; i++) {
8.                 for(int j = i + 1; j < arr.length; j++) {
9.                    if(arr[i] == arr[j])
10.                      System.out.println(arr[j]);
11.                }
12.             }
13.          }
14.       }
```


Program 8 to print the largest element in an array

```java
public class LargestElement_array {
   public static void main(String[] args) {

      //Initialize array
      int [] arr = new int [] {25, 11, 7, 75, 56};
      //Initialize max with first element of array.
      int max = arr[0];
      //Loop through the array
      for (int i = 0; i < arr.length; i++) {
         //Compare elements of array with max
        if(arr[i] > max)
            max = arr[i];
      }
      System.out.println("Largest element present in given array: " + max);
   }
}
```


Java Program 9 to sort the elements of an array in ascending order

```java
public class SortAsc {
   public static void main(String[] args) {

      //Initialize array
      int [] arr = new int [] {5, 2, 8, 7, 1};
      int temp = 0;

      //Displaying elements of original array
      System.out.println("Elements of original array: ");
      for (int i = 0; i < arr.length; i++) {
         System.out.print(arr[i] + " ");
      }

      //Sort the array in ascending order
      for (int i = 0; i < arr.length; i++) {
         for (int j = i+1; j < arr.length; j++) {
            if(arr[i] > arr[j]) {
               temp = arr[i];
               arr[i] = arr[j];
               arr[j] = temp;
            }
         }
      }

      System.out.println();

      //Displaying elements of array after sorting
      System.out.println("Elements of array sorted in ascending order: ");
      for (int i = 0; i < arr.length; i++) {
         System.out.print(arr[i] + " ");
      }
   }
}
```

Java program 10 to find the duplicate words in a string

In this program, we need to find out the duplicate words present in the string and display those words.

Example: big black bug bit a big black dog on his big black nose

```
1.  public class DuplicateWord {
2.    public static void main(String[] args) {
3.      String string = "Big black bug bit a big black dog on his big black nose";
4.      int count;
5.
6.      //Converts the string into lowercase
7.      string = string.toLowerCase();
8.
9.      //Split the string into words using built-in function
10.     String words[] = string.split(" ");
11.
12.     System.out.println("Duplicate words in a given string : ");
13.     for(int i = 0; i < words.length; i++) {
14.       count = 1;
15.       for(int j = i+1; j < words.length; j++) {
16.         if(words[i].equals(words[j])) {
17.           count++;
18.           //Set words[j] to 0 to avoid printing visited word
19.           words[j] = "0";
20.         }
```

```
21.                 }
22.
23.                 //Displays the duplicate word if count is greater than 1
24.                 if(count > 1 && words[i] != "0")
25.                     System.out.println(words[i]);
26.             }
27.         }
28.     }
```