

Assignment -1

Q-1:- Define Object Oriented Programming. Describe four Pillars of OOP.

Definition of OOP:

Object Oriented Programming (OOP) is a paradigm that organizes software design around objects and classes.

Four Pillars of OOP:

A. Encapsulation

- Encapsulation means binding data methods together into a single unit and restricting direct access to data.

Example :-

```
class student {
    private String name; // hidden data
    public void setName (String n)
```

```
        name = n;
```

```
        public String getName ()
```

```
        return name;
```

```
}
```

B. Abstraction :-

- Abstraction means hiding implementation details and showing only necessary features.

Example:-

abstract class Animal {

} abstract void sound(); // only concept.

}

class Dog extends Animal {

} void sound() { }

Hybrid system.out.println("Dog barks");

fins dog's first output

public class Main {

} public static void main (String [] args) { }

Animal a = new Dog(); // abstraction

a.sound();

g
G

animal prints barking

animal prints

C. Inheritance.

- Inheritance means one class can acquire the properties and methods of another class. This promotes code reuse.

Example :-

```
class vehicle
{
    void start()
    {
        System.out.println("Vehicle starts");
    }
}

class car extends vehicle
{
    void drive()
    {
        System.out.println("Car is driving");
    }
}

public class Main
{
    public static void main(String[] args)
    {
        Car c = new Car();
        c.start(); // inheritance
        c.drive(); // own method.
    }
}
```

D. Polymorphism

- Polymorphism means one name, many forms.

Example:-

1. compile polymorphism

public class Main

{ public static void main(String[] args)

 MathOperation m = new MathOperation();

 System.out.println(m.add(5, 10));

 System.out.println(m.add(2.5, 3.5));

}

() subb bion

2. Running Polymorphism

public class Main

{ public static void main(String[] args)

 Animal a = new Cat();

 a.sound(); // Cat's version executed

Conclusion:-

The four pillars of OOP makes programming modular, reusable, secure, easier to maintain.

OOP is widely used in real world applications like banking systems, ecommerce platforms.

Q-2:- Describe compilation and execution of a Java program with a diagram.

Introduction:- Java is a high level, object oriented and platform independent programming language. The execution of a java program involves two major steps: compilation and execution.

Steps of compilation and Execution:-

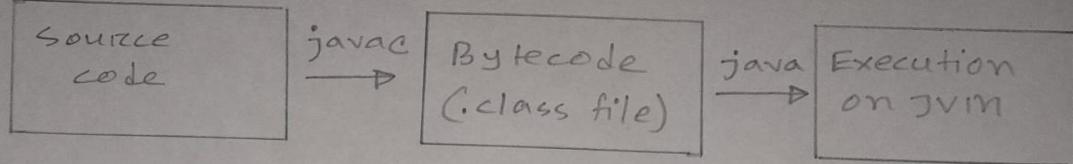
I. Writing the program

- Java programs are written using a text editor.
- The program is saved with the .java extension.

Example: Hellow.java.

```
class Hellow
{
    public static void main(String[] args)
    {
        System.out.println("Hellow, Java!");
    }
}
```

Diagram :- compilation and Execution process.



1. javac → compiler translates .java file into .class
2. java → JVM interprets the bytecode and execution
3. Java is platform-independent because bytecode can run on any system with a JVM.

Conclusion :- The process of java program execution is divided into two phases, compilation and execution, compilation converts human readable source code into platform-independent bytecode while execution is carried out by the JVM.

Q-3:- Trace the output including simulation of following problem.

Trace and simulation of java program.

Program.

public class Assignment

{ public static void main(String[] args)

{ int x=5, z=3;
int y=x++--z*--z;

for(int i=7; i<=3; i++)

{ x=(x++>y)?(x-y):(++y-i);

System.out.println(x+" "+y+" ");

}

System.out.println("In final values:

x=" +x+", y=" +y")

}

2. compilation :-

javac Hellow.java

- The java compiler (javac) translates the source code into bytecode.
- The bytecode is stored in a file with a .class extension.

3. Execution :-

java Hellow

- The java virtual machine (JVM) executes the bytecode.
- JVM converts bytecodes into machine code using an interpreter or JIT (Just-In-Time) compiler.
- output.

Hellow, Java!

Step-1 :- Initialization

$$x = 5, \quad z = 3$$

Step-2 :- Evaluate y

$$y = x++ - -z * --z;$$

$$1. \quad x++ = 5 \quad (\text{then } x = 6)$$

$$2. \quad - -z = 2$$

$$3. \quad \text{second } - -z = 1$$

$$4. \quad \text{so, } \rightarrow y = 5 - (2 * 1) = 3$$

Now,

$$x = 6, \quad y = 3, \quad z = 2$$

Step-3 :- Loop simulation

Iteration (i) Before (i, y) $x++ > y$ Action After Print

$$i = 1 \quad (6, 3) \quad 6 > 3 \rightarrow \text{true} \quad x = x - y \quad (4, 3) \quad 4, 3 \\ = 7 - 3 = 4$$

$$i = 2 \quad (4, 3) \quad 4 > 3 \rightarrow \text{true} \quad x = x - y \quad (2, 3) \quad 2, 3 \\ = 5 - 3 = 2$$

$$i = 3 \quad (2, 3) \quad 2 > 3 \rightarrow \text{false} \quad t + y - i \quad (1, 4) \quad 1, 4 \\ = 4 - 3 = 1$$

final output

$$43 \mid 23 \mid 14$$

1. Print the Factor:



Screenshot of Apache NetBeans IDE 26 showing Java code for printing factors and its execution output.

ConditionalOperation.java (Source tab):

```
1 package conditionaloperation;
2
3 public class ConditionalOperation {
4
5     public static void main(String[] args) {
6         double a = 10, b = 20;
7
8         System.out.println("Enter the value");
9         double large;
10        large = (a > b) ? a : b;
11        System.out.println(large);
12    }
13
14 }
15
```

Output - Factors (run):

```
run:
Enter a number: 55
Factors of 55:
1
5
11
55
BUILD SUCCESSFUL (total time: 4 seconds)
```

Windows taskbar at the bottom:

```
Type here to search 2|3 Rain... 1:57 AM 12:6 INS Windows (CRLF)
[Icons] 1|1
```

2. Print the PowerCalculate:



Screenshot of Apache NetBeans IDE 26 showing Java code for power calculation and its execution output.

PowerCalc.java (Source tab):

```
1 package powercalc;
2
3 import java.util.Scanner;
4
5 public class PowerCalc {
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         System.out.print("Enter base: ");
9         int base = sc.nextInt();
10        System.out.print("Enter exponent: ");
11        int exp = sc.nextInt();
12
13        int result = 1;
14        for (int i = 1; i <= exp; i++) {
15            result *= base;
16        }
17
18        System.out.println(base + "^" + exp + " = " + result);
19    }
20 }
```

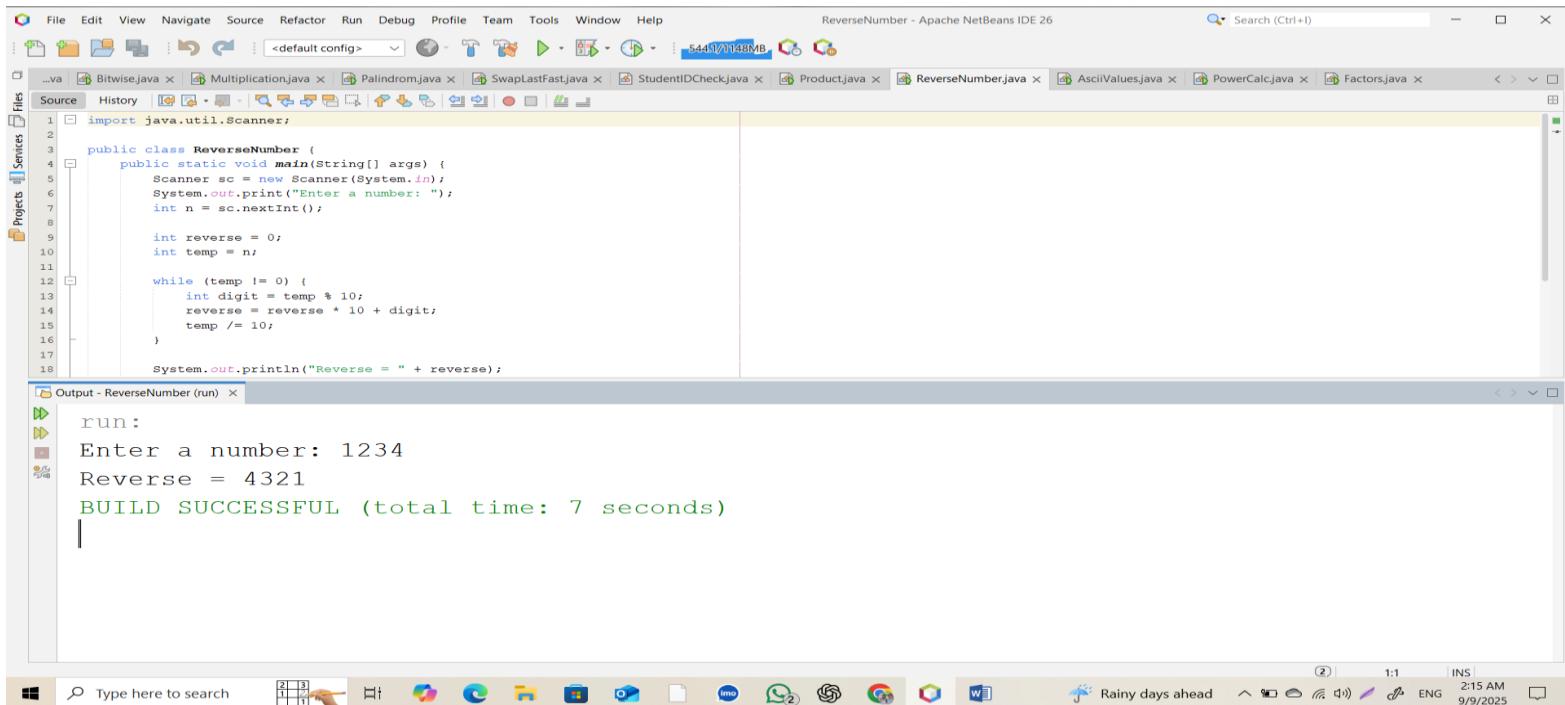
Output - PowerCalc (run):

```
run:
Enter base: 10
Enter exponent: 2
10^2 = 100
BUILD SUCCESSFUL (total time: 21 seconds)
```

Windows taskbar at the bottom:

```
Type here to search 2|3 Rain... 29°C Haze 3:1 2:11 AM INS ENG 9/9/2025
```

3 .Print the Reverse Number :



Apache NetBeans IDE 26

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Source History Projects Services Files

```
1 import java.util.Scanner;
2
3 public class ReverseNumber {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Enter a number: ");
7         int n = sc.nextInt();
8
9         int reverse = 0;
10        int temp = n;
11
12        while (temp != 0) {
13            int digit = temp % 10;
14            reverse = reverse * 10 + digit;
15            temp /= 10;
16        }
17
18        System.out.println("Reverse = " + reverse);
19    }
20 }
```

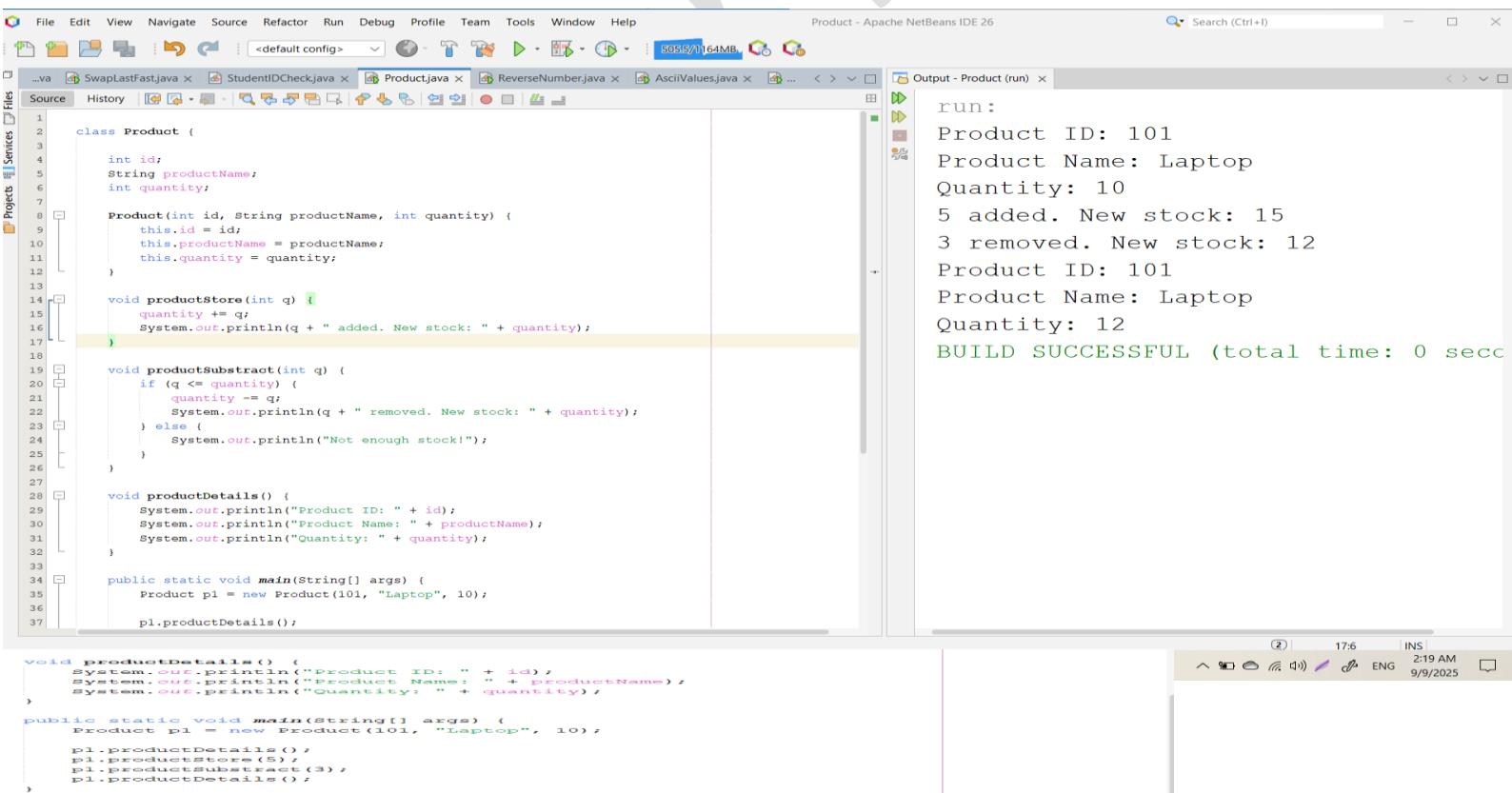
Output - ReverseNumber (run) X

```
run:
Enter a number: 1234
Reverse = 4321
BUILD SUCCESSFUL (total time: 7 seconds)
```

Type here to search

System tray icons: Rainy days ahead, ENG, 2:15 AM, 9/9/2025

4. Print the Product :



Apache NetBeans IDE 26

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Source History Projects Services Files

```
1 class Product {
2
3     int id;
4     String productName;
5     int quantity;
6
7     Product(int id, String productName, int quantity) {
8         this.id = id;
9         this.productName = productName;
10        this.quantity = quantity;
11    }
12
13    void productStore(int q) {
14        quantity += q;
15        System.out.println(q + " added. New stock: " + quantity);
16    }
17
18    void productSubtract(int q) {
19        if (q <= quantity) {
20            quantity -= q;
21            System.out.println(q + " removed. New stock: " + quantity);
22        } else {
23            System.out.println("Not enough stock!");
24        }
25    }
26
27    void productDetails() {
28        System.out.println("Product ID: " + id);
29        System.out.println("Product Name: " + productName);
30        System.out.println("Quantity: " + quantity);
31    }
32
33    public static void main(String[] args) {
34        Product p1 = new Product(101, "Laptop", 10);
35
36        p1.productDetails();
37    }
38
39    void productDetails() {
40        System.out.println("Product ID: " + id);
41        System.out.println("Product Name: " + productName);
42        System.out.println("Quantity: " + quantity);
43    }
44
45    public static void main(String[] args) {
46        Product p1 = new Product(101, "Laptop", 10);
47
48        p1.productDetails();
49        p1.productStore(5);
50        p1.productSubtract(3);
51        p1.productDetails();
52    }
53 }
```

Output - Product (run) X

```
run:
Product ID: 101
Product Name: Laptop
Quantity: 10
5 added. New stock: 15
3 removed. New stock: 12
Product ID: 101
Product Name: Laptop
Quantity: 12
BUILD SUCCESSFUL (total time: 0 seconds)
```

Type here to search

System tray icons: Rainy days ahead, ENG, 2:19 AM, 9/9/2025

5.Print the StudentIDCheak



StudentIDCheak - Apache NetBeans IDE 26

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Source History

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter last 4 digits of Student ID: ");
    int id = sc.nextInt();

    int reverse = 0, temp = id;
    while (temp != 0) {
        reverse = reverse * 10 + temp % 10;
        temp /= 10;
    }

    if (id == reverse) {
        System.out.println(id + " is a Palindrome");
    } else {
        System.out.println(id + " is NOT a Palindrome");
    }

    boolean isPrime = true;
    if (reverse < 2) {
        isPrime = false;
    } else {
        for (int i = 2; i <= reverse / 2; i++) {
            if (reverse % i == 0) {
                isPrime = false;
                break;
            }
        }
    }

    if (isPrime) {
        System.out.println("Reverse number " + reverse + " is Prime");
    } else {
        System.out.println("Reverse number " + reverse + " is NOT Prime");
    }
    sc.close();
}
```

Output - StudentIDCheak (run) x

run:
Enter last 4 digits of Student ID: 1169
1169 is NOT a Palindrome
Reverse number 9611 is NOT Prime
BUILD SUCCESSFUL (total time: 15 seconds)

Type here to search 29°C Haze 8:1 229 AM 9/9/2025

6.Print the SwapLastFast



SwapLastFast - Apache NetBeans IDE 26

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Source History

```
import java.util.Scanner;

public class SwapLastFast {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();

        int num = n;
        int last = num % 10;

        int first = num;
        int count = 0;
        while (first >= 10) {
            first /= 10;
            count++;
        }

        int middle = n % (int) Math.pow(10, count);
        middle /= 10;

        int swapped = last * (int) Math.pow(10, count) + (middle * 10) + first;

        System.out.println("Number after swapping = " + swapped);
        sc.close();
    }
}
```

Output - SwapLastFast (run) x

run:
Enter a number: 50
Number after swapping = 5
BUILD SUCCESSFUL (total time: 4 seconds)

27:20 9/9/2025

7.Print the Palindrome



Apache NetBeans IDE 26

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Source History

```
1 public class Palindrom {
2     public static void main(String[] args) {
3         int num = 121;
4         int original = num;
5         int rev = 0;
6
7         while (num != 0) {
8             int digit = num % 10;
9             rev = rev * 10 + digit;
10            num = num / 10;
11        }
12
13        System.out.println(original);
14    }
15
16
17 }
18 }
```

Output - Palindrom (run) ×

```
run:
121
BUILD SUCCESSFUL (total time: 0 seconds)
```

18:2 INS

8. Print the Multiplication



Apache NetBeans IDE 26

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Source History

```
1 package multiplication;
2
3
4 public class Multiplication {
5
6     public static void main(String[] args) {
7         int num = 5;
8         int i = 1;
9         System.out.println("Enter the Multiple ");
10        for(i = 1; i<=10; i++)
11        {
12            System.out.println(num + "X" + i+ "==" + num*i);
13        }
14    }
15
16 }
17 }
```

Output - Multiplication (run) ×

```
run:
Enter the Multiple
5X1=5
5X2=10
5X3=15
5X4=20
5X5=25
5X6=30
5X7=35
5X8=40
5X9=45
5X10=50
BUILD SUCCESSFUL (total time: 0 seconds)
```

finished building Multiplication (run). Type here to search 2:39 AM 9/9/2025

9. Print the Bitwise

The screenshot shows the Apache NetBeans IDE interface. The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar says "Bitwise - Apache NetBeans IDE 26". The source code editor contains a Java file named Bitwise.java:

```
1 package bitwise;
2
3 public class Bitwise {
4
5     public static void main(String[] args) {
6         int a = 51, b= 36;
7         int c = a & b;
8         System.out.println("Enter the AND ");
9         System.out.println("A & B = " + c);
10        c = a | b;
11        System.out.println("Enter the OR ");
12        System.out.println("A | B = " + c);
13        c = a ^ b;
14        System.out.println("Enter The EXOR ");
15        System.out.println("A ^ B =" + c);
16    }
17
18 }
```

The output window shows the execution results:

```
run:
Enter the AND
A & B = 32
Enter the OR
A | B = 55
Enter The EXOR
A ^ B =23
BUILD SUCCESSFUL (total time: 0 seconds)
```

The taskbar at the bottom shows various application icons, and the system tray indicates "Rain coming".

10. Print the ConditionalOperation

The screenshot shows the Apache NetBeans IDE interface. The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar says "ConditionalOperation - Apache NetBeans IDE 26". The source code editor contains a Java file named ConditionalOperation.java:

```
1 package conditionaloperation;
2
3 public class ConditionalOperation {
4
5     public static void main(String[] args) {
6         double a = 10, b = 20;
7
8         System.out.println("Enter the value");
9         double large;
10        large = (a > b) ? a : b;
11        System.out.println(large);
12    }
13
14
15 }
```

The output window shows the execution results:

```
run:
Enter the value
20.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

The taskbar at the bottom shows various application icons, and the system tray indicates "Rain coming".

11. Print the Factorials

The screenshot shows the Apache NetBeans IDE interface. The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar says "Factorials - Apache NetBeans IDE 26". The source code editor displays a Java file named Factorials.java:

```
1 package factorials;
2
3 public class Factorials {
4
5     public static void main(String[] args) {
6
7         System.out.println("Enter the value ");
8         int i=1, fact = 1;
9         do {
10             System.out.println(fact);
11             fact = fact * i;
12             ++i;
13         } while(i<=6);
14     }
15 }
16
17
18
19
```

The output window titled "Output - Factorials (run)" shows the execution results:

```
run:
Enter the value
1
1
2
6
24
120
BUILD SUCCESSFUL (total time: 0 seconds)
```

The taskbar at the bottom shows various application icons, and the system tray indicates it's "Very humid" with a temperature of 24°C.

12. Print the Number

The screenshot shows the Apache NetBeans IDE interface. The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar says "JavaApplication49 - Apache NetBeans IDE 26". The source code editor displays a Java file named JavaApplication49.java:

```
1 package javaapplication49;
2
3 public class JavaApplication49 {
4
5     public static void main(String[] args) {
6
7         System.out.println("Enter the value ");
8         int i = 1;
9         do {
10             System.out.println(i);
11             i++;
12         } while (i <= 10);
13     }
14 }
15
```

The output window titled "Output - JavaApplication49 (run)" shows the execution results:

```
run:
Enter the value
1
2
3
4
5
6
7
8
9
10
BUILD SUCCESSFUL (total time: 0 seconds)
```

The taskbar at the bottom shows various application icons, and the system tray indicates it's "28°C Haze" with a temperature of 28°C.

13. Print the Nunber cheak

The screenshot shows the Apache NetBeans IDE interface. The left pane displays the code for `NumberCheckUser.java`:1 package numbercheckuser;
2 import java.util.Scanner;
3 public class NumberCheckUser {
4
5 public static void main(String[] args) {
6 Scanner input = new Scanner(System.in);
7 System.out.println("Enter the value ");
8 int a = input.nextInt();
9
10 if(a < 0)
11 {
12 System.out.println("Negative");
13 }
14 if(a > 0)
15 {
16 System.out.println("Positive");
17 }
18 else
19 {
20 System.out.println("Invalid Number");
21 }
22 }
23 }The right pane shows the output window:run:
Enter the value
50
Positive
BUILD SUCCESSFUL (total time: 2 seconds)

14.Print the Cheak Smallest

The screenshot shows the Apache NetBeans IDE interface. The left pane displays the code for `Smallest.java`:1 package smallest;
2
3 public class Smallest {
4 public static void main(String[] args) {
5 char i = 'a';
6 char j = 'b';
7 char k = 'c';
8
9 char smallest = i;
10
11 if (j < smallest) {
12 smallest = j;
13 }
14 if(k < smallest) {
15 smallest = k;
16 }
17
18 System.out.println("Smallest Character: " + smallest);
19 }
20 }The right pane shows the output window:run:
Smallest Character: a
BUILD SUCCESSFUL (total time: 0 seconds)

15. Print the Prime Number

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar says "primeN - Apache NetBeans IDE 26". The left sidebar lists several Java files: SeriesJava, PrimeNumber.java, PrimeN.java, SmallestJava, and NumberCheckJava... The main source editor window displays the following Java code:

```
1 package primen;
2
3 public class PrimeN {
4
5     public static void main(String[] args) {
6         int n = 20;
7
8         boolean prime = true;
9         int i = 1;
10        System.out.println(n);
11        do {
12
13            if (n % i == 0) {
14                System.out.println(n + " is divided by " + i);
15            }
16            i++;
17        } while (i <= n);
18
19        if (prime) {
20            System.out.println(n + " is Prime ");
21        } else {
22            System.out.println(n + " Not Prime");
23        }
24    }
25
26
27 }
```

The right panel shows the "Output - primeN (run)" window with the following text:
run:
20
20 is divided by 1
20 is divided by 2

16. Print the Prime Number from user

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar says "PrimeNumber - Apache NetBeans IDE 26". The left sidebar lists several Java files: SeriesJava, PrimeNumber.java, PrimeN.java, SmallestJava, and NumberCheckJava... The main source editor window displays the following Java code:

```
1 package primenumber;
2
3 import java.util.Scanner;
4
5 public class PrimeNumber {
6
7     public static void main(String[] args) {
8
9         Scanner input = new Scanner(System.in);
10        int n = input.nextInt();
11
12        for (int num = 2; num <= n; num++) {
13            boolean prime = true;
14            for (int i = 2; i <= num / 2; i++) {
15                if (num % i == 0) {
16                    prime = false;
17                    break;
18                }
19            }
20            if (prime) {
21                System.out.println(num);
22            }
23        }
24    }
25
26 }
```

The right panel shows the "Output - PrimeNumber (run)" window with the following text:
run:
15
2
3
5
7
11
13
BUILD SUCCESSFUL (total time: 5 seconds)

17. Print the Series

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar says "Series - Apache NetBeans IDE 26". The left sidebar lists several Java files: PatternClass.java, PatternClass2.java, Factorial.java, Series.java, PrimeNumber...., and primeN.java. The main source editor window displays the following Java code:

```
1 package series;
2
3 import java.util.Scanner;
4
5 public class Series {
6
7     public static void main(String[] args) {
8         Scanner input = new Scanner(System.in);
9         System.out.print("Enter the value =");
10
11        int n = input.nextInt();
12        int i, N = 0;
13        for (i = 1; i <= n; i = i + 2) {
14            System.out.print(i + " ");
15
16            N = N + i;
17        }
18
19        System.out.println();
20        System.out.println(N);
21    }
22
23 }
24
```

The right panel shows the "Output - Series (run)" tab with the following text:

```
run:
Enter the value =15
1 3 5 7 9 11 13 15
64
BUILD SUCCESSFUL (total time: 3 seconds)
```

The taskbar at the bottom shows various application icons and the system clock.

18. Print the Factorials

The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar says "Factorial - Apache NetBeans IDE 26". The left sidebar lists several Java files: PatternClass.java, PatternClass2.java, Factorial.java, Series.java, PrimeNumber...., and primeN.java. The main source editor window displays the following Java code:

```
1 package factorial;
2
3 import java.util.Scanner;
4
5 public class Factorial {
6
7     public static void main(String[] args) {
8         Scanner input = new Scanner(System.in);
9
10        int n, i, fact = 1;
11        System.out.print("Entre the value ");
12        n = input.nextInt();
13        for (i = 1; i <= n; i++) {
14            fact = fact * i;
15
16            System.out.print(" ");
17            System.out.println(fact);
18        }
19
20    }
21
22 }
```

The right panel shows the "Output - Factorial (run)" tab with the following text:

```
run:
Entre the value 5
1
2
6
24
120
BUILD SUCCESSFUL (total time: 3 seconds)
```

The taskbar at the bottom shows various application icons and the system clock.

19. Print the Pattern

The screenshot shows the Apache NetBeans IDE interface. On the left, the code editor displays `PatternClass.java` with the following content:

```
1 package patternclass;
2
3 import java.util.Scanner;
4
5 public class PatternClass {
6
7     public static void main(String[] args) {
8         Scanner input = new Scanner(System.in);
9
10        int n = input.nextInt();
11        System.out.println(" Enter the value ");
12        for (int i = 1; i <= n; i++) {
13            for (int j = 1; j <= i; j++) {
14                System.out.print("*");
15            }
16            System.out.println();
17        }
18    }
19 }
20 }
```

The right side of the interface shows the `Output - PatternClass (run)` window with the following text:
run:
5
Enter the value
*
* *
* * *
* * * *
BUILD SUCCESSFUL (total time: 2 seconds)

20. Print the Small latter

The screenshot shows the Apache NetBeans IDE interface. On the left, the code editor displays `AbcdWhile.java` with the following content:

```
1 package abcdwhile;
2
3
4 public class AbcdWhile {
5
6
7     public static void main(String[] args) {
8         char i = 'a';
9         System.out.println("Enter the letter ");
10        while(i < 'z')
11        {
12            i++;
13            System.out.println(" " +i);
14        }
15    }
16 }
17
18 }
```

The right side of the interface shows the `Output - AbcdWhile (run)` window with the following text:
run:
Enter the letter
b
c
d
e
f
g
h
i
j
k
l
m
n
o
p
q
r
s

21. Print the Even or Odd

The screenshot shows the Apache NetBeans IDE interface. The left pane displays the code for a Java class named EvenOdd. The right pane shows the output window with the results of running the program. The code uses a Scanner to read an integer from the user and prints "Even" if it's divisible by 2, or "Odd" otherwise.

```
1 package evenodd;
2
3 import java.util.Scanner;
4
5 public class EvenOdd {
6
7     public static void main(String[] args) {
8         Scanner input = new Scanner(System.in);
9
10        double A, B;
11        A = input.nextInt();
12
13        if (A % 2 == 0) {
14            System.out.println("Even ");
15        } else {
16            System.out.println("Odd ");
17        }
18    }
19
20 }
```

Output - EvenOdd (run) x

```
run:
10
Even
BUILD SUCCESSFUL (total time: 3 seconds)
```

22. Print the Temperature

The screenshot shows the Apache NetBeans IDE interface. The left pane displays the code for a Java class named Temperature. The right pane shows the output window with the results of running the program. The code uses a Scanner to read a temperature value in Celsius and prints it converted to Fahrenheit using the formula $F = 1.8 * (C + 32)$.

```
1 package temperature;
2
3 import java.util.Scanner;
4
5 public class Temperature {
6
7     public static void main(String[] args) {
8         Scanner input = new Scanner(System.in);
9         System.out.print("Enter the value ");
10        double C,F;
11        C = input.nextDouble();
12        F = 1.8 * (C + 32);
13        System.out.println(F);
14    }
15
16 }
17
```

Output x

Temperature (run) x Temperature (run) #2 x

```
run:
Enter the value
20
93.60000000000001
BUILD SUCCESSFUL (total time: 3 seconds)
```

23. Print the Librarye

The screenshot shows the Eclipse IDE interface. On the left, the code editor displays a Java class named `Librarye` with methods for creating books and displaying their details. On the right, the `Output` view shows the program's execution results, including the output of two book objects and a successful build message.

```
1 package librarye;
2
3 public class Librarye {
4
5     String bookName;
6     String authorName;
7     int quantity;
8
9     Librarye(String bookName, String authorName, int quantity) {
10        this.bookName = bookName;
11        this.authorName = authorName;
12        this.quantity = quantity;
13    }
14
15    void display() {
16        System.out.println("\nBook Name: " + bookName
17                            + "\nAuthor Name : " + authorName + "\nQuantity : " + quantity);
18    }
19
20    public static void main(String[] args) {
21        Librarye book1 = new Librarye("Physics", "Alamgir", 3);
22        book1.display();
23        Librarye book2 = new Librarye("CSE", "Computer", 3);
24
25        book2.display();
26    }
27 }
28
```

```
run:
Book Name: Physics
Auther Name : Alamgir
Quantity : 3

Book Name: CSE
Auther Name : Computer
Quantity : 3
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name : Md Alamgir Hosen

ID: 02724205101169

Batch : 65th

Section: C

Depertment of CSE, City University

Github Link : <https://github.com/alamgirsu/alamgirsu2526.git>

Alamgir Hosen

Alamgir Hosen