1. Program to show use of variable and data-types.

```java
public class MainVar
{
    public static void main(String arg[])
    {
        int      myNum      =   5;          // Integer (whole number)
        float    myFloatNum =   5.99f;      // Floating point number
        char     myLetter   =   'D';        // Character
        boolean  myBool     =   true;       // Boolean
        String   myText     =   "Hello";    // String
    }
}
```

2. Program for operators.

```java
public class Operator
{
    void arithmatic()
    {
        int     sum     =   10 + 5;
        int     diff    =   10 - 3;
        int     mult    =   10 * 2;
        int     div     =   10 / 2;
        int     mod     =   10 % 4;
        int     a       =   5;

        a++;
        a--;
    }

    void comparative()
    {
        int x = 5;
        int y = 3;

        System.out.println(x == y);
        System.out.println(x > y);
        System.out.println(x < y);
    }

    void logical()
    {
        int x = 5;
        int y = 3;

        System.out.println(  (x > 3) && (y < 10)  );
        System.out.println(  (x > 3) || (y < 10)  );
        System.out.println(  !(x > 3)  );
    }

    public static void main(String arg[])
    {

    }
}
```

3. Program to determine even odd numbers.

```
public class EvenOdd
{
        public static void main(String arg[])
        {
                int num, rem;

                num =   21;
                rem =   num%2;

                if( rem == 0 )
                {
                    System.out.println("even");
                }
                else
                {
                    System.out.println("odd");
                }

        }
}
```

4. Program to find maximum of 3 numbers.

```
public class Max
{
    static int getMax(int a, int b, int c)
    {
        int max;

        if( a>b  &&  a>c )
        {
            max = a;
        }
        else if( b>c )
        {
            max = b;
        }
        else
        {
            max = c;
        }

        return(max);
    }

    public static void main(String arg[])
    {
        int max  =  getMax(35, 70, 15);

        System.out.println(max);
    }
}
```

5. Program using loops.

```
public class Loops
{
        public static void main(String arg[])
        {
                int i;

                i=1;
                while(  i < 11  )
                {
                    System.out.println(i);
                    i++;
                }


                for( i=1; i<11; i++)
                {
                    System.out.println(i);
                }


                i=1;
                do
                {
                    System.out.println(i);
                    i++;
                }
                while(  i < 11  );

        }
}
```

6. Program using break and continue.

```
public class Loop
{
        public static void main(String arg[])
        {
                int i;


                for( i=1; i<11; i++)
                {
                    System.out.println(i);
                    break;
                }


                for( i=1; i<11; i++)
                {
                    if( i > 5 )
                    {
                        continue;
                    }
                    System.out.println(i);
                }

        }
}
```

7. Program to print table of number.

```java
public class Table
{

    static void printTable(int num)
    {
        int i, multi;

        for( i=1; i<11; i++)
        {
            multi   =   num * i;
            System.out.println(multi);
        }
    }

    public static void main(String arg[])
    {
        printTable(7);
    }
}
```

8. Program using arrays.

```
public class Arrays
{
        public static void main(String arg[])
        {
                int i;

                String[] color = {"Red", "Blue", "Green", "Pink"};

                System.out.println(color[2]);

                System.out.println(color.length);

                for ( i=0; i < color.length; i++ )
                {
                    System.out.println(color[i]);
                }


                int[][] myNumbers = { {1, 2, 3, 4}, {5, 6, 7} };
                System.out.println(myNumbers[1][2]);
        }
}
```

9. Program using overloading.

```
public class Overloading
{
        void add(int a, int b)
        {
            int c = a + b;
        }

        void add(int a, int b, int c)
        {
            int d = a + b + c;
        }


        public static void main(String arg[])
        {
            Overloading obj =   new Overloading();

            obj.add(10, 3);

            obj.add(10, 3, 1);

        }
}
```

10. Program using inheritance.

```
class Parent
{
        String name, city;
}


public class Child extends Parent
{
        int age;

        public static void main(String arg[])
        {
            Child obj    =   new Child();
            obj.name     =   "Amit";
            obj.city     =   "Sehore";
            obj.age      =   25;

            Parent  p1   =   new Parent();
            Parent  p2   =   new Child();

        }
}
```

11. Program using interface.

```
interface Shape
{
    //implicitly public, static and final
    public String LABLE    =    "Shape";

    //interface methods are implicitly abstract and public
    int getArea();
}

class Rectangle implements Shape
{
    int height, width;

    public int getArea()
    {
        int area = height * width;
        return(area);
    }
}

public class MainInterface
{
    public static void main(String[] args)
    {
        // Shape s1    =    new Shape();

        Rectangle r1    =    new Rectangle();

    }
}
```

12. Program using over-riding.

```
class Parent
{
        void show()
        {
            System.out.println(" inside parent class");
        }
}


public class Overriding extends Parent
{
        void show()
        {
            System.out.println(" inside Child class");
        }

        public static void main(String arg[])
        {
            Parent      p1  =   new Parent();
            Overriding  c1  =   new Overriding();

            p1.show();
            c1.show();

        }
}
```

13. Program using polymorphism.

```
class Parent
{
        void show()
        {
            System.out.println(" inside parent class");
        }
}


public class Polym extends Parent
{
        void show()
        {
            System.out.println(" inside Child class");
        }

        public static void main(String arg[])
        {
            Parent   p1  =  new Parent();
            Parent   p2  =  new Polym();

            p1.show();
            p2.show();

        }
}
```

14. Program using threads.

```
class BlueThread extends Thread
{
        public void run()
        {
                System.out.println("running blue thread");
        }
}


class Main implements Runnable
{
        public void run()
        {
                System.out.println("running green thread");
        }
}

public class ThreadUser
{
        public static void main(String arg[])
        {
            BlueThread t1    =   new BlueThread();
            t1.start();

            Main obj         =   new Main();
            Thread t3        =   new Thread(obj);
            t3.start();
        }
}
```

15. Program for input and output.

```
import java.io.InputStreamReader;
import java.io.IOException;

public class InputReader
{

    public static void main(String args[]) throws IOException
    {
          InputStreamReader rObj = new InputStreamReader(System.in);
          char ch;

          int ascii    =   rObj.read();
          ch           =   (char)ascii;

          ascii        =   System.in.read();
          ch           =   (char)ascii;

          System.out.print(ch);
    }
}
```

16. Program to write a file.

```
import java.io.FileWriter;
import java.io.IOException;

public class FWriter
{
        public static void main(String arg[]) throws IOException
        {
            FileWriter fw    =   new FileWriter("./sample.txt");

            fw.write('A');
            fw.write('P');
            fw.write('P');
            fw.write('L');
            fw.write('E');

            fw.close();

        }
}
```

17. Program to read a file.

```
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;

public class FReader
{
    public static void main(String arg[]) throws IOException, FileNotFoundException
    {
        FileReader fr    =   new FileReader("./sample.txt");
        int ascii;
        char ch;

        ascii   =   fr.read();

        while (ascii!=-1)
        {
            ch      =   (char)ascii;
            System.out.print(ch);

            ascii   =   fr.read();
        }

        fr.close();

    }
}
```

18. Program using exception handline.

```
import java.io.*;

public class ExceptionTest
{

    public static void readFile()
    {
        try
        {
            FileReader fr   =   new FileReader("./apple.txt");
        }
        catch(FileNotFoundException e)
        {
            System.out.println(e.getMessage());
        }
        finally
        {
            System.out.println("Finally is always executed");
        }
    }

    public static void readFile1() throws FileNotFoundException
    {
        FileReader fr   =   new FileReader("./apple.txt");
    }

    public static void main(String args[])
    {
        // readFile();
        // readFile1();
    }
}
```