Semester II

Semester 2	24BCA201T	Object Oriented Programming with Java (Theory)
2 Credits	TPE : 2:0:0	Core Course – CC

Course Outcomes: On successful completion of the course the learner will be able to

CO#	Course Outcomes
BCA201T.1	Explain the fundamental principles and concepts of object-oriented programming (OOP) in programming languages.
BCA201T.2	Design and develop distributed applications using Java, utilizing socket programming, RMI, or other distributed computing technologies.
BCA201T.3	Design and implement GUI components, layouts, and event handling mechanisms.

Object Oriented Programming with Java (Theory)

Course Content :

Unit 1: Introduction to Java (6 Hours)

Java and Java applications; Java Development Kit (JDK); Java is interpreted, Byte Code, JVM; Object-oriented programming; Simple Java programs. Data types and other tokens: Boolean variables, int, long, char, operators, arrays, white spaces, literals, assigning values; Creating and destroying objects; Access specifiers. Operators and Expressions: Arithmetic Operators, Bitwise operators, Relational operators, The Assignment Operator, The ? Operator; Operator Precedence; Logical expression; Type casting; Strings, Control Statements: Selection statements, iteration statements, Jump Statements.

Unit 2: Classes (6 Hours)

Classes in Java; Declaring a class; Class name; Super classes; Constructors; Creating instances of class; Inner classes. Inheritance: Simple, multiple, and multilevel inheritance; Overriding, overloading. Exception handling: Exception handling in Java. The Applet Class: Two types of Applets; Applet basics; Applet Architecture; An Applet skeleton; Simple Applet display methods; Requesting repainting; Using the Status Window; The HTML APPLET tag; Passing parameters to Applets; getDocumentbase() and getCodebase(); ApletContext and showDocument(); The AudioClip Interface; The AppletStub Interface; Output to the Console.

Unit 3: Multi Threaded Programming (6 Hours)

What are threads? How to make the classes threadable; Extending threads; Implementing runnable; Synchronization; Changing state of the thread; Bounded buffer problems, read-write problem, producer consumer problems. Event Handling: Two event handling mechanisms; The delegation event model; Event classes; Sources of events; Event listener interfaces; Using the delegation event model; Adapter classes; Inner classes. Swings: Swings: The origins of Swing; Two key Swing features; Components and Containers; The Swing Packages; A simple Swing Application; Create a Swing Applet; Jlabel and ImageIcon; JTextField;The Swing Buttons; JTabbedpane; JScrollPane; JList; JComboBox; JTable.

Unit 4: The Concept of JDBC (6 Hours)

The Concept of JDBC; JDBC Driver Types; JDBC Packages; A Brief Overview of the JDBC process; Database Connection; Associating the JDBC/ODBC Bridge with the Database; Statement Objects; ResultSet; Transaction Processing; Metadata, Data types; Exceptions. Servlets: Background; The Life Cycle of a Servlet; Using Tomcat for Servlet Development; A simple Servlet; The Servlet API; The Javax.servlet Package; Reading Servlet Parameter; The Javax.servlet.http package; Handling HTTP Requests and Responses; Using Cookies; Session Tracking.

Unit 5: Java Server Pages
(JSP) (6 Hours)

JSP, JSP Tags, Tomcat, Request String, Us5er Sessions, Cookies, Session Objects Java Remote Method Invocation: Remote Method Invocation concept; Server side, Client side. Enterprise java Beans; Deployment Descriptors; Session Java Bean, Entity Java Bean; Message-Driven Bean; The JAR File

Suggested Readings:

- **Schildt, H. (2007). Java: The Complete Reference (Chapters 1, 2, 3, 4, 5, 6, 8, 10, 11, 21, 22, 29, 30, 31)**

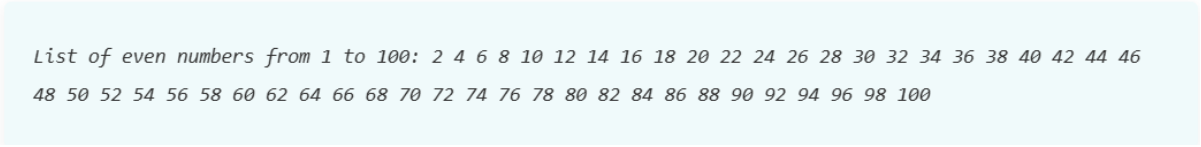
- **Keogh, J. (2007). J2EE: The Complete Reference (Chapters 5, 6, 11, 12, 15)**

Java Practice Programs

1. Java Program to Display Even Numbers From 1 to 100.

```
public class DisplayEvenNumbersExample1
{
public static void main(String args[])
{
int number=100;
System.out.print("List of even numbers from 1 to "+number+": ");
for (int i=1; i<=number; i++)
{
//logic to check if the number is even or not
//if i%2 is equal to zero, the number is even
if (i%2==0)
{
System.out.print(i + " ");
}
}
}
}
```

Output:

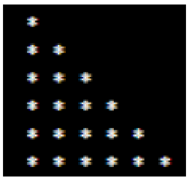


```
List of even numbers from 1 to 100: 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46
48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100
```

2. Right Triangle Star Pattern

```
public class RightTrianglePattern
{
public static void main(String args[])
{
//i for rows and j for columns
//row denotes the number of rows you want to print
int i, j, row=6;
//outer loop for rows
for(i=0; i<row; i++)
{
//inner loop for columns
for(j=0; j<=i; j++)
{
//prints stars
System.out.print("* ");
}
//throws the cursor in a new line after printing each line
System.out.println();
}
}
}
```

Output:



```
*
* *
* * *
* * * *
* * * * *
* * * * * *
```

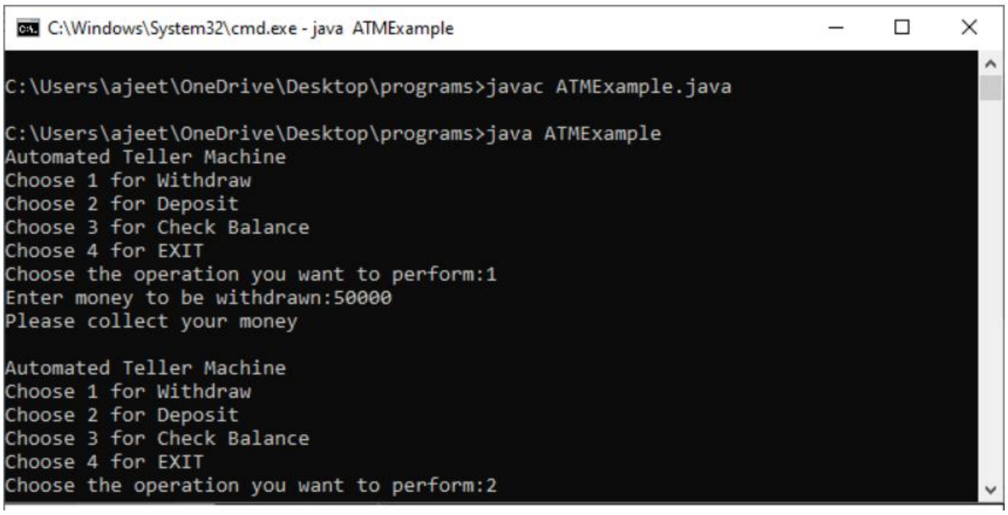
3. A Java program to create an ATM program for representing ATM transaction.

```
//import required classes and packages
import java.util.Scanner;

//create ATMExample class to implement the ATM functionality
public class ATMExample
{
    //main method starts
    public static void main(String args[] )
    {
        //declare and initialize balance, withdraw, and deposit
        int balance = 100000, withdraw, deposit;

        //create scanner class object to get choice of user
        Scanner sc = new Scanner(System.in);

        while(true)
        {
            System.out.println("Automated Teller Machine");
            System.out.println("Choose 1 for Withdraw");
            System.out.println("Choose 2 for Deposit");
            System.out.println("Choose 3 for Check Balance");
            System.out.println("Choose 4 for EXIT");
            System.out.print("Choose the operation you want to perform:");

            //get choice from user
            int choice = sc.nextInt();
            switch(choice)
            {
                case 1:
        System.out.print("Enter money to be withdrawn:");
```

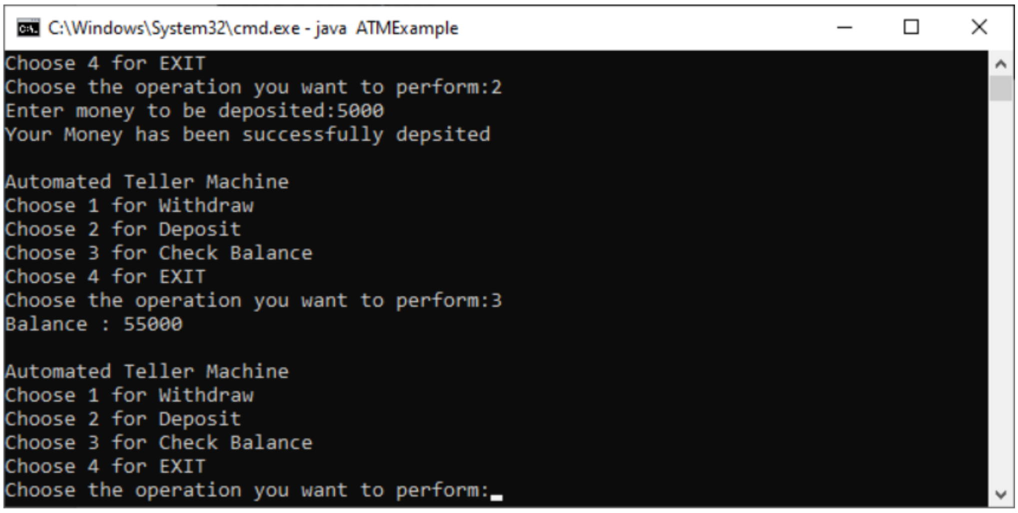
```
//get the withdrawl money from user
withdraw = sc.nextInt();

//check whether the balance is greater than or equal to the withdrawal amount
if(balance >= withdraw)
{
   //remove the withdrawl amount from the total balance
   balance = balance - withdraw;
   System.out.println("Please collect your money");
}
else
{
   //show custom error message
   System.out.println("Insufficient Balance");
}
System.out.println("");
break;

      case 2:

System.out.print("Enter money to be deposited:");

//get deposite amount from te user
deposit = sc.nextInt();

//add the deposit amount to the total balanace
balance = balance + deposit;
System.out.println("Your Money has been successfully depsited");
System.out.println("");
break;
```

```
        case 3:
    //displaying the total balance of the user
    System.out.println("Balance : "+balance);
    System.out.println("");
    break;
         case 4:
    //exit from the menu
    System.exit(0);
      }
    }
  }
}
```

Output:



```
C:\Users\ajeet\OneDrive\Desktop\programs>javac ATMExample.java

C:\Users\ajeet\OneDrive\Desktop\programs>java ATMExample
Automated Teller Machine
Choose 1 for Withdraw
Choose 2 for Deposit
Choose 3 for Check Balance
Choose 4 for EXIT
Choose the operation you want to perform:1
Enter money to be withdrawn:50000
Please collect your money

Automated Teller Machine
Choose 1 for Withdraw
Choose 2 for Deposit
Choose 3 for Check Balance
Choose 4 for EXIT
Choose the operation you want to perform:2
```

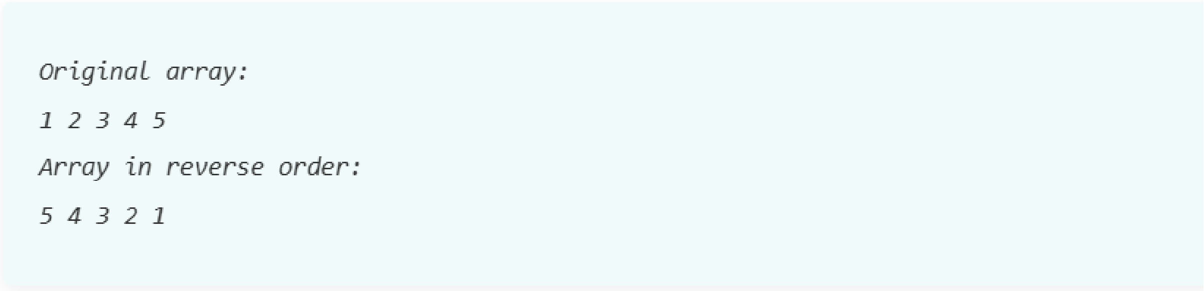


```
Choose 4 for EXIT
Choose the operation you want to perform:2
Enter money to be deposited:5000
Your Money has been successfully depsited

Automated Teller Machine
Choose 1 for Withdraw
Choose 2 for Deposit
Choose 3 for Check Balance
Choose 4 for EXIT
Choose the operation you want to perform:3
Balance : 55000

Automated Teller Machine
Choose 1 for Withdraw
Choose 2 for Deposit
Choose 3 for Check Balance
Choose 4 for EXIT
Choose the operation you want to perform:
```

4. Program to print the elements of an array in reverse order.

```
public class ReverseArray {
    public static void main(String[] args) {
        //Initialize array
        int [] arr = new int [] {1, 2, 3, 4, 5};
        System.out.println("Original array: ");
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
        System.out.println("Array in reverse order: ");
        //Loop through the array in reverse order
        for (int i = arr.length-1; i >= 0; i--) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

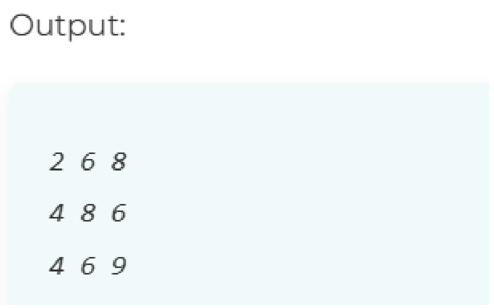
Output:



```
Original array:
1 2 3 4 5
Array in reverse order:
5 4 3 2 1
```

5. Java Program to add two matrices.

```
public class MatrixAdditionExample{
public static void main(String args[]){
//creating two matrices
int a[][]={{1,3,4},{2,4,3},{3,4,5}};
int b[][]={{1,3,4},{2,4,3},{1,2,4}};
//creating another matrix to store the sum of two matrices

int c[][]=new int[3][3];  //3 rows and 3 columns


//adding and printing addition of 2 matrices

for(int i=0;i<3;i++){

for(int j=0;j<3;j++){

c[i][j]=a[i][j]+b[i][j];    //use - for subtraction

System.out.print(c[i][j]+" ");

}

System.out.println(); //new line

}

}}
```



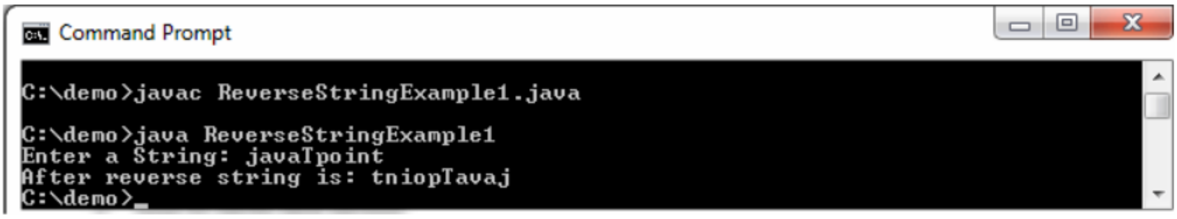
Output:

```
2 6 8
4 8 6
4 6 9
```

6. Program to reserve a string in Java without using reverse function.

```
import java.util.Scanner;

class ReverseStringExample1

{

public static void main(String args[])

{

String s;

Scanner sc=new Scanner(System.in);

System.out.print("Enter a String: ");

s=sc.nextLine();                  //reading string from user

System.out.print("After reverse string is: ");

for(int i=s.length();i>0;--i)              //i is the length of the string

{
```

```
System.out.print(s.charAt(i-1));          //printing the character at index i-1

}

}

}
```

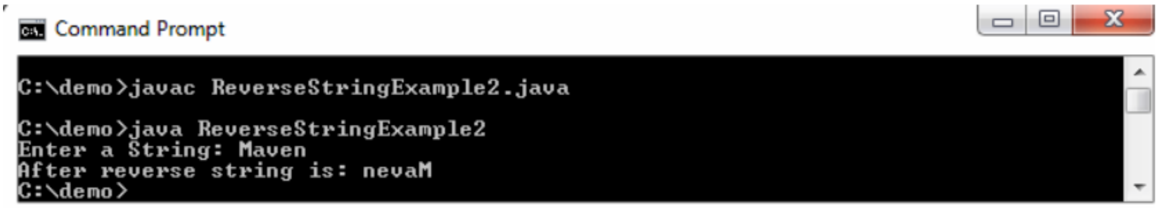


```
C:\demo>javac ReverseStringExample1.java

C:\demo>java ReverseStringExample1
Enter a String: javaTpoint
After reverse string is: tniopTavaj
C:\demo>_
```

Using while loop

```
import java.util.Scanner;

class ReverseStringExample2

{

public static void main(String args[])

{

String s;

Scanner sc=new Scanner(System.in);               //reading string from user

System.out.print("Enter a String: ");

s=sc.nextLine();

System.out.print("After reverse string is: ");

int i=s.length();                //determining the length of the string

while(i>0)

{

System.out.print(s.charAt(i-1));              //printing the character at index i-1

i--;                          //decreasing the length of the string

}

}

}
```

Output:



```
C:\demo>javac ReverseStringExample2.java

C:\demo>java ReverseStringExample2
Enter a String: Maven
After reverse string is: nevaM
C:\demo>
```