Lab Code:20EC504/1C

Programming with JAVA Lab Manual

Department of Electronics & Communication Engineering

Bapatla Engineering College :: Bapatla

(Autonomous)

G.B.C. Road, Mahatmajipuram, Bapatla-522102, Guntur (Dist.)

Andhra Pradesh, India.

E-Mail:bec.principal@becbapatla.ac.in

Web:www.becbapatla.ac.in

Contents

Bapatla Engineering College :: Bapatla
(Autonomous)

Vision

- To build centers of excellence, impart high quality education and instill high standards of ethics and professionalism through strategic efforts of our dedicated staff, which allows the college to effectively adapt to the ever changing aspects of education.

- To empower the faculty and students with the knowledge, skills and innovative thinking to facilitate discovery in numerous existing and yet to be discovered fields of engineering, technology and interdisciplinary endeavors.

Mission

- Our Mission is to impart the quality education at par with global standards to the students from all over India and in particular those from the local and rural areas.

- We continuously try to maintain high standards so as to make them technologically competent and ethically strong individuals who shall be able to improve the quality of life and economy of our country.

Bapatla Engineering College :: Bapatla

(Autonomous)

Department of Electronics and Communication Engineering

Vision

To produce globally competitive and socially responsible Electronics and Communication Engineering graduates to cater the ever changing needs of the society.

Mission

- To provide quality education in the domain of Electronics and Communication Engineering with advanced pedagogical methods.

- To provide self learning capabilities to enhance employability and entrepreneurial skills and to inculcate human values and ethics to make learners sensitive towards societal issues.

- To excel in the research and development activities related to Electronics and Communication Engineering.

Bapatla Engineering College :: Bapatla

(Autonomous)

Department of Electronics and Communication Engineering

Program Educational Objectives (PEO's)

PEO-I: Equip Graduates with a robust foundation in mathematics, science and Engineering Principles, enabling them to excel in research and higher education in Electronics and Communication Engineering and related fields.

PEO-II: Impart analytic and thinking skills in students to develop initiatives and innovative ideas for Start-ups, Industry and societal requirements.

PEO-III: Instill interpersonal skills, teamwork ability, communication skills, leadership, and a sense of social, ethical, and legal duties in order to promote lifelong learning and Professional growth of the students.

Program Outcomes (PO's)

Engineering Graduates will be able to:

PO1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7.Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9. Individual and Teamwork: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these

to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12. Life-long learning: Recognize the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Bapatla Engineering College :: Bapatla

(Autonomous)

Department of Electronics and Communication Engineering

Program Specific Outcomes (PSO's)

PSO1: Develop and implement modern Electronic Technologies using analytical methods to meet current as well as future industrial and societal needs.

PSO2: Analyze and develop VLSI, IoT and Embedded Systems for desired specifications to solve real world complex problems.

PSO3: Apply machine learning and deep learning techniques in communication and signal processing.

Programming with JAVA Lab
III B.Tech – I Semester (Code: 20EC504/1C)

Lectures	0	Tutorial	0	Practical	3	Credits	1
Continuous Internal Assessment			50	Semester End Examination (3 Hours)			50

Prerequisites: C++ Fundamentals

Course Objectives: Students will

- Basics of java and OOPS concepts
- Demonstrate the use of classes and objects in writing java programs
- Illustrate various reusable concepts to implement java programs
- Explain the Error handling mechanisms and Multithreading concepts in java

Course Outcomes: After studying this course, the students will be able to

CO1	Write and implement simple JAVA programs using Object Oriented Programming concepts
CO2	Understand the concepts of classes and objects to develop JAVA programs
CO3	Develop reusable programs using the concepts of inheritance, interfaces and packages.
CO4	Apply the concepts of Exception handling and Multithreading while implementing

Mapping of Course Outcomes with Program Outcomes & Program Specific Outcomes

	PO's												PSO's		
CO	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**	**3**
CO1	3	2			3										2
CO2	3	2	2		3										2
CO3	3	2	2		3										2
CO4	3	2			3										2
AVG	3	2	2		3										2

LIST OF LAB EXPERIMENTS

1. Java Basic Programs
2. Java Array Programs
3. Implement the concept of Scope and lifetime of a variable
4. Implement the concept of polymorphism
5. Implement the concept of objects and classes
6. Illustrate the different types of Constructors
7. Write a Java program using static, this and final keyword
8. Implement the concept of Method overloading
9. Implement reusability concept using inheritance
10. Develop Java programs using Abstract Class
11. Implement multiple inheritance using interface
12. Write a Java program to demonstrate packages
13. Implement User defined Exceptions in java
14. Implement Built-in Exceptions in java
15. Develop Java programs using Multithreading

1. Java Basic Programs

Aim:

Write a java program to print Factorial of five numbers and ten values of Fibonacci series.

Software required: JDK

a) Program: Factorial of five numbers

```
class FactorialExample
{
   public static void main(String args[])
{
        int i,fact=1;
        int number=5;//It is the number to calculate factorial
        for(i=1;i<=number;i++)
        {
                fact=fact*i;
        }
  System.out.println("Factorial of "+number+" is: "+fact);
 }
}
```

Output:

Factorial of 5 is: 120

b) Program: Fibonacci series

```
class FibonacciExample1
{
public static void main(String args[])
{
        int n1=0,n2=1,n3,i,count=10;
        System.out.print(n1+" "+n2);//printing 0 and 1
        for(i=2;i<count;++i)
        {
                n3=n1+n2;
                System.out.print(" "+n3);
                n1=n2;
                n2=n3;
        }

}}
```

Output:

0 1 1 2 3 5 8 13 21 34

2. Java Array Programs

Aim:

Write a java program to print elements of an array, Addition and Multiplication of two matrices.

Software Required: JDK

a) Program: Elements of an Array

```
public class PrintArray
{
   public static void main(String[] args)
{
        //Initialize array
        int [] arr = new int [] {1, 2, 3, 4, 5};
        System.out.println("Elements of given array: ");
        //Loop through the array by incrementing value of i
        for (int i = 0; i < arr.length; i++)
         {
            System.out.print(arr[i] + " ");
        }
    }
}
```

Output:

```
Elements of given array:
1 2 3 4 5
```

b) Program: Addition of 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];
System.out.print(c[i][j]+" ");
}
System.out.println();//new line
}
}
}
```

Output:

2 6 8
4 8 6
4 6 9

C) Program:Multiplication of two Matrices

```
public class MatrixMultiplicationExample
{
public static void main(String args[])
{
//creating two matrices
int a[][]={{1,1,1},{2,2,2},{3,3,3}};
int b[][]={{1,1,1},{2,2,2},{3,3,3}};

//creating another matrix to store the multiplication of two matrices
int c[][]=new int[3][3];  //3 rows and 3 columns

//multiplying and printing multiplication of 2 matrices
for(int i=0;i<3;i++)
{
        for(int j=0;j<3;j++)
        {
                c[i][j]=0;
        for(int k=0;k<3;k++)
        {
                c[i][j]+=a[i][k]*b[k][j];
        }//end of k loop
                System.out.print(c[i][j]+" ");  //printing matrix element
        }//end of j loop
                System.out.println();//new line
}
}}
```

Output:

6 6 6

12 12 12

18 18 18

3. Implement the concept of Scope and lifetime of a variable

Aim:

Write a java program to show the scope of a variable.

Software Required: JDK

Program:

```
public class Scope
    {
        public static void main(String[] args)
    {
      int n1=10,n2;
      if(n1==10)
      {
              n2=20;
              System.out.println("n1 and n2 : "+n1 +" " +n2 );
      }
    System.out.println("n1:"+n1);
    }
    }
```

Output:

n1 and n2 : 10 20

n1: 10

4. Implement the concept of polymorphism

Aim:

Write the java program to implement the runtime polymorphism

Software Required: JDK

Program:

```
class Bike
{
        void run()
        {
        System.out.println("running");
        }
}
class Splendor extends Bike
{
        void run()
        {
        System.out.println("running safely with 60km");
        }

  public static void main(String args[])
{
  Bike b = new Splendor();//upcasting
    b.run();
  }
}
```

Output:

running safely with 60km

5. Implement the concept of objects and classes

Aim: Write a java program to implement the classes and objects

Software Required: JDK

Program:

```
class Student
{
        int rollno;
        String name;
 void insertRecord(int r, String n)
{
       rollno=r;
       name=n;
 }
 void displayInformation()
{
System.out.println(rollno+" "+name);
}
}
class TestStudent
{
 public static void main(String args[])
{
         Student s1=new Student();
       Student s2=new Student();
       s1.insertRecord(111,"Karan");
       s2.insertRecord(222,"Aryan");
       s1.displayInformation();
       s2.displayInformation();
 }
}
```

Output:

111 Karan

222 Aryan

6. Illustrate the different types of Constructors

Aim: Write a java program to implement default and parameterized constructor.

Software Required: JDK

Program: Default Constructor

```
class Bike1
{
//creating a default constructor
        Bike1()
        {
        System.out.println("Bike is created");
        }
//main method
public static void main(String args[])
{
//calling a default constructor
        Bike1 b=new Bike1();
}
}
```

Output:

Bike is created

Program: Parameterized Constructor

```
class Student
{
         int id;
         String name;
   //creating a parameterized constructor
   Student(int i,String n)
{
   id = i;
   name = n;
   }
   //method to display the values
   void display()
{
System.out.println(id+" "+name);
}
  public static void main(String args[])
{
   //creating objects and passing values
   Student4 s1 = new Student4(111,"Karan");
   Student4 s2 = new Student4(222,"Aryan");
```

```
    //calling method to display the values of object
    s1.display();
    s2.display();
   }
}
```

Output:

```
111  Karan
222  Aryan
```

7. Write a Java program using static, this and final keyword

Aim: Write a java program to demonstrate the use of static, this and final keywords

Software Required: JDK

Program: Use of Static Keyword

```
class Student
{
   int rollno;//instance variable
   String name;
   static String college ="BEC";//static variable
   //constructor
   Student(int r, String n)
{
   rollno = r;
   name = n;
   }
   //method to display the values
   void display ()
{
System.out.println(rollno+" "+name+" "+college);}
}
//Test class to show the values of objects
public class TestStaticVariable1
{
 public static void main(String args[])
{
 Student s1 = new Student(111,"Karan");
 Student s2 = new Student(222,"Aryan");
 //we can change the college of all objects by the single line of code
 //Student.college="BBDIT";
 s1.display();
 s2.display();
 }
}
```

Output:

```
111  Karan  BEC
222  Aryan  BEC
```

Program: Use of this Keyword

```
class Student
{
        int rollno;
        String name;
        float fee;
        Student(int rollno,String name,float fee)
        {
                this.rollno=rollno;
                this.name=name;
                this.fee=fee;
        }
void display()
{
System.out.println(rollno+" "+name+" "+fee);}
}

class TestThis2
{
public static void main(String args[])
{
Student s1=new Student(111,"ankit",5000f);
Student s2=new Student(112,"sumit",6000f);
s1.display();
s2.display();
}
}
```

Output:

```
111  ankit  5000.0
112  sumit  6000.0
```

Program: Use of Final Keyword

```
class Bike
{
        final void run()
        {
        System.out.println("running");
        }
}

class Honda extends Bike
{
        void run()
        {
        System.out.println("running safely with 100kmph");
        }
```

```
  public static void main(String args[])
{
  Honda honda= new Honda();
  honda.run();
  }
}
```

Output:

Compile Time Error

8. Implement the concept of Method overloading

Aim: To Write a java program for implementing the Method overloading

Software Required: JDK

Program:

```
class Adder
{
static int add(int a,int b)
{
        return a+b;
        }
static int add(int a,int b,int c)
{
return a+b+c;
}
}
class TestOverloading1
{
public static void main(String[] args)
{
        System.out.println(Adder.add(11,11));
        System.out.println(Adder.add(11,11,11));
}
}
```

Output:

```
22
33
```

9. Implement reusability concept using inheritance

Aim: Write a java program to implement Single, Multilevel and Hierarchical inheritance

Software Required: JDK

Program: Single level inheritance

```
class Animal
{
        void eat()
        {
        System.out.println("eating...");
        }
}
class Dog extends Animal
{
        void bark()
        {
        System.out.println("barking...");
        }
}
class TestInheritance
{
public static void main(String args[])
{
        Dog d=new Dog();
        d.bark();
        d.eat();
}
}
```

Output:

```
barking..
eating..
```

Program: Multilevel inheritance

```
class Animal
{
        void eat()
        {
        System.out.println("eating...");
        }
}
```

```
class Dog extends Animal
{
        void bark()
        {
        System.out.println("barking...");
        }
}
class BabyDog extends Dog
{
        void weep()
        {
        System.out.println("weeping...");
        }
}
class TestInheritance2
{
public static void main(String args[]){
        BabyDog d=new BabyDog();
d.weep();
d.bark();
d.eat();
}
}
```

Output:

```
        weeping..
         barking..
        eating..
```

Program: Hierarchical inheritance

```
class Animal{
void eat(){System.out.println("eating...");}
}
class Dog extends Animal{
void bark(){System.out.println("barking...");}
}
class Cat extends Animal{
void meow(){System.out.println("meowing...");}
}
class TestInheritance3{
public static void main(String args[]){
Cat c=new Cat();
c.meow();
c.eat();
}}
```

Output:

```
meowing..
eating...
```

10. Develop Java programs using Abstract Class

Aim: To write a java program for implementing the Abstract class

Software Required: JDK

Program:

```
abstract class Shape
{
        abstract void draw();
}
//In real scenario, implementation is provided by others i.e. unknown by
end user
class Rectangle extends Shape
{
        void draw()
        {
        System.out.println("drawing rectangle");
        }
}
class Circle1 extends Shape
{
        void draw()
        {
        System.out.println("drawing circle");
        }
}
//In real scenario, method is called by programmer or user
class TestAbstraction1
{
public static void main(String args[])
{
Shape s=new Circle1();
s.draw();
}
}
```

Output:

```
drawing circle
```

11. Implement multiple inheritance using interface

Aim: Write a java program to implement multiple inheritance

Software Required: JDK

Program:

```
interface Printable
{
        void print();
}
interface Showable
{
        void show();
}
class A7 implements Printable,Showable
{
        public void print()
        {
        System.out.println("Hello");
        }
        public void show()
        {
        System.out.println("Welcome");
        }
public static void main(String args[])
{
A7 obj = new A7();
obj.print();
obj.show();
 }
}
```

Output:

```
Hello
Welcome
```

12. Write a Java program to demonstrate packages

Aim: Write a java program to demonstrate the packages

Software Required: JDK

Program:

```
//save by A.java
package pack;
public class A
{
  public void msg()
{
System.out.println("Hello");
}
}
//save by B.java
package mypack;
import pack.*;
 class B
{
  public static void main(String args[])
{
          A obj = new A();
        obj.msg();
  }
}
```

Output:

Hello

13. Implement User defined Exceptions in java

Aim: Write a java program to implement user defined exceptions

Software Required: JDK

Program:

```
// class representing custom exception
class InvalidAgeException  extends Exception
{
    public InvalidAgeException (String str)
    {
        // calling the constructor of parent Exception
        super(str);
    }
}

// class that uses custom exception InvalidAgeException
public class TestCustomException1
{

    // method to check the age
    static void validate (int age) throws InvalidAgeException
        {
                if(age < 18)
                {

                // throw an object of user defined exception
                 throw new InvalidAgeException("age is not valid to vote");
                }
                 else
                {
                        System.out.println("welcome to vote");
                }
        }

    // main method
    public static void main(String args[])
    {
        try
        {
            // calling the method
            validate(13);
        }
        catch (InvalidAgeException ex)
        {
            System.out.println("Caught the exception");
```

```
            // printing the message from InvalidAgeException object
            System.out.println("Exception occured: " + ex);
        }

        System.out.println("rest of the code...");
    }
}
```

Output:

```
Caught the exception
Exception occured: InvalidAgeException: age is not valid to vote
rest of the code
```

14. Implement Built-in Exceptions in java

Aim: Write a java program to demonstrate Arithmetic Exception

Software Required: JDK

Program:

```
public class ExcepTest
{
  public static void main(String args[])
 {
     try
      {
       int b = 0;
       int c = 1/b;
       System.out.println("c :" + c);
     }
     catch (ArithmeticException e)
      {
       System.out.println("Exception thrown  :" + e);
      }
     System.out.println("Out of the block");
  }
}
```

Output:

```
Exception thrown  :java.lang.ArithmeticException: / by zero
Out of the block
```

15. Develop Java programs using Multithreading

Aim: Write a java program for thread creation by extending the thread class

Software Required: JDK

Program:

```
class MultithreadingDemo extends Thread {
    public void run()
    {
        try {
            // Displaying the thread that is running
            System.out.println(
                "Thread " + Thread.currentThread().getId()
                + " is running");
        }
        catch (Exception e) {
            // Throwing an exception
            System.out.println("Exception is caught");
        }
    }
}

// Main Class
public class Multithread {
    public static void main(String[] args)
    {
        int n = 8; // Number of threads
        for (int i = 0; i < n; i++) {
            MultithreadingDemo object
                = new MultithreadingDemo();
            object.start();
```

```
        }
    }
}
```

Output:

```
Thread 9 is running
Thread 14 is running
Thread 13 is running
Thread 12 is running
Thread 11 is running
Thread 10 is running
Thread 15 is running
Thread  16 is running
```

REFERENCES

1. The Complete Reference Java J2SE 7th Edition by Herbert Schildt, McGraw-Hill Companies
2. Big Java 2nd Edition, Cay Horstmann, John Wiley and Sons.
3. www.javatpoint.com
4. www.geeksforgeeks.org
5. www.tutorialspoint.com