Experiment No 1

Title : Basic Java Programs Based on I/O, Operators, and Strings

Code :

```
import java.util.Scanner;
class Start
{
   public static void main(String[] args)
   {
      Scanner s = new Scanner(System.in);
      System.out.print("Enter a number:");
      int num1 = s.nextInt();
      System.out.print("Enter another number:");
      int num2 = s.nextInt();

      int sum = num1 + num2;
      int diff = num1 - num2;
      int prod = num1 * num2;
      int quot = 0;
      if (num2 != 0)
      {            quot = num1 / num2;            }
      int rem = 0;
      if (num2 != 0)
      {          rem = num1 % num2;             }

      System.out.println("Sum: " + sum);
      System.out.println("Difference: " + diff);
      System.out.println("Product: " + prod);
      if (num2 != 0)  {
         System.out.println("Quotient: " + quot);
         System.out.println("Remainder: " + rem);
      }  else   {
         System.out.println("Cannot divide by zero.");
      }

      System.out.print("Enter a string:");
      String str1 = s.next();
      String str2 = "World";
      String str3 = str1.concat(str2);
      System.out.println("Concatenated string: " + str3);
      System.out.println("Length of str1: " + str1.length());
      System.out.print("Substring of str3 (0,3): " );
      System.out.print( str3.substring(0, Math.min(3, str3.length())));
      System.out.println("Uppercase str1: " + str1.toUpperCase());
      s.close();
   }
}
```

Output

```
Enter a number:10
Enter another number:5
Sum: 15
Difference: 5
Product: 50
Quotient: 2
Remainder: 0
Enter a string:Hello
Concatenated string: HelloWorld
Length of str1: 5
Substring of str3 (0,3): Hell
Uppercase str1: HELLO
```

Experiment No. 2

Title : Implement basic java programs based on control statements.

Code :

```
import java.util.*;

class Control
{
   public static void main(String[] args)
   {
      int num = 10;
      if (num > 0) {
         System.out.println("Number 10 is positive.");
      }
      else if (num < 0) {
         System.out.println("Number is negative.");
      }
      else {
         System.out.println("Number is zero.");
      }

         Scanner s = new Scanner(System.in);
         System.out.print("Enter grade:");
         char num1 = s.next().charAt(0);

      char grade = num1;
      switch (grade) {
         case 'A':
            System.out.println("Excellent!");
            break;
         case 'B':
            System.out.println("Well done");
            break;
         case 'C':
            System.out.println("Good");
            break;
         default:
            System.out.println("Invalid grade");
      }

      System.out.println("Natural Numbers till 3:");
      System.out.println("For loop:");
      for (int i = 1; i <= 3; i++) {
         System.out.print(" "+ i );
      }
```

```
        System.out.println("\nWhile:");
        int j = 1;
        while (j <= 3) {
            System.out.print( " " + j );
            j++;
        }

        System.out.println("\nDo-While:");
        int k = 1;
        do {
            System.out.print(" " + k);
            k++;
        } while (k <= 3);
        System.out.println("\n____________________");
        for (int x = 0; x < 5; x++) {
            if (x == 2) {
                System.out.println("'Continue' at x = 2");
                continue;
            }
            if (x == 4) {   System.out.println("'Break' at x = 4");
                break;
            }
            System.out.println(x);
        }
    }

}
```

Output

```
Number 10 is positive.
Enter grade:A
Excellent!
Natural Numbers till 3:
For loop:
 1 2 3
While:
 1 2 3
Do-While:
 1 2 3
____________________
0
1
'Continue' at x = 2
3
'Break' at x = 4
```

Experiment No 3

Title : Java Programs Using Constructors and Constructor Overloading

```
class Box {
   double w;
   double h;
   double d;
   Box(){
      w = 1;
      h = 1;
      d = 1;
      System.out.println("Default constructor called.");
   }

   Box(double side){
      w = side;
      h = side;
      d = side;
      System.out.println("Cube constructor called.");
   }

   Box(double width, double height, double depth) {
      w = width;
      h = height;
      d = depth;
      System.out.println("Parameterized constructor called.");
   }

   double volume() {
      return w * h * d;
   }

   public static void main(String[] args) {
      Box b1 = new Box();
      System.out.println("Volume of b1: " + b1.volume());
      Box b2 = new Box(5);
      System.out.println("Volume of b2: " + b2.volume());
      Box b3 = new Box(2, 3, 4);
      System.out.println("Volume of b3: " + b3.volume());
   }
}
```

Output

```
Default constructor called.
Volume of b1: 1.0
Cube constructor called.
Volume of b2: 125.0
Parameterized constructor called.
Volume of b3: 24.0
```

Experiment No 4

Title : Java Programs Using Methods (Static, Non-Static, Recursive) and Method Overloading

Code :

```
class Methods
{
    void normal() {
        System.out.println("This is a non-static method.");
    }
    static void stat() {
        System.out.println("This is a static method.");
    }
    void display(int a) {
        System.out.println("Method with int: " + a);
    }
    void display(String s) {
        System.out.println("Method with String: " + s);
    }
    void display(int a, double b) {
        System.out.println("Method with int and double: " + a + ", " + b);
    }
    long factorial(int n) {
        if (n == 0 || n == 1) {
            return 1;
        } else {
            return n * factorial(n - 1);
        }
    }

    public static void main(String[] args) {
        Methods m = new Methods();
        m.normal();
        Methods.stat();
        stat();
        m.display(10);
        m.display("Hello");
        m.display(5, 2.5);
        int num = 5;
        System.out.println("Factorial of " + num + " is " + m.factorial(num));
    }
}
```

Output

```
This is a non-static method.
This is a static method.
This is a static method.
Method with int: 10
Method with String: Hello
Method with int and double: 5, 2.5
Factorial of 5 is 120
```

Experiment No 5

Title : Java Programs to Demonstrate Inheritance and Method Overriding

Code :

```
class Animal {
   void sound() {
      System.out.println("Animal makes a sound");
   }
   void eat() {
      System.out.println("Animal eats");
   }
}
class Dog extends Animal
{
   @Override
   void sound() {
      System.out.println("Dog barks");
   }
   void fetch() {
      System.out.println("Dog fetches ball");
   }

   public static void main(String[] args) {
      Animal a1 = new Animal();
      a1.sound();
      a1.eat();
      Dog d1 = new Dog();
      d1.sound();
      d1.eat();
      d1.fetch();
      Animal a2 = new Dog();
      a2.sound();
      a2.eat();
   }
}
```

Output :

```
Animal makes a sound
Animal eats
Dog barks
Animal eats
Dog fetches ball
Dog barks
Animal eats
```

Experiment No 6

Title : Java Programs Using Abstract Keyword, Final Keyword, and Interfaces

Code :

```
interface Shape {
   double pi = 3.14159;
   void draw();
   double area();
}

abstract class Figure implements Shape {
   final int sides;
   Figure(int s) {
      this.sides = s;
   }
   abstract void color();

   public void displaySides() {
      System.out.println("Number of sides: " + sides);
   }
}

class Circle extends Figure {
   double r;

   Circle(double radius) {
      super(0);
      this.r = radius;
   }

   @Override
   public void draw() {
      System.out.println("Drawing Circle");
   }

   @Override
   public double area() {
      return Shape.pi * r * r;
   }

   @Override
   void color() {
      System.out.println("Circle is Red.");
   }

   public static void main(String[] args) {
      final String GREETING = "Shape Demo";
      System.out.println(GREETING);
```

```
        Circle c = new Circle(7.0);
        c.draw();
        System.out.println("Area of Circle: " + c.area());
        c.color();
        c.displaySides();
        System.out.println("Value of pi from interface: " + Shape.pi);
    }
}
```

Output :

```
Shape Demo
Drawing Circle
Area of Circle: 153.93791
Circle is Red.
Number of sides: 0
Value of pi from interface: 3.14159
```

Experiment No. 7

Title : Implement basic java programs based on arrays.

Code :

```
import java.util.Scanner;
import java.io.*;

class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter no.of value in it : ");
        int n = sc.nextInt();

        int[] arr = new int[n];
        for (int i=0; i<n ; i++ ){
            arr[i] = sc.nextInt();
        }


        System.out.println("You entered:");
        for (int num : arr) {
            System.out.print(num + " ");
        }

        sc.close();

    }
}
```

Output :

```
Enter no.of value in it : 5
2
3
1
2
3
You entered:
2 3 1 2 3
```

Experiment No. 8

Title : Implement a Java program to demonstrate Exception handling.

Code :

```
import java.util.Scanner;
import java.io.*;

class Main {
   public static void main(String[] args) {
      System.out.println("Hello, Lets check if there is any bug in following Lines \n");
         String  number = "hello";
         int[] arr = {1, 2, 3};
         int n = arr.length;
         int idx = 5 ;
      try {         //  Runtime error: ArrayIndexOutOfBoundsException
         System.out.println("Accessing invalid index: " + arr[idx]);
      } catch ( Exception e ) {
         System.out.println("Something went wrong.");
      } finally {
         System.out.println("finally - The 'try catch' is finished \n\n");
      }
      if ( idx > 3 ) { throw new ArrayIndexOutOfBoundsException(" \n
         CUSTOM ERROR - Assessing element is out of bound of array you have given ") ; }
         // using throw keyword
   }
}
```

Output :

```
Hello, Lets check if there is any bug in following Lines

ERROR!
Something went wrong.
finally - The 'try catch' is finished


Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException:
 CUSTOM ERROR - Assessing element is out of bound of array you have given
    at Main.main(Main.java:25)
```

Experiment No. 9

Title : Implement a Java program to demonstrate Exception handling.

a) Pre-defined package usage (`java.util`)

```
import java.util.ArrayList;                // Using a pre-defined package: java.util

public class PredefinedPackageDemo {
    public static void main(String[] args) {
        ArrayList<String> names = new ArrayList<>();
        names.add("Alice");
        names.add("Bob");
        names.add("Charlie");

        System.out.println("List of names:");
        for (String name : names) {
            System.out.println(name);
        }
    }
}
```

b) User-defined package

Step 1: Create a package named `mypackage` *File: `mypackage/Greeting.java`*

```
package mypackage;

public class Greeting {
    public void sayHello() {
        System.out.println("Hello from user-defined package!");
    }
}
```

Step 2: Use the package in another file *File: `Main.java`*

```
import mypackage.Greeting;

public class Main {
    public static void main(String[] args) {
        Greeting greet = new Greeting();
        greet.sayHello();
    }
}
```