Free Software Movement Karnataka
www.fsmk.org

Basics of Java Programming Lab (BPLCK105C/205C)

I/II Semester
(common to all branches)
LAB MANUAL

Prabodh C P
Asst Professor
Dept of CSE, SIT
Volunteer, FSMK
Visit : **https://tinyurl.com/yc5h2mu9**

The versioned repository of all the programs can be found here as a GitLab Repository
https://gitlab.com/lab_manuals/java_manual_vtu_2022_23

Contents

Listings

Preface

Usage of Free and Open Source Software

This manual has been prepared entirely using Free Software. The following Free Software has been used in preparation of this manual.

Operating System Ubuntu 22.04.1 LTS (Jammy Jellyfish)

Linux Kernel 5.15.0-56-generic

Java Compiler javac 19.0.1

Version Control git 2.34.1

Typesetting Texmaker 5.0.3 with LaTeX

Image Editing GIMP 2.10.30

 A repository of all the programs can be found as GitLab repository in the following link https://gitlab.com/lab_manuals/java_manual_vtu_2022_23. Looking for your feedback. If you want to contribute let me know by sending a PR on the git repo mentioned earlier.

Use and spread the word of Free Software. Free Software leads to a Free Society!

Prabodh C P

Chapter 1

Quadratic Equation

Question

Write a JAVA program that prints all real solutions to the quadratic equation ax2+bx+c=0. Read in a, b, c and use the quadratic formula.

Java Code

```
import java.util.Scanner;

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

        System.out.print("Enter a: ");
        double a = input.nextDouble();

        System.out.print("Enter b: ");
        double b = input.nextDouble();

        System.out.print("Enter c: ");
        double c = input.nextDouble();

        double discriminant = b * b - 4 * a * c;
        if (discriminant > 0) {
            System.out.println("The equation has real and distinct roots.");
            double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
            double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);
            System.out.println("The roots are " + root1 + " and " + root2);
        } else if (discriminant == 0) {
            System.out.println("The equation has real and equal roots.");
            double root = -b / (2 * a);
            System.out.println("The root is " + root);
        } else {
            System.out.println("The equation has no real roots.");
            double realp = -b / (2 * a);
            double imagp = (Math.sqrt(-discriminant)) / (2 * a);
            System.out.printf("The roots are (%.2f + i%.4f) and (%.2f - i%.4f)\n",
    realp, imagp, realp, imagp);
        }
    }
}
```

Listing 1.1: QuadraticEquationDemo.java

Output

==========

```
putta:~/.../Program1$ javac QuadraticEquationDemo.java
putta:~/.../Program1$ java QuadraticEquationDemo
Enter a: 1
Enter b: 4
Enter c: 4
The equation has real and equal roots.
The root is -2.0

putta:~/.../Program1$ java QuadraticEquationDemo
Enter a: 1
Enter b: -5
Enter c: 6
The equation has real and distinct roots.
The roots are 3.0 and 2.0

putta:~/.../Program1$ java QuadraticEquationDemo
Enter a: 1
Enter b: 3
Enter c: 3
The equation has no real roots.
The roots are (-1.50 + i0.8660) and (-1.50 - i0.8660)
```

Listing 1.2: OUTPUT

Chapter 2

Array Multiplication

Question

Write a JAVA program for multiplication of two arrays.

Java Code

```
import java.util.Scanner;
import java.util.Arrays;

public class MultiplyArraysDemo {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the number of elements in the first array: ");
        int n = input.nextInt();
        int[] array1 = new int[n];
        for (int i = 0; i < n; i++) {
            System.out.print("Enter element " + (i+1) + " of the first array: ");
            array1[i] = input.nextInt();
        }

        System.out.print("Enter the number of elements in the second array: ");
        int m = input.nextInt();
        int[] array2 = new int[m];
        for (int i = 0; i < m; i++) {
            System.out.print("Enter element " + (i+1) + " of the second array: ");
            array2[i] = input.nextInt();
        }

        if (n != m) {
            System.out.println("Error: Arrays have different length, not possible to multiply them.");
            return;
        }

        int[] result = new int[n];
        for (int i = 0; i < n; i++) {
            result[i] = array1[i] * array2[i];
        }

        System.out.println("Result of multiplying arrays: " + Arrays.toString(result));
    }
}
```

Listing 2.1: MultiplyArraysDemo.java

Output

==========

```
putta:~/.../Program2$ javac MultiplyArraysDemo.java
putta:~/.../Program2$ java MultiplyArraysDemo
Enter the number of elements in the first array: 4
Enter element 1 of the first array: 1
Enter element 2 of the first array: 2
Enter element 3 of the first array: 3
Enter element 4 of the first array: 4
Enter the number of elements in the second array: 4
Enter element 1 of the second array: 4
Enter element 2 of the second array: 3
Enter element 3 of the second array: 2
Enter element 4 of the second array: 1
Result of multiplying arrays: [4, 6, 6, 4]

putta:~/.../Program2$ java MultiplyArraysDemo
Enter the number of elements in the first array: 3
Enter element 1 of the first array: 1
Enter element 2 of the first array: 2
Enter element 3 of the first array: 3
Enter the number of elements in the second array: 2
Enter element 1 of the second array: 4
Enter element 2 of the second array: 5
Error: Arrays have different length, not possible to multiply them.
```

Listing 2.2: OUTPUT

Chapter 3

Shift Operation

Question

Demonstrate the following operations and sign extension with Java programs

i) ≪ ii) ≫ iii) ⋙

Java Code

```
import java.util.Scanner;

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

        System.out.print("\nLeft Shift Operator\nEnter a number : ");
        int num = sc.nextInt();
        System.out.print("Enter the number of positions for the Left Shift Operation : ");
        int pos = sc.nextInt();

        int res = num << pos;
        System.out.println("\nResult of Left Shift Operation");
        System.out.printf("\n%d << %d = %d\n", num, pos, res);

        System.out.print("\nRight Shift Operator\nEnter a number : ");
        num = sc.nextInt();
        System.out.print("Enter the number of positions for the Right Shift Operation : ");
        pos = sc.nextInt();

        res = num >> pos;
        System.out.println("\nResult of Right Shift Operation");
        System.out.printf("\n%d >> %d = %d\n", num, pos, res);
        if(num < 0){
            System.out.println("We observe that Even after the Right Shift operation the sign bit of negative number is preserved. We call this sign extension.");
        }
        System.out.print("\nUnsigned Right Shift Operator\nEnter a number : ");
        num = sc.nextInt();
        System.out.print("Enter the number of positions for the Unsigned Right Shift Operation : ");
        pos = sc.nextInt();

        res = num >>> pos;
        System.out.println("\nResult of Unsigned Right Shift Operation");
        System.out.printf("\n%d >>> %d = %d\n", num, pos, res);
    }
```

```
}
```

Listing 3.1: OperatorDemoSignExtension.java

Output

==========

```
putta:~/.../Program10$ javac OperatorDemoSignExtension.java
putta:~/.../Program10$ java OperatorDemoSignExtension

Left Shift Operator
Enter a number : 4
Enter the number of positions for the Left Shift Operation : 3

Result of Left Shift Operation

4 << 3 = 32

Right Shift Operator
Enter a number : -65
Enter the number of positions for the Right Shift Operation : 2

Result of Right Shift Operation

-65 >> 2 = -17
We observe that Even after the Right Shift operation the sign bit of negative number
    is preserved. We call this sign extension

Unsigned Right Shift Operator
Enter a number : -65
Enter the number of positions for the Unsigned Right Shift Operation : 2

Result of Unsigned Right Shift Operation

-65 >>> 2 = 1073741807

putta:~/.../Program10$ java OperatorDemoSignExtension

Left Shift Operator
Enter a number : 4
Enter the number of positions for the Left Shift Operation : 2

Result of Left Shift Operation

4 << 2 = 16

Right Shift Operator
Enter a number : 57
Enter the number of positions for the Right Shift Operation : 2

Result of Right Shift Operation

57 >> 2 = 14

Unsigned Right Shift Operator
Enter a number : 57
Enter the number of positions for the Unsigned Right Shift Operation : 2

Result of Unsigned Right Shift Operation

57 >>> 2 = 14
```

Listing 3.2: OUTPUT

Chapter 4

Sorting

Question

Write a JAVA program to sort list of elements in ascending and descending order.

Java Code

```
import java.util.Scanner;

public class ExchangeSortDemo {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number of elements in the array: ");
        int n = sc.nextInt();
        int[] arr = new int[n];

        System.out.println("Enter the elements of the array: ");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

        for (int i = 0; i < n - 1; i++) {
            for (int j = i + 1; j < n; j++) {
                if (arr[i] > arr[j]) {
                    int temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                }
            }
        }

        System.out.println("\nSorted array in ascending order: ");
        for (int i = 0; i < n; i++) {
            System.out.print(arr[i] + " ");
        }

        for (int i = 0; i < n - 1; i++) {
            for (int j = i + 1; j < n; j++) {
                if (arr[i] < arr[j]) {
                    int temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                }
            }
        }

        System.out.println("\nSorted array in descending order: ");
```

```
41          for (int i = 0; i < n; i++) {
42              System.out.print(arr[i] + " ");
43          }
44
45      }
46  }
```

Listing 4.1: ExchangeSortDemo.java

Output

==========

```
1  putta:~/.../Program4$ javac ExchangeSortDemo.java
2  putta:~/.../Program4$ java ExchangeSortDemo
3  Enter the number of elements in the array:
4  6
5  Enter the elements of the array:
6  6 1 5 2 4 3
7
8  Sorted array in ascending order:
9  1 2 3 4 5 6
10 Sorted array in descending order:
11 6 5 4 3 2 1
12
13 putta:~/.../Program4$ java ExchangeSortDemo
14 Enter the number of elements in the array:
15 5
16 Enter the elements of the array:
17 9 1 8 2 5
18
19 Sorted array in ascending order:
20 1 2 5 8 9
21 Sorted array in descending order:
22 9 8 5 2 1
```

Listing 4.2: OUTPUT

Chapter 5

Student Details

Question

Create a JAVA class called Student with the following details as variables within it.
USN
NAME
BRANCH
PHONE
PERCENTAGE
Write a JAVA program to create n Student objects and print the USN, Name, Branch, Phone, and percentage of these objects with suitable headings.

Java Code

```
import java.util.Scanner;
import java.util.List;
import java.util.ArrayList;

class StudentType {
    private String USN;
    private String NAME;
    private String BRANCH;
    private String PHONE;
    private double PERCENTAGE;

    public StudentType(String USN, String NAME, String BRANCH, String PHONE, double
    PERCENTAGE) {
        this.USN = USN;
        this.NAME = NAME;
        this.BRANCH = BRANCH;
        this.PHONE = PHONE;
        this.PERCENTAGE = PERCENTAGE;
    }

    public String getUSN() {
        return USN;
    }

    public String getNAME() {
        return NAME;
    }

    public String getBRANCH() {
        return BRANCH;
    }

    public String getPHONE() {
```

```
        return PHONE;
    }

    public double getPERCENTAGE() {
        return PERCENTAGE;
    }
}


public class StudentDemo {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        List<StudentType> students = new ArrayList<>();

        System.out.println("Enter the number of students:");
        int n = sc.nextInt();

        for (int i = 1; i <= n; i++) {
            System.out.println("Enter the details of student " + i + ":");
            System.out.print("USN: ");
            String USN = sc.next();
            System.out.print("NAME: ");
            String NAME = sc.next();
            System.out.print("BRANCH: ");
            String BRANCH = sc.next();
            System.out.print("PHONE: ");
            String PHONE = sc.next();
            System.out.print("PERCENTAGE: ");
            double PERCENTAGE = sc.nextDouble();

            students.add(new StudentType(USN, NAME, BRANCH, PHONE, PERCENTAGE));
        }
        System.out.println("\nSTUDENT DETAILS\n====================");
        System.out.println("USN" + "\t\t" + "NAME" + "\t" + "BRANCH" + "\t" + "PHONE
    " + "\t\t" + "PERCENTAGE");
        for (StudentType student : students) {
            System.out.println(student.getUSN() + "\t" + student.getNAME() + "\t" +
    student.getBRANCH() + "\t" + student.getPHONE() + "\t" + student.getPERCENTAGE())
    ;
        }
    }
}
```

Listing 5.1: StudentDemo.java

Output

==========

```
putta:~/.../Program5$ javac StudentDemo.java
putta:~/.../Program5$ java StudentDemo
Enter the number of students:
2
Enter the details of student 1:
USN: 1SI22CS036
NAME: Rajesh
BRANCH: CSE
PHONE: 7187238165
PERCENTAGE: 89.7

Enter the details of student 2:
USN: 1PE22EC088
NAME: Sheela
BRANCH: ECE
```

```
PHONE: 8238887233
PERCENTAGE: 84.5

STUDENT DETAILS
====================
USN      NAME    BRANCH  PHONE        PERCENTAGE
1SI22CS036  Rajesh  CSE 7187238165  89.7
1PE22EC088  Sheela  ECE 8238887233  84.5
```

Listing 5.2: OUTPUT

Chapter 6

Polymorphism - Method Overloading

Question

Write a JAVA program demonstrating Method overloading and Constructor overloading.

Java Code

```
class PointType {
    private double x;
    private double y;

    // Constructor overloading
    public PointType() {
        this.x = 0;
        this.y = 0;
    }

    public PointType(double x, double y) {
        this.x = x;
        this.y = y;
    }

    // Method overloading to calculate distance between two PointType objects
    public double distance(PointType point) {
        System.out.println("\nCalculating distance between two PointType objects");
        return Math.sqrt(Math.pow(x - point.x, 2) + Math.pow(y - point.y, 2));
    }

    // Method overloading to calculate distance between a PointType object and a
   point specified by its coordinates
    public double distance(double x, double y) {
        System.out.println("Calculating distance between a PointType object and a
   point specified by its coordinates");
        return Math.sqrt(Math.pow(this.x - x, 2) + Math.pow(this.y - y, 2));
    }

    public void show(){
        System.out.printf("(%.1g,%.1g)\n",x,y);
    }
}

class PointDemo{
    public static void main(String[] args) {
        PointType point1 = new PointType(1, 1);
        PointType point2 = new PointType(7, 9);
        System.out.print("\npoint1 coordinates :"); point1.show();
        System.out.print("point2 coordinates :"); point2.show();
```

```
39          double dVal = point1.distance(point2);
40          System.out.println("Distance between point1 and point2: " + dVal );
41
42          PointType point3 = new PointType();
43          System.out.print("\npoint3 coordinates :"); point3.show();
44          dVal = point3.distance(3,4);
45          System.out.println("Distance between point3 and (3, 4): " + dVal);
46      }
47  }
```

Listing 6.1: PointDemo.java

Output

==========

```
1  putta:~/.../Program6$ javac PointDemo.java
2  putta:~/.../Program6$ java PointDemo
3
4  point1 coordinates :(1,1)
5  point2 coordinates :(7,9)
6
7  Calculating distance between two PointType objects
8  Distance between point1 and point2: 10.0
9
10 point3 coordinates :(0,0)
11 Calculating distance between a PointType object and a point specified by its
       coordinates
12 Distance between point3 and (3, 4): 5.0
```

Listing 6.2: OUTPUT

Chapter 7

Inheritance

Question

Design a super class called Staff with details as StaffId, Name, Phone, Salary. Extend this class by writing three subclasses namely Teaching (domain, publications), Technical (skills), and Contract (period). Write a JAVA program to read and display at least 3 staff objects of all three categories.

Java Code

```
class Staff {
    private int staffId;
    private String name;
    private String phone;
    private double salary;

    public Staff(int staffId, String name, String phone, double salary) {
        this.staffId = staffId;
        this.name = name;
        this.phone = phone;
        this.salary = salary;
    }

    public int getStaffId() {
        return staffId;
    }

    public String getName() {
        return name;
    }

    public String getPhone() {
        return phone;
    }

    public double getSalary() {
        return salary;
    }

    public void DisplayInfo(){
        System.out.println("Name: " + this.getName());
        System.out.println("Staff ID: " + this.getStaffId());
        System.out.println("Phone: " + this.getPhone());
        System.out.println("Salary: " + this.getSalary());
    }
}

```

```
38 class Teaching extends Staff {
39     private String domain;
40     private int publications;
41 
42     public Teaching(int staffId, String name, String phone, double salary, String
       domain, int publications) {
43         super(staffId, name, phone, salary);
44         this.domain = domain;
45         this.publications = publications;
46     }
47 
48     public String getDomain() {
49         return domain;
50     }
51 
52     public int getPublications() {
53         return publications;
54     }
55 
56     public void DisplayInfo(){
57         super.DisplayInfo();
58         System.out.println("Domain: " + this.getDomain());
59         System.out.println("Publications: " + this.getPublications());
60     }
61 }
62 
63 class Technical extends Staff {
64     private String skills;
65 
66     public Technical(int staffId, String name, String phone, double salary, String
       skills) {
67         super(staffId, name, phone, salary);
68         this.skills = skills;
69     }
70 
71     public String getSkills() {
72         return skills;
73     }
74     public void DisplayInfo(){
75         super.DisplayInfo();
76         System.out.println("Skills: " + this.getSkills());
77      }
78 }
79 
80 class Contract extends Staff {
81     private int period;
82 
83     public Contract(int staffId, String name, String phone, double salary, int
       period) {
84         super(staffId, name, phone, salary);
85         this.period = period;
86     }
87 
88     public int getPeriod() {
89         return period;
90     }
91     public void DisplayInfo(){
92         super.DisplayInfo();
93         System.out.println("Period: " + this.getPeriod()+" months");
94     }
95 }
96 
```

```
97
98
99  public class StaffTypeDemo {
100     public static void main(String[] args) {
101         Teaching t1 = new Teaching(1, "Rajesh Nayak", "9822546534", 75000, "Computer
        Science", 15);
102         Teaching t2 = new Teaching(2, "Sita Devi", "8787432499", 80000, "Mathematics
        ", 20);
103         Teaching t3 = new Teaching(3, "John Peter", "8528734373", 85000, "Physics",
        25);
104         Technical te1 = new Technical(4,"Ramesha","9473673642",90000,"Java,Python,C
        ++");
105         Technical te2 = new Technical(5,"Suresha","8917612332",95000,"JavaScript,
        React,Node.js");
106         Technical te3 = new Technical(6,"Dinesha","9944222323",100000,"Python,
        TensorFlow,Keras");
107         Contract c1 = new Contract(7, "Abida Begum", "9323786211", 75000, 6);
108         Contract c2 = new Contract(8, "Lily Thomas", "8776551219", 80000, 12);
109         Contract c3 = new Contract(9, "Seema Jain", "9922324343", 85000, 18);
110         //display the staff objects
111         System.out.println("Teaching Staff 1:");
112         t1.DisplayInfo();
113         System.out.println("\nTeaching Staff 2:");
114         t2.DisplayInfo();
115         System.out.println("\nTeaching Staff 3:");
116         t3.DisplayInfo();
117
118         System.out.println("\nTechnical Staff 1:");
119         te1.DisplayInfo();
120
121         System.out.println("\nTechnical Staff 2:");
122         te2.DisplayInfo();
123
124         System.out.println("\nTechnical Staff 3:");
125         te3.DisplayInfo();
126
127         System.out.println("\nContract Staff 1:");
128         c1.DisplayInfo();
129
130         System.out.println("\nContract Staff 2:");
131         c2.DisplayInfo();
132
133         System.out.println("\nContract Staff 3:");
134         c3.DisplayInfo();
135     }
136 }
```

Listing 7.1: StaffTypeDemo.java

Output

```
==================================
1  putta:~/.../Program7$ javac StaffTypeDemo.java
2  putta:~/.../Program7$ java StaffTypeDemo
3  Teaching Staff 1:
4  Name: Rajesh Nayak
5  Staff ID: 1
6  Phone: 9822546534
7  Salary: 75000.0
8  Domain: Computer Science
9  Publications: 15
10
11 Teaching Staff 2:
```

```
Name: Sita Devi
Staff ID: 2
Phone: 8787432499
Salary: 80000.0
Domain: Mathematics
Publications: 20

Teaching Staff 3:
Name: John Peter
Staff ID: 3
Phone: 8528734373
Salary: 85000.0
Domain: Physics
Publications: 25

Technical Staff 1:
Name: Ramesha
Staff ID: 4
Phone: 9473673642
Salary: 90000.0
Skills: Java,Python,C++

Technical Staff 2:
Name: Suresha
Staff ID: 5
Phone: 8917612332
Salary: 95000.0
Skills: JavaScript,React,Node.js

Technical Staff 3:
Name: Dinesha
Staff ID: 6
Phone: 9944222323
Salary: 100000.0
Skills: Python,TensorFlow,Keras

Contract Staff 1:
Name: Abida Begum
Staff ID: 7
Phone: 9323786211
Salary: 75000.0
Period: 6 months

Contract Staff 2:
Name: Lily Thomas
Staff ID: 8
Phone: 8776551219
Salary: 80000.0
Period: 12 months

Contract Staff 3:
Name: Seema Jain
Staff ID: 9
Phone: 9922324343
Salary: 85000.0
Period: 18 months
```

Listing 7.2: OUTPUT

Chapter 8

Dynamic Dispatch

Question

Demonstrate dynamic dispatch using abstract class in JAVA.

Java Code

```
abstract class Shape {
    abstract void draw();
}

class Circle extends Shape {
    void draw() {
        System.out.println("Drawing Circle");
    }
}

class Square extends Shape {
    void draw() {
        System.out.println("Drawing Square");
    }
}

class DynamicDispatchDemo {
    public static void main(String[] args) {
        Shape s;
        s = new Circle();
        s.draw();
        s = new Square();
        s.draw();
    }
}
```

Listing 8.1: DynamicDispatchDemo.java

Output

==========

```
putta:~/.../Program8$ javac DynamicDispatchDemo.java
putta:~/.../Program8$ java DynamicDispatchDemo
Drawing Circle
Drawing Square
```

Listing 8.2: OUTPUT

Chapter 9

Packages and Acess modifiers

Question

Create two packages P1 and P2. In package P1, create class A, class B inherited from A, class C . In package P2, create class D inherited from class A in package P1 and class E. Demonstrate working of access modifiers (private, public, protected, default) in all these classes using JAVA.

Java Code

```java
// Package P1
package com.P1;

// Class A with private and protected variables and public methods
public class A {
    private int x;
    protected int y;

    public void setX(int x) {
        this.x = x;
    }

    public int getX() {
        return x;
    }

    public void setY(int y) {
        this.y = y;
    }

    public int getY() {
        return y;
    }
}
```

Listing 9.1: A.java

```java
// Package P1
package com.P1;


// Class B inherited from A with default variable and public method
public class B extends A {
    int z;

    public void setZ(int z) {
        this.z = z;
    }
```

```
12
13      public int getZ() {
14          return z;
15      }
16  }
```

Listing 9.2: B.java

```
1  // Package P1
2  package com.P1;
3
4  // Class C with private variable and public method
5  public class C {
6      private int w;
7
8      public void setW(int w) {
9          this.w = w;
10     }
11
12     public int getW() {
13         return w;
14     }
15 }
```

Listing 9.3: C.java

```
1  // Package P2
2  package com.P2;
3
4  // Class D inherited from A with protected variable and public method
5  public class D extends com.P1.A {
6      protected int v;
7
8      public void setV(int v) {
9          this.v = v;
10     }
11
12     public int getV() {
13         return v;
14     }
15 }
```

Listing 9.4: D.java

```
1  // Package P2
2  package com.P2;
3
4  // Class E with private variable and public method
5  public class E {
6      private int u;
7
8      public void setU(int u) {
9          this.u = u;
10     }
11
12     public int getU() {
13         return u;
14     }
15 }
```

Listing 9.5: E.java

```
1
2  import com.P1.A;
3  import com.P1.B;
4  import com.P1.C;
5  import com.P2.D;
6  import com.P2.E;
7
8  public class PackageDemo {
9
10     public static void main(String[] args) {
11
12         // Creating an object of class A in package P1
13         A a = new A();
14         a.setX(10);
15         a.setY(20);
16         System.out.println("Value of x: " + a.getX());
17         System.out.println("Value of y: " + a.getY());
18
19         // Creating an object of class B in package P1
20         B b = new B();
21         b.setX(30);
22         b.setY(40);
23         b.setZ(50);
24         System.out.println("Value of x: " + b.getX());
25         System.out.println("Value of y: " + b.getY());
26         System.out.println("Value of z: " + b.getZ());
27
28         // Creating an object of class C in package P1
29         C c = new C();
30         c.setW(60);
31         System.out.println("Value of w: " + c.getW());
32
33         // Creating an object of class D in package P2
34         D d = new D();
35         d.setX(70);
36         d.setY(80);
37         d.setV(90);
38         System.out.println("Value of x: " + d.getX());
39         System.out.println("Value of y: " + d.getY());
40         System.out.println("Value of v: " + d.getV());
41
42         // Creating an object of class E in package P2
43         E e = new E();
44         e.setU(100);
45         System.out.println("Value of u: " + e.getU());
46     }
47 }
```

Listing 9.6: PackageDemo.java

The above files should be stored in the folder structure as shown in the below diagram.

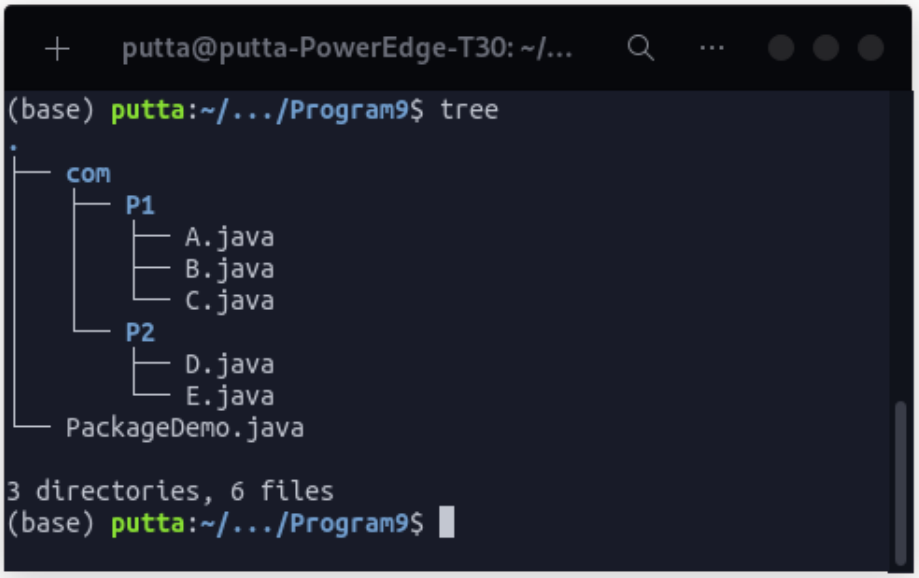

Figure 9.1: Package Structure

First create a folder **com** in the same folder where you saved *PackageDemo.java*. Then within **com** directory create two directories **P1** and **P2**. Within **P1** directory store the files *A.java*, *B.java* and *C.java*. Within **P2** directory store the files *D.java* and *E.java*.

Output

==========

```
putta:~/.../Program9$ javac PackageDemo.java
putta:~/.../Program9$ java PackageDemo
Value of x: 10
Value of y: 20
Value of x: 30
Value of y: 40
Value of z: 50
Value of w: 60
Value of x: 70
Value of y: 80
Value of v: 90
Value of u: 100
```

Listing 9.7: OUTPUT

Chapter 10

Exception Handling

Question

Write a JAVA program to read two integers a and b. Compute a/b and print, when b is not zero. Raise an exception when b is equal to zero. Also demonstrate working of ArrayIndexOutOfBoundException.

Java Code

```java
import java.util.Scanner;

public class ExceptionDemo {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter two integers: ");
        int a = sc.nextInt();
        int b = sc.nextInt();
        try {
            if (b == 0) {
                throw new ArithmeticException("Division by zero");
            }
            int result = a / b;
            System.out.println("Result of integer division: " + result);
        } catch (ArithmeticException e) {
            System.out.println(e.getMessage());
        }
        try {
            int[] arr = new int[5];
            arr[10] = 50;   //invalid index generates exception
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Listing 10.1: ExceptionDemo.java

Output

==========

```
putta:~/.../Program10$ javac ExceptionDemo.java
putta:~/.../Program10$ java ExceptionDemo

Enter two integers:
9 2
Result of integer division: 4
Index 10 out of bounds for length 5

putta:~/.../Program10$ java ExceptionDemo

Enter two integers:
3 0
Division by zero
Index 10 out of bounds for length 5
```

Listing 10.2: OUTPUT