**Batch: B3 Roll No.: 121**

**Experiment / assignment / tutorial No.01**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **TITLE : Complex Arithmetic** |

**AIM:** Write a program to perform addition, subtraction, multiplication and division of two complex numbers. Demonstrate the use of classes and objects.

Variations :

Implementation of Program with One class

Accessibility with static and non-static methods within class and outside class.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected OUTCOME of Experiment:**

**CO2:** Explore arrays, vectors, classes and objects in C++ and Java **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. E. Balagurusamy , “Programming with Java” McGraw-Hill.
2. Sachin Malhotra, Saurabh Choudhary, “Programming in Java”, Oxford Publications.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Pre Lab/ Prior Concepts:**

The Scanner class is a class in java.util, which allows the user to read values of various types. There are far more methods in class Scanner than you will need in this course. We only cover a small useful subset, ones that allow us to read in numeric values from either the keyboard or file without having to convert them from strings and determine if there are more values to be read.

Scanner in = new Scanner(System.in);  // System.in is an InputStream

 Numeric and String Methods

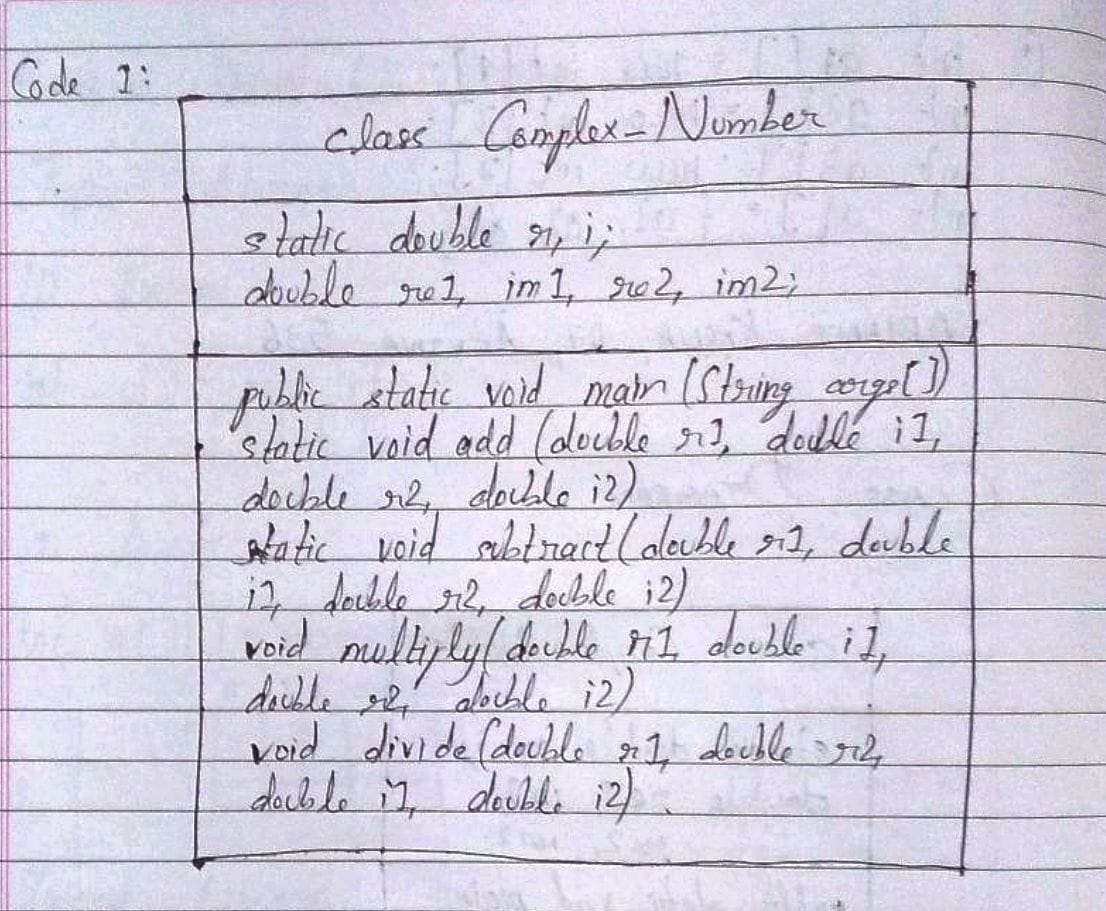
|  |  |
| --- | --- |
| **Method** | **Returns** |
| int nextInt() | Returns the next token as an int. If the next token is not an integer,InputMismatchException is thrown. |
| long nextLong() | Returns the next token as a long. If the next token is not an integer,InputMismatchException is thrown. |
| float nextFloat() | Returns the next token as a float. If the next token is not a float or is out of range, InputMismatchException is thrown. |
| double nextDouble() | Returns the next token as a long. If the next token is not a float or is out of range, InputMismatchException is thrown. |
| String next() | Finds and returns the next complete token from this scanner and returns it as a string; a token is usually ended by whitespace such as a blank or line break. If not token exists,NoSuchElementException is thrown. |
| String nextLine() | Returns the rest of the current line, excluding any line separator at the end. |
| void close() | Closes the scanner. |

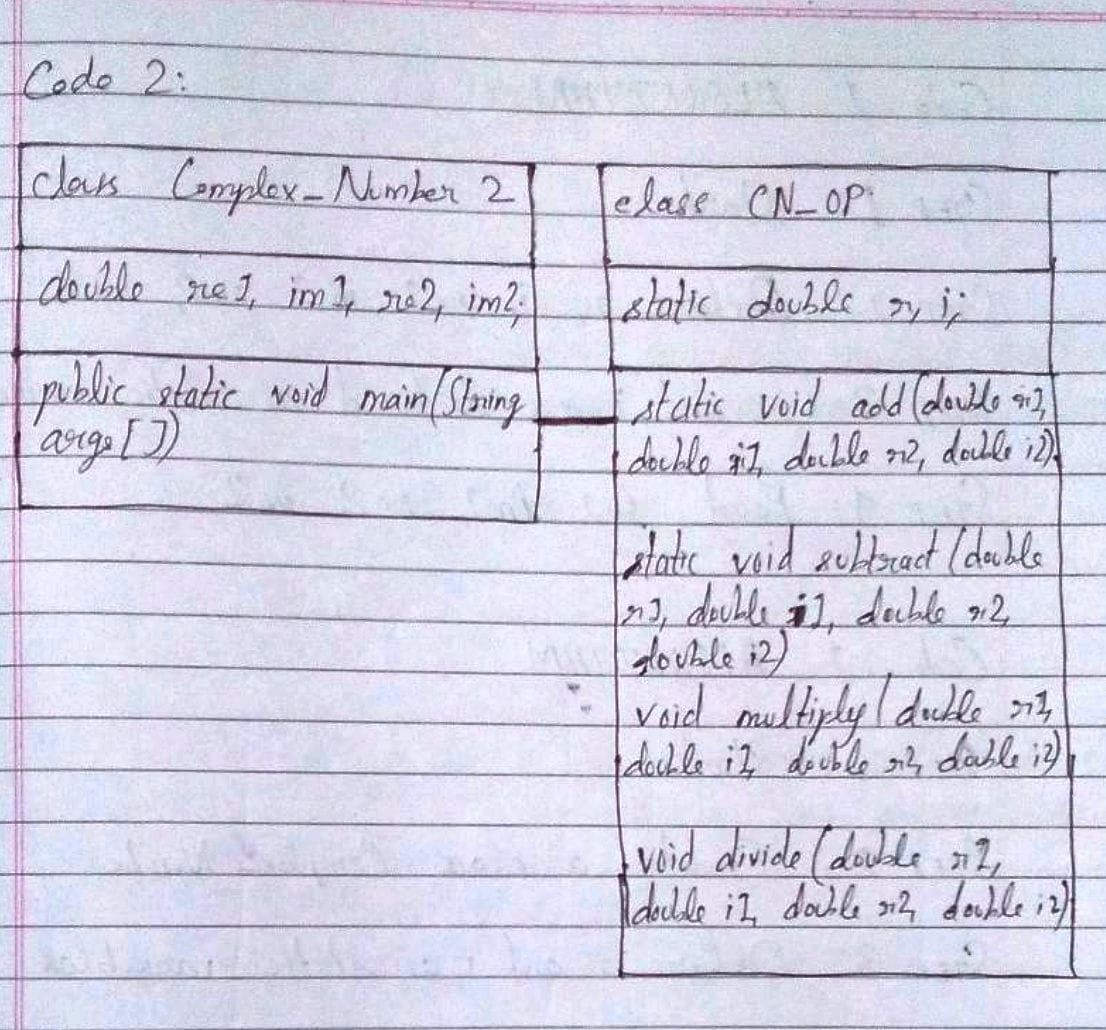
The Scanner looks for tokens in the input. A token is a series of characters that ends with what Java calls whitespace. A whitespace character can be a blank, a tab character, a carriage return. Thus, if we read a line that has a series of numbers separated by blanks, the scanner will take each number as a separate token. .

The numeric values may all be on one line with blanks between each value or may be on separate lines.   Whitespace characters (blanks or carriage returns) act as separators.  The next method returns the next input value as a string, regardless of what is keyed.  For example, given the following code segment and data

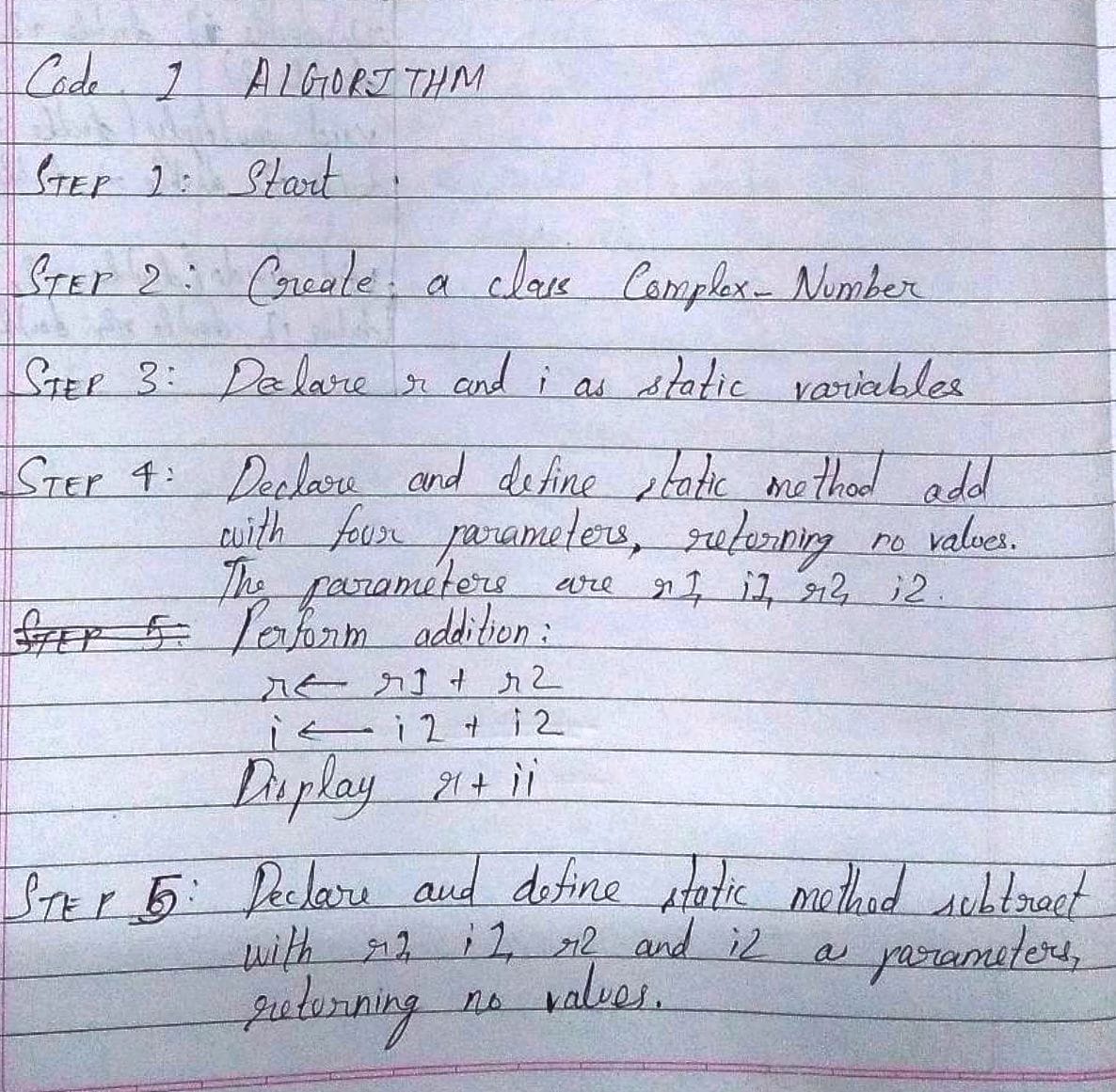
* int number = in.nextInt();
* float real = in.nextFloat();
* long number2 = in.nextLong();
* double real2 = in.nextDouble();
* String string = in.next();

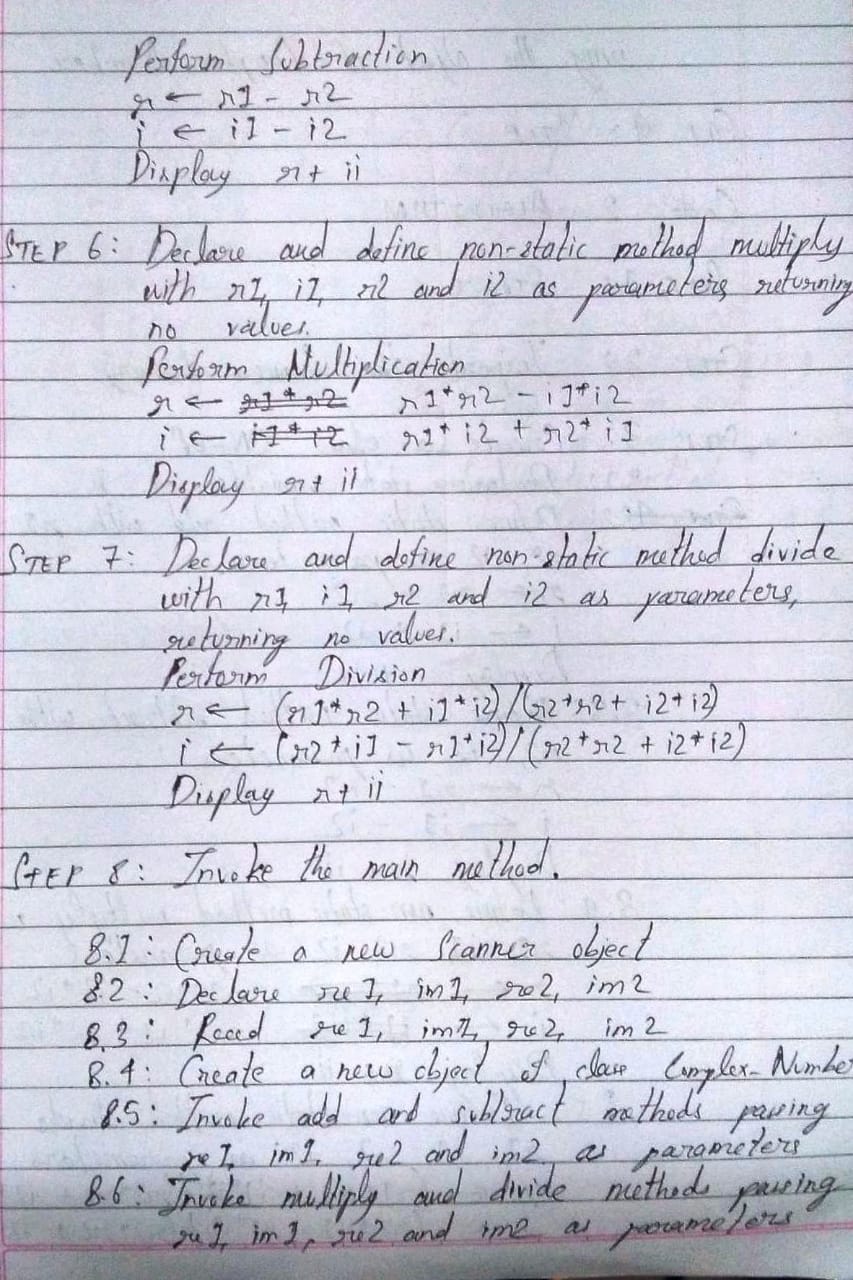
**Class Diagram:**

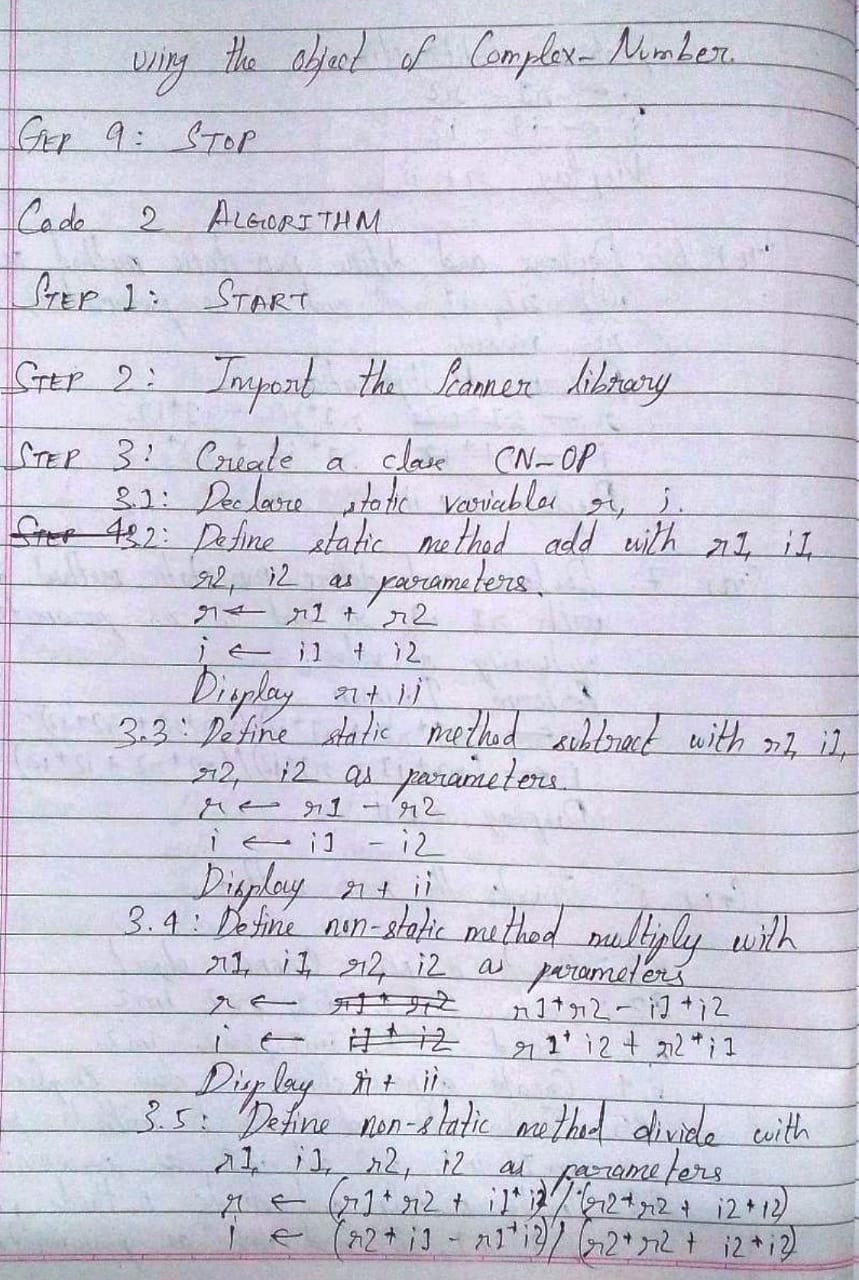


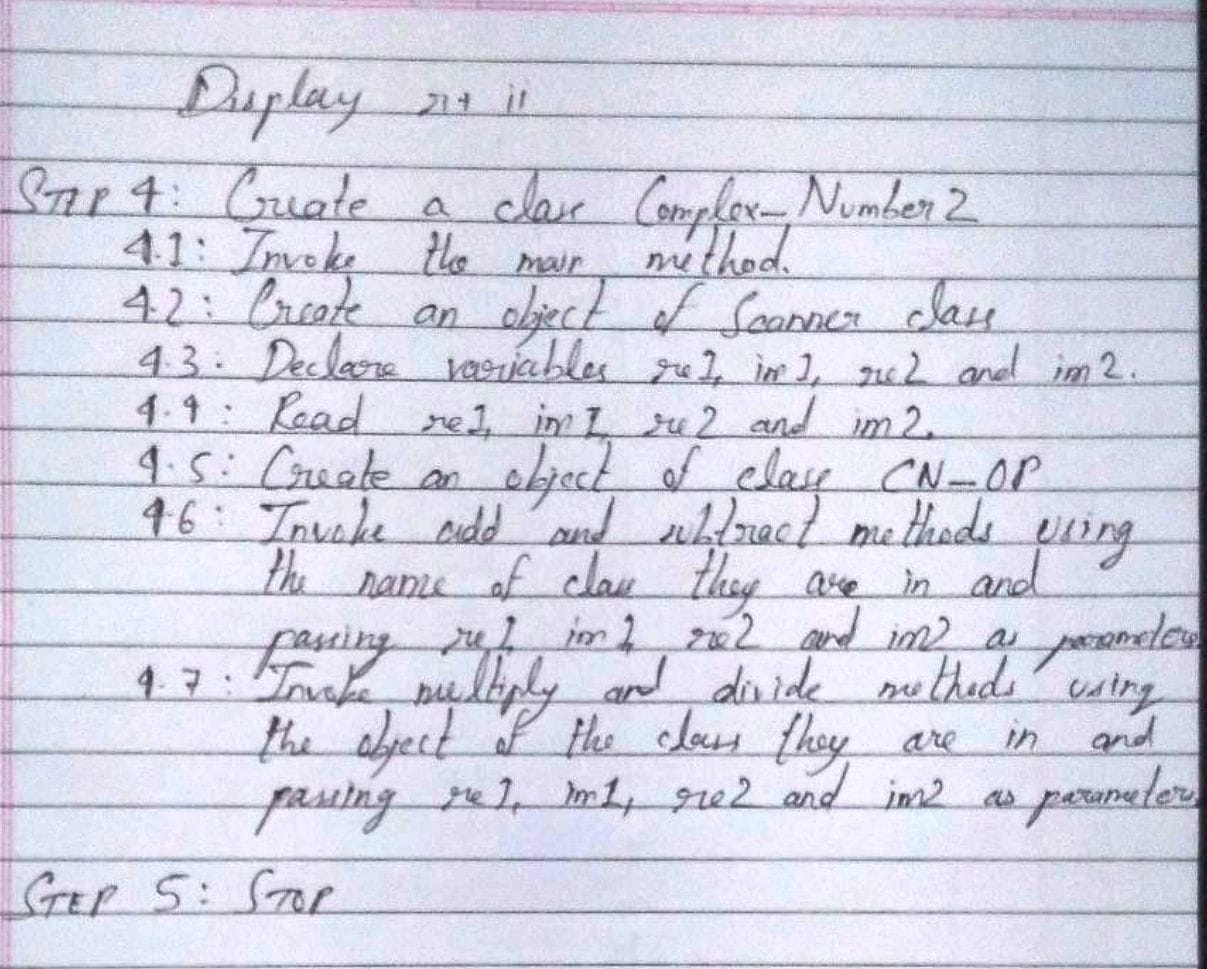


**Algorithm:**









**Implementation details:**

Code 1:

import java.util.\*;  
class Complex\_Number  
{  
static double r, i;  
static void add(double r1, double i1, double r2, double i2)  
{  
r = r1 + r2;  
i = i1 + i2;  
System.out.println("The sum of "+r1+"+i"+i1+" and "+r2+"+i"+i2+" is "+r+"+i"+i);  
}  
static void subtract(double r1, double i1, double r2, double i2)  
{  
r = r1 - r2;  
i = i1 - i2;  
System.out.println("The difference of "+r1+"+i"+i1+" and "+r2+"+i"+i2+" is "+r+"+i"+i);  
}  
void multiply(double r1, double i1, double r2, double i2)  
{  
r = (r1\*r2 - i1\*i2);

i = (r2\*i1 + r1\*i2);  
System.out.println("The product of "+r1+"+i"+i1+" and "+r2+"+i"+i2+" is "+r+"+i"+i);  
}  
void divide(double r1, double i1, double r2, double i2)  
{  
r = (r1\*r2 + i1\*i2)/(r2\*r2 + i2\*i2);  
i = (r2\*i1 - r1\*i2)/(r2\*r2 + i2\*i2);  
System.out.println("The quotient of "+r1+"+i"+i1+" and "+r2+"+i"+i2+" is "+r+"+i"+i);  
}  
public static void main(String args[])  
{  
Scanner sc = new Scanner(System.in);  
double re1, im1, re2, im2;  
System.out.println("In this program, the accessibility of static and non-static methods inside a class is shown.");  
System.out.println("Enter the real part of the first complex number: ");  
re1 = sc.nextDouble();  
System.out.println("Enter the imaginary part of the first complex number: ");  
im1 = sc.nextDouble();  
System.out.println("Enter the real part of the second complex number: ");  
re2 = sc.nextDouble();  
System.out.println("Enter the imaginary part of the second complex number: ");  
im2 = sc.nextDouble();  
Complex\_Number obj = new Complex\_Number();  
add(re1, im1, re2, im2); //accessing a static method; object is not required  
subtract(re1, im1, re2, im2); //accessing a static method; object is not required  
obj.multiply(re1, im1, re2, im2); //accessing a non-static method; object is required  
obj.divide(re1, im1, re2, im2); //accessing a non-static method; object is required  
}  
}

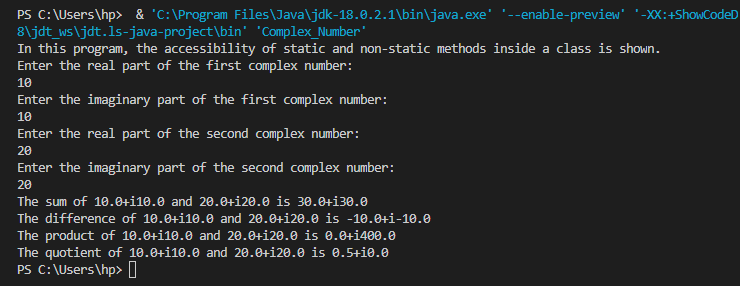
Code 2:

import java.util.\*;  
class CN\_OP  
{  
static double r, i;  
static void add(double r1, double i1, double r2, double i2)  
{  
r = r1 + r2;  
i = i1 + i2;  
System.out.println("The sum of "+r1+"+i"+i1+" and "+r2+"+i"+i2+" is "+r+"+i"+i);  
}  
static void subtract(double r1, double i1, double r2, double i2)  
{  
r = r1 - r2;  
i = i1 - i2;  
System.out.println("The difference of "+r1+"+i"+i1+" and "+r2+"+i"+i2+" is "+r+"+i"+i);  
}  
void multiply(double r1, double i1, double r2, double i2)  
{  
r = (r1\*r2 - i1\*i2);

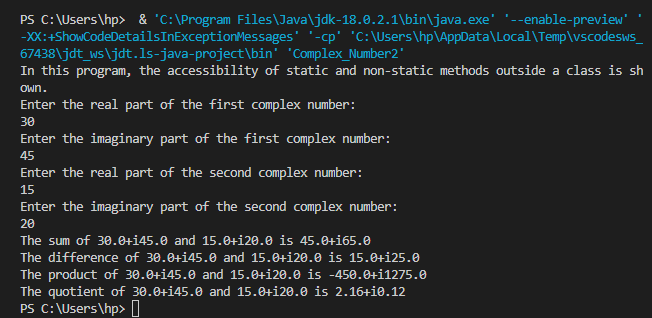
i = (r2\*i1 + r1\*i2);  
System.out.println("The product of "+r1+"+i"+i1+" and "+r2+"+i"+i2+" is "+r+"+i"+i);  
}  
void divide(double r1, double i1, double r2, double i2)  
{  
r = (r1\*r2 + i1\*i2)/(r2\*r2 + i2\*i2);  
i = (r2\*i1 - r1\*i2)/(r2\*r2 + i2\*i2);  
System.out.println("The quotient of "+r1+"+i"+i1+" and "+r2+"+i"+i2+" is "+r+"+i"+i);  
}  
}  
class Complex\_Number2  
{  
public static void main(String args[])  
{  
Scanner sc = new Scanner(System.in);  
double re1, im1, re2, im2;  
System.out.println("In this program, the accessibility of static and non-static methods outside a class is shown.");  
System.out.println("Enter the real part of the first complex number: ");  
re1 = sc.nextDouble();  
System.out.println("Enter the imaginary part of the first complex number: ");  
im1 = sc.nextDouble();  
System.out.println("Enter the real part of the second complex number: ");  
re2 = sc.nextDouble();  
System.out.println("Enter the imaginary part of the second complex number: ");  
im2 = sc.nextDouble();  
CN\_OP ob = new CN\_OP();  
CN\_OP.add(re1, im1, re2, im2); //accessing static method from outside the class  
CN\_OP.subtract(re1, im1, re2, im2); //accessing static method from outside the class  
ob.multiply(re1, im1, re2, im2); //accessing non-static method from outside class  
ob.divide(re1, im1, re2, im2); //accessing non-static method from outside class  
}  
}

**Output:**

Output for code 1:



Output for code 2:



**Conclusion:**

Thus, in this experiment, we learnt how to accept input of various data types from the user using Scanner class. Further, we learnt about static and non-static methods and the method of calling them from the main class (i.e., the class which has got the “public static void main()” statement and object declaration of the Scanner class). It was found that to call a non-static method which is within a class, an object of that class is required. But that is not the case with a static method. On the other hand, to call a static method which belongs to a class other than the main class, it is necessary to mention the name of the class which has the method, followed by the dot operator and then the method name in order to call that method from the main class while to call a non-static method belonging to another class, an object of that class is required.

**Date: \_\_17-09-22\_\_ Signature of faculty in-charge**

**Post Lab Descriptive Questions:**

Q.1 Write a program to find the area and circumference of a circle using two classes.

Ans.

import java.util.\*;

class calculation

{

    void area(int R)

    {

        double ar;

        ar = 3.14\*R\*R;

        System.out.println("The area of the circle is: "+ar);

    }

    void perimeter(int R)

    {

        double pr;

        pr = 2\*3.14\*R;

        System.out.println("The perimeter of the circle is: "+pr);

    }

}

class circle

{

    public static void main(String args[])

    {

        Scanner sc = new Scanner(System.in);

        int r;

        System.out.println("Enter the radius of the circle: ");

        r = sc.nextInt();

        calculation ob = new calculation();

        ob.area(r);

        ob.perimeter(r);

    }

}

Q.2 Write the output of following program

1. **public** **class** BreakExample2 {
2. **public** **static** **void** main(String[] args) {
3. //outer loop
4. **for**(**int** i=1;i<=3;i++){
5. //inner loop
6. **for**(**int** j=1;j<=3;j++){
7. **if**(i==2&&j==2){
8. //using break statement inside the inner loop
9. **break**;
10. }
11. System.out.println(i+" "+j);
12. }
13. }
14. }
15. }

**Output:**

1 1

1 2

1 3

2 1

3 1

3 2

3 3

Q.3 Why is Java known as a platform independent language?

Ans. When it is said that Java is a platform independent language, it implies that java code can be executed on any machine having any operating system. Normally, programs are written in High Level Languages which can be read by the programmer. However, the CPU cannot directly understand such code. The written code has to be converted to Machine Level Language (binary digits of 1s and 0s) in order for the CPU to be able to understand it. Thus, a compiler translates the source code into machine level code. Either the source code is directly converted to machine code, or it is converted to an intermediate representation that is interpreted by a virtual machine. This intermediate representation in Java is the Java Byte Code. The Byte Code makes Java platform independent. Each system has its own JVM (Java Virtual Machine) which gets installed automatically when the JDK (Java Development Kit) software is installed. For every operating system, separate JVM is available which is suited for that operating system. Thus, while installing JDK, the user is asked to download the file which is suitable for the Operating System currently in use on that device. Thus, though Java itself is platform independent, the JVM is not.

Q.4 Write a recursive static method for calculation of gcd of a number.

Ans.

int G = 1;

void g\_c\_d(int A, int B)

{

for(int i = 1; i <= A && i <= B; i++)

{

if(A%i==0 && B%i==0)

G = i;

}

System.out.println("The GCD of "+A+" and "+B+" is: "+G);

}