OBJECT ORIENTED PROGRAMMING LAB MANUAL

II Year I Semester

2019 - 20

Prepared by:

Sri.Y.K.Viswanadham
Assoc. Professor

Mr. B.Sobhan Babu
Assistant Professor

GUDLAVALLERU ENGINEERING COLLEGE
(An Autonomous Institution with Permanent Affiliation to JNTUK, Kakinada)
Seshadri Rao Knowledge Village, Gudlavalleru – 521356

GUDLAVALLERU ENGINEERING COLLEGE

(An Autonomous Institution with Permanent Affiliation to JNTUK, Kakinada)
Seshadri Rao Knowledge Village, Gudlavalleru – 521356

DEPARTMENT OF INFORMATION TECHNOLOGY

INSTITUTE VISION & MISSION

Institute Vision:

To be a leading institution of engineering education and research, preparing students for leadership in their fields in a caring and challenging learning environment.

Institute Mission:

- To produce quality engineers by providing state-of-the-art engineering education.
- To attract and retain knowledgeable, creative, motivated and highly skilled individuals whose leadership and contributions uphold the college tenets of education, creativity, research and responsible public service.
- To develop faculty and resources to impart and disseminate knowledge and information to students and also to society that will enhance educational level, which in turn, will contribute to social and economic betterment of society.
- To provide an environment that values and encourages knowledge acquisition and academic freedom, making this a preferred institution for knowledge seekers.
- To provide quality assurance.
- To partner and collaborate with industry, government, and R&D institutes to develop new knowledge and sustainable technologies and serve as an engine for facilitating the nation’s economic development.
- To impart personality development skills to students that will help them to succeed and lead.

- To instil in students the attitude, values and vision that will prepare them to lead lives of personal integrity and civic responsibility.
- To promote a campus environment that welcomes and makes students of all races, cultures and civilizations feel at home.
- Putting students face to face with industrial, governmental and societal challenges.

DEPARTMENT VISION & MISSION

VISION

To be a centre of innovation by adopting changes in Information Technology, imparting quality education, research to produce visionary computer professionals and entrepreneurs.

MISSION

- ➢ To provide an academic environment in which students are given the essential resources for solving real-world problems and work in multidisciplinary teams.
- ➢ To impart value based education and research among students, particularly belonging to rural areas, for their sustained growth in technological aspects and leadership.
- ➢ To collaborate with the industry for making the students adoptable to evolving changes in Information Technology and related areas.

PROGRAMME EDUCATIONAL OBJECTIVES(PEOs):-

PEO1: To exhibit analytical skills in modeling and solving computing problems by applying mathematical, scientific and engineering knowledge and to pursue their higher studies.

PEO2: To communicate effectively with multi-disciplinary teams to develop quality software systems with an orientation towards research and development for lifelong

PEO3: To address industry and societal needs for the growth of global economy using emerging technologies by following professional ethics.

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **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.
3. **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.
4. **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.
5. **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.
6. **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.
7. **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.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **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.
11. **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.

12. **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.

PROGRAM SPECIFIC OUTCOMES

Students will be able to

13.

PSO1: Organize, maintain and protect IT Infrastructural resources.

PSO2: Design and Develop web, mobile, and smart apps based software solutions to the real

Course Objectives

- To demonstrate object oriented programming concepts.
- To introduce the creation of GUI using AWT components.

Course Outcomes

Upon successful completion of the course, the students will be able to

- use inheritance to extend the functionality of classes.
- prepare code that exhibits polymorphism.
- examine multi tasking with multi threading.
- create packages for reusability.
- create an effective GUI using AWT components.
- implement event handling.

OBJECT ORIENTED PROGRAMMING LAB	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**PSO 1**	**PSO 2**
CO1: use inheritance to extend the functionality of classes.	**3**	**3**	**3**		**2**			**2**	**3**	**2**		**2**		**3**
CO2: prepare code that exhibits polymorphism.	**3**	**3**	**3**		**2**			**2**	**3**	**2**		**2**		**3**
CO3: examine multi tasking with multi threading.	**3**	**3**	**3**		**2**			**2**	**3**	**2**		**2**		**3**
CO4: create packages for reusability.	**3**	**3**	**3**		**2**			**2**	**3**	**2**		**2**		**3**
CO5: create an effective GUI using AWT components..	**3**	**3**	**3**		**2**			**2**	**3**	**2**		**2**		**3**
CO6: implement event handling.	**3**	**3**	**3**		**2**			**2**	**3**	**2**		**2**		**3**
	3	**3**	**3**		**2**			**2**	**3**	**2**		**2**		**3**

Index

Additional Experiments

EXP NO:1 Write a Java code to display default initial values of all primitive data types in java

AIM: To display default initial values of all primitive data types in java

DESCRIPTION: **Primitive types** are the most basic **data types** available within the **Java** language. There are 8: boolean , byte , char , short , int , long , float and double . These **types** serve as the building blocks of **data** manipulation in **Java**.

ALGORITHM:

Different data types:

```
step 1: start
step 2: Class: Exp1
step 3: int a;
step 4: char b;
step 5: float c;
step 6: double d;
step 7: long e;
step 8: boolean f;
step 9: print Default initial value of byte all variables
step 10 :stop
```

Primitive Data types:

```
class Exp1
{
        int i;
        short s;
        boolean b;
        long l;
        byte by;
        double d;
        float f;
        char ch;
        static public void main(String[] args)
        {
                Exp1 p=new Exp1();
                System.out.println("Default initial value of byte variable is\t"+p.by);
                System.out.println("Default initial value of short variable is\t"+p.s);
                System.out.println("Default initial value of int variable is\t"+p.i);
                System.out.println("Default initial value of long variable is\t"+p.l);
                System.out.println("Default initial value of float variable is\t"+p.f);
                System.out.println("Default initial value of double variable is\t"+p.d);
                System.out.println("Default initial value of char variable is : "+p.ch);
                System.out.println("Default initial value of boolean variable is\t"+p.b);
        }}
```

Output:

```
Default initial value of byte variable is→0
Default initial value of byte variable is→0
Default initial value of short variable is→0
Default initial value of short variable is→0
Default initial value of int variable is→0
Default initial value of int variable is→0
Default initial value of long variable is→0
Default initial value of long variable is→0
Default initial value of float variable is→0.0
Default initial value of float variable is→0.0
Default initial value of double variable is→0.0
Default initial value of double variable is→0.0
Default initial value of char variable is :
Default initial value of char variable is :
Default initial value of boolean variable is→false
Default initial value of Boolean variable is→false
```

EXP NO: 2 Write a java program to sort given list of numbers

AIM: To sort given list of numbers in a java program.

DESCRIPTION:

Sorting is the process of arranging elements either in ascending order or descending order. Sorting can be done in many ways. Numbers of input elements are read into array and compare first number with remaining numbers and arrange the numbers in to its position, and second numbers is compared with remaining numbers and arrange that number in to its position and so on. Swapping process is used to arrange elements in sorting order.

ALGORITHM:

```
Step 1: Start the program.
Step 2: import utility package for 'Scanner' class.
Step 3: Write class name for writing the program.
      begin
Step 4: write 'main' function.
      begin
Step 5: Create Scanner object.
Step 6: Print "Enter size of an array:"/
Step 7: Read 'n' value.
Step 8: create Dynamic array.
Step 9: Print "Enter array elements:".
Step 10: for i=0 to n-1
        Loop
           Read array a[i].
        End loop.
Step 11: for i=0 to n-1
        Loop
           for j=i+1 to n-1
              Loop
                 If( a[i]>a[j]) then
```

```
                        t=a[i].
                       a[i]=a[j].
                      a[j]=t.
                  End if
                End loop
               End loop
               Print “Ascending order is:”.
                   for i=0 to n-1
                     loop
                        print array a[i].
                     End loop
                 End main
               End class
          Step 12: Stop.
```

PROGRAM:

```
import java.util.*;
class Sorting
{
public static void main(String args[])
{
Scanner s=new Scanner(System.in);
System.out.println("enter size of an array:");
int n=s.nextInt();
int a[]=new int[n];
System.out.println("enter array elements:");
for(int i=0;i<n;i++)
a[i]=s.nextInt();
for(int i=0;i<n;i++)
for(int j=i+1;j<n;j++)
if(a[i]>a[j])
{
int t=a[i];a[i]=a[j];a[j]=t;
}
System.out.println("Ascending order is:");
for(int i=0;i<n;i++)
System.out.println(a[i]);
}
}
```

OUTPUT:

```
K:\>javac Sorting.java
K:\>java Sorting
enter size of an array:
5
enter array elements:
12
1
21
3
69
Ascending order is:
1
3
12
21
69
```

Viva Questions:-

1. What is the time complexity of above sorting algorithm?
2. List different types of sorting techniques and which one is the best sorting algorithm?
3. What is internal sorting and external sorting?
4. What is bubble sort and why it is called as bubble sort?

EXP NO: 3 Write a Java code to test whether the given number is Armstrong number or not.

AIM: To test whether the given number is Armstrong number or not in java.

DESCRIPTION: J**ava Program** to Check **Armstrong** Number. ... In case of an **Armstrong** number of 3 digits, the sum of cubes of each digits is equal to the number itself. For example: 153 = 1*1*1 + 5*5*5 + 3*3*3 // 153 is an **Armstrong** number.

ALGORITHM:

```
Step 1: Class: Armstrong

Step 2: variable s=0,k,n
Step 3: while n>0 {
Step 4: s=s+(n%10)*(n%10)*(n%10);
step 5:  n/=10; }
step 6: if(k==s)
        1) k is an armstrong number  (or)
        2) k is not an armstrong number
Step 7: Stop.
```

Armstrong:

```
import java.util.*;
class Exp3
{
static public void main(String...args)
        {
        int s=0;
                Scanner s=new Scanner(System.in);
                System.out.println("Enter the number");
                int n=s.nextInt();
                int k=n;
                while(n>0)
                {
                        s=s+(n%10)*(n%10)*(n%10);
                        n/=10;
                }
                if(k==s)
                System.out.println("The given number "+k+" is an armstrong number");
                else
                System.out.println("The given number "+k+" is not an armstrong number");
        }}
```

Output:

```
Enter the number
153
The given number 153 is an armstrong number
```

EXP NO: 4 Write a java program to perform transpose of a given matrix

AIM: **T**o perform transpose of a given matrix in java.

DESCRIPTION: The **transpose of a matrix** is a new **matrix** whose rows are the columns of the original. This means when you **transpose** a **matrix** the columns of the new **matrix** becomes the rows of the original **matrix** and vice-versa. In short, to **transpose** a **matrix**, just swap the rows and columns of the **matrix**.

ALGORITHM:

```
Step 1: Start the program.
          Step 1: Write class name Exp4
          Step 2: write ‘main ()’ function
          Step 3:   Begin
                    Input order of matrix r
                    Input order of matrix c
          Step 4: print “The actual matrix is:"
          Step 5: swap the rows and columns of the matrix.
          Step 6: print “The transpose of the matrix is:”
          Step 7: Stop the program.
```

Transpose of Matrix:

```
import java.util.Scanner;
class Exp4
{
public static void main(String[] args)
{
Scanner s=new Scanner(System.in);
System.out.println("Enter order of the matrix:");
int r=s.nextInt();
int c=s.nextInt();
System.out.println("Enter elements into the matrix:");
int a[][]=new int[r][c];
for(int i=0;i<r;i++)
for(int j=0;j<c;j++)
a[i][j]=s.nextInt();
System.out.println("The actual matrix is:");
for(int i=0;i<r;i++)
{
        for(int j=0;j<c;j++)
        System.out.print(a[i][j]+"\t");
        System.out.println();
}
System.out.println("The transpose of the matrix is:");
for(int i=0;i<c;i++)
{
        for(int j=0;j<r;j++)
        System.out.print(a[j][i]+"\t");
        System.out.println();
}
}
}
```

Output:

```
Enter order of the matrix:
2       2
Enter elements into the matrix:
1       1
2       2
The actual matrix is:
1  1
2  2
The transpose of the matrix is:
    1   2
    1   2
```

EXP NO: 5 Write JAVA code to access the instance variables by using 'this' keyword.

AIM: To access instance variables in Java by using 'this' keyword

DESCRIPTION: *this* is a keyword in Java. It can be used inside the m*ethod* or *constructor* of a class. It(*this)* works as a reference to the current object, whose method or constructor is being invoked. This keyword can be used to refer to any member of the current object from within an instance method or a constructor.

ALGORITHM:

```
Step1: Class: class Account
Step2: Instance variable a and b
Step 3: To set value for a and b
this.a =a;
this.b =b;
Step 4 Method Show data: To display the values for a and b.
Step 5 Main method: To  create an object for Account class and call methods set data and show
                         data.
```

Program:

```
class Account{
int a;
int b;

 public void setData(int a ,int b){
  this.a = a;
  this.b = b;
 }
 public void showData(){
   System.out.println("Value of A ="+a);
   System.out.println("Value of B ="+b);
 }
 public static void main(String args[]){
   Account obj = new Account();
   obj.setData(2,3);
   obj.showData();
 }
}
```

Output:

```
c:\staff\javac Account.java
c:\staff\java Account
Value of A=2
Value of B=3
```

EXP NO: 6 Write a java program to illustrate the use of ‘super’ keyword

AIM: To illustrate the use of ‘super’ keyword in a java program

DESCRIPTION:

Generally the ‘super’ keyword is only applicable to inheritance programs. There is more number of advantages of super keyword. The following are some important advantages of ‘super’ keyword.

1. ‘super’ keyword is useful to access base class instance variables and base class methods.
2. ‘super’ keyword is also useful to access base class constructors.
3. Overridden methods are also accessed with ‘super’ keyword.

ALGORITHM 1: (For accessing base class instance variable and base class method).

```
Step 1: Start the program.
Step 2: Write class name A
             Begin
             a=100;
            Function display()
             begin
               Print  “It is class A”.
             End function
           End class.
Step 3: Write another class B inherits the properties of Class A
             Begin
                Function display ()
                 Begin
                  super.display().   (Calling of above class A function)
                  Print “It is class B”.
                  Print “Class A instance variable is:”super.a
                 End function.
              End class.
Step 4: Write class S
              Begin
Step 5: write ‘main ()’ function
              Begin
                        Create object k for class B.
                        Call the display function with object k.
               End main
               End class S
Step 6: Stop the program.
```

PROGRAM-1:

```
class A
{
int a=100;
void display()
{
System.out.println("It is class A");
}
}
class B extends A
{
void display()  //overridden method
{
super.display();   //calling of super class method
System.out.println("It is class B");
System.out.println("class A instance variable value is:"+super.a);  //accessing of super class instance variable
}
}
class S
{
public static void main(String[] args)
{
B k=new B();
k.display();
}
}
```

Output:

```
E:\ >javac S.java
E:\ >java S
It is class A
It is class B
class A instance variable value is:100
```

ALGORITHM 2: (For accessing base class constructor).

```
Step 1: Start the program.
 Step 2: Write class name Base
          Begin
Step 3: declare two instance variables a,b.
Step 4: create constructor with class name
        Base( int x,int y)
        Begin
                a=x; b=y;
                Print “It is Base Class”.
        End Base constructor.
        End Base class.
Step 5: Create another class ‘Derived’; it inherits the properties of Base class
Begin
        Declare two instance variables x, y.
        Create constructor for derived class with four parameters.
                Derived (int m,int n, int p, int q)
                Begin
                        Calling of base class constructor with ‘super’ keyword
                        super (m, n);  //java keyword.
                        x=p.
                        y=q.
                        Print “It is Derived class:”
                        Print a and b values. (a, b are base class instance variables)
                        Print x and y values. (x, y are derived class instance variables)
                 End Derived constructor.
                 End Class Derived.
Step 6: Create class with name ‘S2’.
Begin
Begin ‘main ()’
        Crete Object for Derived class d with four parameters. (10, 20, 30, 40)
End main.
End class S2.
Step 7: Stop the program.
```

PROGRAM-2:

```
class Base
{
int a,b;
Base(int x,int y)  //parameterized constructor.
{
a=x;
b=y;
System.out.println("It is Base class:");
}
}
class Derived extends Base
{
int a,b;
Derived(int m,int n,int p,int q)
{
super(m,n); //calling of super class constructor with derived class constructor parameters.
a=p;
b=q;
System.out.println("It is Derived class:");
System.out.println("a= "+super.a+"  b="+super.b);
System.out.println("a= "+a+"  b="+b);
}
}
class S2
{
public static void main(String args[])
{
Derived d=new Derived(10,20,30,40);
}
}
```

Output:

```
E:\>javac S2.java
E:\>java S2
It is Base class:
It is Derived class:
a= 10  b=20
x= 30  y=40
```

Viva Questions:

1. What is the difference between function overloading and function overriding?
2. How to avoid overriding in java?
3. What is the need of ‘super’ keyword in java?
4. What are the keywords that are not used with ‘super’ keyword?
5. What is the need of constructor in java?
6. What are the advantages of inheritance?

EXP NO: 7 Write Java code to Overload a method called area(), where the method computes the area of a square if the number of parameters are 1 otherwise if the numbers of parameter are 2 , it needs to compute the area of a rectangle.

AIM: To compute the area of a square using Java code Overload a method.

DESCRIPTION: Method **Overloading** is a feature that allows a class to have more than one method having the same name, if their argument lists are different. It is similar to constructor **overloading in Java**, that allows a class to have more than one constructor having different argument lists.

ALGORITHM:

```
Step: Class: Exp7
Step: call method area(int x,int y)
Step: return (int x, int y)
Step: Main method
Step: Read values a,b,c
Step: Input Enter side of a square in meters
Step: Input Enter length of a rectangle in meters
Step: Input Enter breadth of a rectangle in meters
Step: Print Area of a square
```

Overload area:

```
import java.util.*;
class Exp7
{
int area(int k)
{
return k*k;
}
int area(int x,int y)
{
        return x*y;
}
public static void main(String...args)
{
        Scanner s=new Scanner(System.in);
        Exp7 e=new Exp7();
        System.out.println("Enter side of a square in meters: ");
        int a=s.nextInt();
        System.out.println("Enter length of a rectangle in meters: ");
        int b=s.nextInt();
        System.out.println("Enter breadth of a rectangle in meters: ");
        int c=s.nextInt();
        System.out.println("area of a square is "+e.area(a)+ " sqm");
        System.out.println("area of a rectangle is "+e.area(b,c)+" sqm");
        }
}
```

Output:

```
Enter side of a square in meters:
25
Enter length of a rectangle in meters:·
8
Enter breadth of a rectangle in meters:·
9
area of a square is 625 sqm
area of a rectangle is 72 sqm
```

EXP NO:8 Write a java program to Create an abstract class named shape, that contains an empty method named ‘numberofsides ()’. Provide three classes named Trapezoid, Triangle and Hexagon, such that each one of the classes contains only the method ‘numberofsides ()’, that contains the number of sides in the given geometrical figure

AIM: To Create an abstract class named shape method named ‘numberofsides ()’ with properties of three shape classes named Trapezoid, Triangle and Hexagon.

DESCRIPTION:

An abstract class contains abstract methods and non-abstract methods. Abstract methods do not contain definitions; just they are used in derived classes. The abstract methods in abstract class must use within Derived classes. Derived class methods are accessed with base class object by using abstract classes.

ALGORITHM 1:

```
Step 1: Start the program.
Step 2: Create an abstract class with name ‘shape’.
         Begin
             Create abstract function numberofsides().  (No definition).
        End abstract class.
Step 3: Create class Trapezoid, it extends the properties of ‘shape’ class.
        Begin
            Function numberofsides()
            Begin
                Print “Number of sides in Trapezoid are 4”.
            End function.
        End class.
Step 4: Create class Triangle, it also extends the properties of ‘shape’ class.
        Begin
            Function numberofsides()
            Begin
                Print “Number of sides in Triangle are 3”.
            End function.
        End class.
Step 5: Create class Hexagon, it also extends the properties of ‘shape’ class.
        Begin
            Function numberofsides()
            Begin
                Print “Number of sides in Hexagon are 6”.
            End function.
        End class.
Step 6: Create class ‘Sides’
        Begin
            Write ‘main’ function
             Begin
                Base class object s1=derived class Trapezoid.
                Base class object s2=derived class Triangle.
```

```
              Base class object s3=derived class Hexagon.
              Access ‘numberofsides()’ method with three objects s1,s2,s3.
          End main
         End class.
Step 7: Stop the program.
```

PROGRAM:

```
abstract class shape
{
abstract void numberofsides();
}
class Trapezoid extends shape
{
void numberofsides()
{
System.out.println("Number of sides in Trapezoid are 4");
}
}
class Triangle extends shape
{
void numberofsides()
{
System.out.println("Number of sides in Triangle are  3");
}
}
class Hexagon extends shape
{
void numberofsides()
{
System.out.println("Number of sides in Hexagon are 6");
}
}
class Sides
{
public static void main(String args[])
{
shape s1=new Trapezoid();
shape s2=new Triangle();
shape s3=new Hexagon();
s1.numberofsides();
s2.numberofsides();
s3.numberofsides();
}
}
```

Output:

```
E:\>javac Sides.java
E:\>java Sides
Number of sides in Trapezoid are 4
Number of sides in Triangle are 3
Number of sides in Hexagon are 6
```

Viva Questions:

1. What is dynamic method dispatch in java?
2. What is the difference between runtime polymorphism and compile time polymorphism?
3. What is dynamic binding?
4. Java supports global variables or not?
5. Give list of some OOP languages?

EXP NO: 9 Write a java program to illustrate multiple inheritances.

AIM: To illustrate multiple inheritances in a java program.

DESCRIPTION:

Java does not support multiple inheritances. The main problem with multiple inheritances is base class ambiguity in multi-path inheritance. This multiple inheritances feature is provided with interfaces in java. Interfaces are same as classes, but methods are abstract and public in interface. The variables in interface are final and static. By using ‘implements’ keyword interfaces are used with classes. Interface extends another interface; a class can implement more than one interface. All methods in interface are abstract.

ALGORITHM 1: (simple example with two interfaces).

```
Step 1: Start the program.
Step 2: Create interface with name A
            Begin
                  a=10, b=20.  (by default a, b are final and static).
                  Function display1().     (Abstract method).
           End interface A.
Step 3: Create another interface with name B.
           Begin
                  x=30, y=40.
                  Function display2().
           End interface B.
Step 4: Create class Inter implements interface A and interface B
            Begin
                  Function display1()
                  Begin
                        Print “Interface A method”.
                  End display1.
                  Function display1.
                  Begin
                        Print “Interface B method”.
                  End function display2.
                  Function display3 ()
                  Begin
                        k=a+b+x+y.
                        Print “k”.
                  End display3.
             End class Inter.
Step 5: Create class Multiple
            Begin
                  Write main function.
                     Begin
                        Create object for class Inter. (Inter r=new Inter()).
                        Access three function display1 (), display2 () and
```

```
                                        display3()  with  object 'r'.
                              End main function.
                              End class Multiple.
                    Step 6: Stop the program.
```

PROGRAM-1:

```
interface A
{
int a=10,b=20;
void display1();
}
interface B
{
int x=30,y=40;
void display2();
}
class Inter implements A,B
{
int k;
public void display1()
{
System.out.println("Interface A method");
}
public void display2()
{
System.out.println("Interface B method");
}
void display3()
{
k=a+b+x+y;
System.out.println(k);
}
}
class Multiple
{
public static void main(String args[])
{
Inter r=new Inter();
r.display1();
r.display2();
r.display3();
}
}
```

Output:

```
E:\>javac Multiple.java
E:\>java Multiple
Interface A method
Interface B method
100
```

<u>ALGORITHM 2</u>: (example with three interfaces).

```
Step 1: Start the program.
Step 2: create interface ‘sdetails’ for student details.
            Begin
                no=1220, name=”john”, address=”Gudivada”.
                Function display();   (abstract method).
            End ‘sdetails’ interface.
Step 3: create interface ‘smarks’ for student marks.
            Begin
                s1=55, s2=90, s3=44, s4=66, s5=90, s6=55. (six subjects marks).
                Function percentage ();
            End ‘smarks’ interface.
Step 4: create interface ‘college’ for student college details.
            Begin
                ccode=110, cname=”GEC”, caddress=”Gudlavalleru”.
            Function cdisplay().
            End ‘college’ interface.
Step 5: create class ‘Xyz’ implements above three interfaces ‘sdetails’, ‘smarks’, ‘college’.
        Begin
            Function display ()
            Begin
                Print “Given student number is:” no.
                Print “Given student name is:” name.
                Print “Given student address is:” address.
            End function.
            Function percentage ()
            Begin
                p=((s1+s2+s3+s4+s5+s6)/600)*100.
                Print “Student marks Percentage is:” p.
            End function.
            Function cdisplay ()
            Begin
                Print “Given student college code is:” ccode.
                Print “Given Student College name is” cname.
                Print “Given Student college address is:” caddress.
            End function.
          End class Xyz.
Step 6: Create class ‘Multiple’
        Begin
            Write ‘main’ function
            Begin
             Create object ‘m’ for ‘Xyz’ class.
             Call ‘display ()’, ‘percentage ()’ and ‘cdisplay ()’ by using object ‘m’.
            End main.
        End class.
Step 7: Stop the program.
```

PROGRAM-2:

```
interface sdetails
{
int no=543;
String name="K.Srikanth";
String address="Unikili";
void display();
}
interface smarks
{
int s1=55,s2=90,s3=44,s4=66,s5=90,s6=55;
void percentage();
}
interface college
{
int ccode=48;
String cname="GEC";
String caddress="Gudlavalleru";
void cdisplay();
}
class Xyz implements sdetails,smarks,college   //multiple inheritances with interfaces(abstract methods)
{
double p;
public void display()
{
System.out.println("Given student number is:"+no);
System.out.println("Given student name is:"+name);
System.out.println("Given student address is:"+address);
}
public void percentage()
{
p=((double)(s1+s2+s3+s4+s5+s6)/600)*100;
System.out.println("Student marks Percentage is:"+p);
}
public void cdisplay()
{
System.out.println("Given student college code is:"+ccode);
System.out.println("Given student college name is:"+cname);
System.out.println("Given student college address is:"+caddress);
}
}
class Multiple
{
public static void main(String args[])
{
```

```
Xyz m=new Xyz();
m.display();
m.percentage();
m.cdisplay();
}
}
```

Output:

```
E:\>javac Multiple.java
E:\>java Multiple
Given student number is: 543
Given student name is: K.Srikanth
Given student address is: Unikili
Student marks Percentage is: 66.66666666666666
Given Student College code is: 48
Given Student College name is: GEC
Given Student College address is: Gudlavalleru
```

Viva Questions:

1. What are the advantages of interfaces?
2. What is abstract method and abstract class?
3. What are the advantages of ‘final’ keyword in java?
4. How to create object in java?
5. What is the main difference between interface and class?

EXP NO: 10 Write a java program to Create and access a user defined package where the package contains a class named CircleDemo, which in turn contains a method called circleArea() which takes radius of the circle as the parameter and returns the area of the circle.

AIM: To Create and access a user defined package and a class named CircleDemo, in which circleArea() which takes radius of the circle as the parameter and returns the area of the circle.

DESCRIPTION:

Generally packages are two types first one is built in packages, second one is user defined packages. Built in packages are import java.io.*. import java.util.*, etc. These packages are under the control of JVM (Java virtual Machine). User defined packages contains two steps, first one is creating a package and second one is importing that created package into another program. Package class, instance variables and its methods are in public access modifiers. Radius of circle is calculated by πr^2. Packages avoid name conflicts between classes and its methods. Namespace problems are avoided with help of packages. Package hierarchy is also possible.

ALGORITHM 1: (for creating package)

```
    Step 1: Start the program.
    Step 2: Create package named as ‘p’.
    Step 3: Write public class CircleDemo
            Begin
                    Declare constant value for PI=3.14.
             Write public function CircleArea(double r)
            Begin
                    Return PI*r*r.
            End function.
            End class.
```

PROGRAM 1: (for creating package)

```
package p;
public class CircleDemo
{
final double pi=3.14;
public double circleArea(double r)
{
return pi*r*r;
}
}
```

ALGORITHM 2: (importing package)

```
    Step 1: Start the program.
    Step 2: import the above created package ‘p’ with import keyword.
    Step 3: import java.util.* package for scanner class.
    Step 4: Write a class Area
            Begin
    Step 5: Write main function
```

```
        Begin
            Create Scanner object s.
            Print “Enter radius of circle:”.
            Read radius into variable ‘v’.
            Create object ‘z’ for ‘CircleDemo’ class which is from package ‘p’.
            Print “Area of Circle is” z.circleArea(v).
        End main.
        End class.
    Step 6: Stop the program.
```

PROGRAM 2: (importing package)

```
import p.*;
import java.util.*;
class Area
{
public static void main(String args[])
{
Scanner s=new Scanner(System.in);
System.out.println("Enter radius of Circle:");
double v=s.nextDouble();
CircleDemo z=new CircleDemo();
System.out.println("Area of circle is:"+z.circleArea(v));
}
}
```

Output:

```
E:\>javac –d E:\ CircleDemo.java
E:\>javac Area.java
E:\>java Area
Enter radius of Circle:
5
Area of circle is: 78.5
```

Viva Questions:

1. What is the advantage of using packages?
2. Define package in Java?
3. List some predefined or built in packages of java?
4. What is the meaning of ‘*’ symbol in importing packages?
5. What is namespace problem, who manages the namespaces in computer system?
6. How to access package classes and methods with qualified names?

EXP NO: 11 Write a java program to Create three threads (by using **Thread** class and **Runnable** interface) where the First thread displays “Good Morning” every one second, the second thread displays “Hello” every two seconds and the third thread displays “Welcome” every three seconds.

AIM: To Create three threads by using **Thread** class and **Runnable** interface one thread displays every second “Good Morning”, the second displays “Hello” and the third thread displays “Welcome” every three seconds.

.

DESCRIPTION:

Thread is light weight process that shares common address space. Threads are created in two ways, by extending Thread class and by implementing Runnable interface. Multithreading is most useful in java program. Some of the threads are paused while some other threads are running. Synchronization useful to, enter only one thread at a time into critical section. Threads are mutually exclusive to enter into critical section.

ALGORITHM:

```
Step 1: Start the program.
Step 2: Write a Class Mythread1 which inherits the properties from Thread class.
        Begin
            Write synchronized public method run ()
             Begin
                 For i=1 to 10
                 Loop
                     Suspend Thread with 1second.
                     Print “Good Morning”.
                 End loop.
             End run.
        End class.
Step 3: Write another class Mythread2 which implements Runnable interface.
       Begin
            Write synchronized public method run ()
             Begin
                 For i=1 to 10
                 Loop
                     Suspend Thread with 2seconds.
                     Print “Hello”.
                 End loop.
             End run.
        End class.
Step 4: Write a Class Mythread3 which inherits the properties from Thread class.
        Begin
            Write synchronized public method run ()
             Begin
                 For i=1 to 10
```

```
                Loop
                        Suspend Thread with 3 seconds.
                        Print “Welcome”.
                End loop.
          End run.
      End class.
Step 5: Create class named Thread5
        Begin
Step 6: Write main function
        Begin
                Create object for class Mythread1 namely t1.
                Create object for class Mythread2 namely t2.
                Create object for class Mythread3 namely t3.
                Create Thread object y and assign t2 object with Thread class.
                Start t1 Thread by using t1.start() method.
                Start y Thread object by using y.start() method.
                Start t3 Thread by using t3.start() method.
        End main.
        End class.
Step 7: Stop the program.
```

PROGRAM:

```
class Mythread1 extends Thread
{
synchronized public void run()
{
try
{
for(int i=0;i<10;i++)
{
Thread.sleep(1000);
System.out.println("Good morning");
}
}
catch(Exception e)
{
}
}
}
class Mythread2 implements Runnable
{
synchronized public void run()
{
try
{
```

```
for(int i=0;i<10;i++)
{
Thread.sleep(2000);
System.out.println("hello");
}
}
catch(Exception e)
{
}
}
}
class Mythread3 extends Thread
{
synchronized public void run()
{
try
{
for(int i=0;i<10;i++)
{
Thread.sleep(3000);
System.out.println("welcome");
}}
catch(Exception e)
{
}
}
}
class Thread5
{
public static void main(String args[])
{
Mythread1 t1=new Mythread1();
Mythread2 t2=new Mythread2(); //implements Runnable
Mythread3 t3=new Mythread3();
Thread y=new Thread(t2);   //implements Runnable,
t1.start();
y.start();
t3.start();
}
}.
```

Output:

```
E:\>javac Thread5.java
E:\>java Thread5

Good morning
Good morning
hello
Good morning
welcome
Good morning
hello
Good morning
: : : : : : : :
```

Viva Questions:

1. What is Race around condition, how it is eliminated in java?
2. List different Thread constructors?
3. What is the difference between ‘yield ()’ method and ‘suspend ()’ method?
4. How threads are useful in java?
5. What is Inter Thread Communication?
6. What are different types of Thread priorities in java?
7. What is the need of ‘isAlive()’ method in Threads?

EXP NO: 12 Write a java program to Display the position of x and y co-ordinates of the cursor movement using mouse.

AIM: To Display the position of x and y co-ordinates of the cursor movement using mouse in a java program.

DESCRIPTION:

The positions of x and y axis coordinates are determined with two interfaces MouseListener, MouseMotionListener. These two interfaces are most useful to get different Moues event abstract methods. ‘getX()’ and ‘getY()’ methods are used to get x and y coordinates. In this program the ‘mouseMoved()’ method used to get the values of x-coordinate and y-coordinate, remaining methods must be placed without anybody(because they are abstract methods). The applet programs are must include within ‘html’ code, and run with ‘appletViewer’ command.

ALGORITHM :

Step 1: Start the program.

```
Step 2: import three packages (java.awt.*; , java.applet.*; ,  java.awt.event.*;)
Step 3: Create a class with public specifier and extends that class with Applet class, implement
        with MouseListener, MouseMotionListener interfaces.
         Begin
         Initialize two variables x=0, y=0.
      Create Function init() with public specifier.
`       Begin
                 Set applet background color to red.
                 Set foreground color of applet to magenta.
         Add mouse motion listener event to applet window for detecting mouse movements.
         End function.
         Create function mouseMoved (MouseEvent e)
         Begin
              X=e.getX().  ‘get x-coordinate into variable x’
              Y=e.getY().  ‘get y-coordinate into variable y’
         End function.
         Create function mouseReleased(MouseEvent e) with public access specifier
         Begin

         End function.
         Create function mouseEntered(MouseEvent e) with public access  specifier
         Begin

         End function.
         Create function mouseExited(MouseEvent e) with public access specifier
         Begin

         End function.
         Create function mousePressed(MouseEvent e) with public access specifier
         Begin

         End function.
         Create function mouseClicked(MouseEvent e) with public access specifier
         Begin
         End function.
         Create function paint(Graphics g) with public access specifier
         Begin
         Set required font name, style and size to display x, y-coordinate values in applet window.
                 Draw x-coordinate output into applet window with specified position.
                 Draw y-coordinate output into applet window with specified position.
         End function.
End class.
Step 4: stop the program.
```

PROGRAM:

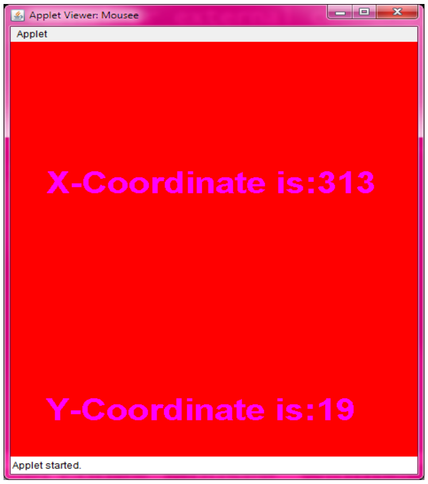
```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;
public class Mousee extends Applet implements MouseListener, MouseMotionListener
{
int x=0,y=0;
public void init()
{
setBackground(Color.red);
setForeground(Color.magenta);
addMouseMotionListener(this);
}
public void mouseMoved(MouseEvent e)
{
x=e.getX();
y=e.getY();
repaint();
}
public void mouseReleased(MouseEvent e)
{}
public void mouseEntered(MouseEvent e)
{
}
public void mouseExited(MouseEvent e)
{}
public void mousePressed(MouseEvent e)
{

}
public void mouseClicked(MouseEvent e)
{

}
public void mouseDragged(MouseEvent e)
{
}
public void paint(Graphics g)
{
g.setFont(new Font("Arial",Font.BOLD,40));
g.drawString("X-Coordinate is:"+x,40,200);
g.drawString("Y-Coordinate is:"+y,40,500);
}
}
```

ALGORITHM for HTML code:

```
    Step 1: Start the program.
    Step 2: write ‘html’ beginning tag.
    Step 3: write an applet tag with code, width and height attributes.
    Step 4: write end tag for applet tag.
    Step 5: write end tag for html tag.
    Step 6: Stop the program.
```

OUTPUT:

```
K:\>javac Mousee.java
K:\>appletViewer Mousee.html
```



```
X-Coordinate is:313

Y-Coordinate is:19
Applet started.
```

Viva Questions:

1. What is the difference between remote applet and local applet?
2. What is the difference between application and applet?
3. How to execute applet programs?
4. What are different awt controls?
5. What is event and List different Mouse events?

EXP NO: 13 Write a java program to handle keyboard events, which echoes keystrokes to the applet window and shows the status of each key event in the status bar.

AIM: To handle keyboard events, which echoes keystrokes to the applet window and shows the status of each key event in the status bar in a java program.

DESCRIPTION:

The keyboard events are getting from KeyListener interface. This interface is useful to get keyboard events like key pressed, key released and key typed. Java contains different VK(Virtual Key) values for non-echoed keys. For example Enter key virtual key value is get by VK_ENTER command. The key event result is shown status bar which is located at the bottom of applet window. The typed key is printed in applet window.

ALGORITHM :

```
Step 1: Start the program.
Step 2: import three packages (java.awt.*; , java.applet.*; ,  java.awt.event.*;)
Step 3: Create a class with public specifier and extends that class with Applet class, implement
        With KeyListener
         Begin
         Initialize string variable str to blank.
         Create function init() with public access specifier
         Begin
         Set background color of an applet to green.
         Add key listener to applet window for detecting key events.
         End function.
         Create function keyPressed(KeyEvent e) with public access specifier.
         Begin
                 Show status “key pressed” on the bottom of applet window.

                 If(k.getting_key_code=enter key virtual key event) then
                         Show status “Enter key pressed:”
                         Call paint() function to perform repaint().
                 End if
         Create function keyReleased(KeyEvent e) with public access specifier
         Begin
                 Show Status “Key Up” on the bottom of applet window.
         End function.
         Create function KeyTyped(KeyEvent e) with public access specifier
         Begin
                 msg=concatenate previous msg value to present typed key value.
                 Call paint() function to perform repaint().
        End function.
         Create function paint(Graphics g) with public access specifier
         Begin
         Draw msg value on applet window with ‘drawString()’ method at specified locations.
         End function.
Step 4: Stop the program.
```

PROGRAM:

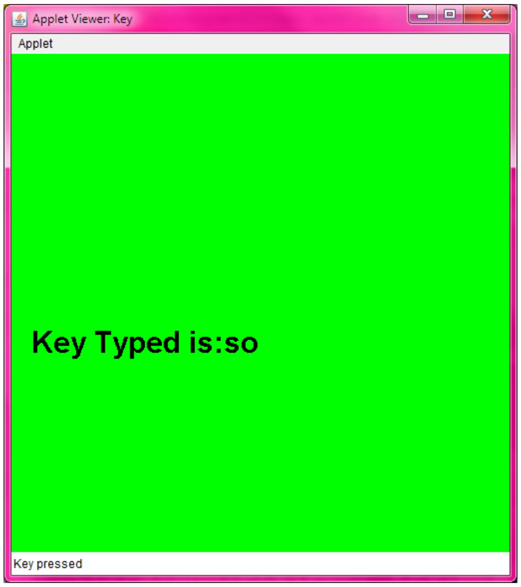
```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
public class Key extends Applet implements KeyListener
{
String msg="";
public void init()
{
setBackground(Color.green);
addKeyListener(this);
}
public void keyPressed(KeyEvent k)
{
showStatus("Key pressed");
if(k.getKeyCode()==KeyEvent.VK_ENTER)
showStatus("Enter key is pressed:");
repaint();
}
public void keyReleased(KeyEvent k)
{
showStatus("Key Up");
}
public void keyTyped(KeyEvent k)
{
msg+=k.getKeyChar();
repaint();
}
public void paint(Graphics g)
{
g.setFont(new Font("Arial",Font.BOLD,30));
g.drawString("Key Typed is:"+msg,20,300);
}
}
```

ALGORITHM for HTML code:

```
        Step 1: Start the program.
        Step 2: write ‘html’ beginning tag.
        Step 3: write an applet tag with code, width and height attributes.
        Step 4: write end tag for applet tag.
        Step 5: write end tag for html tag.
        Step 6: Stop the program.
```

OUPUT:

```
E:\ >javac Key.java
E:\ >appletViewer key.html
```



Key Typed is:so

Key pressed

Viva Questions:

1. What is VK in key board event? List different VK codes?
2. What is the ‘**showStatus()’** method?
3. What is the difference between ‘**combobox’** and ‘**Listbox’**?
4. Give any three real time examples of an applet?
5. What is echo key and non-echo key?

EXP NO: 14 Write a java program to create a list of items and display the selected item of the list in a text field. Arrange these components by using Flow Layout control.

AIM: To create a list of items and display the selected item of the list in a text field to arrange these components by using Flow Layout control.

DESCRIPTION:

Combo-box or drop-down list is used to create list of items. Among the list of items, the selected item is displayed with in text field. It uses 'ItemListener' interface to get selected item from the list of items. These two items are arranged in Flow layout control. Flow layout control is used to display the set of controls in left, right or middle in applet window. "itemStateChanged()' method is used to get selected item from the drop down list.

ALGORITHM :

```
Step 1: Start the program.
Step 2: import three packages (java.awt.*; , java.applet.*; ,  java.awt.event.*;)
Step 3: Create a class with public specifier and extends that class with Applet class, implement
        ItemListener
    Begin
        Declare choice object c1;
        Declare textfield object t1;
        Declare label object t2;
    Create function init() with public access specifier
    Begin
        Set background color to blue.
        Set foreground color to red
        Set layout to Left.
        Create objects for choice, texfield and labels.
        Add the items C++, JAVA, SmallTalk and Eiffel into list.
        Add choice, label and textfields to applet window.
        Add Item listener to choice object.
    Create function itemStateChanged(ItemEvent e) with public specifier
    Begin
        Call paint method as 'repaint().
    End function.
    Create function paint (Graphics g) with public specifier
    Begin
        Assign selected item in to textbox, with getSelectedItem() method.
    End function.
End class.
Step 4: Stop the program.
```

ALGORITHM for HTML code:

```
    Step 1: Start the program.
    Step 2: write ‘html’ beginning tag.
    Step 3: write an applet tag with code, width and height attributes.
    Step 4: write end tag for applet tag.
    Step 5: write end tag for html tag.
    Step 6: Stop the program.
```

PROGRM:

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
public class Combo extends Applet implements ItemListener
{
Choice c1;
TextField t1;
Label l1;
public void init()
{
setForeground(Color.blue);
setBackground(Color.red);
setLayout(new FlowLayout(FlowLayout.LEFT));
c1=new Choice();
t1=new TextField(10);
l1=new Label("Selected OOP's Language is:");
c1.addItem("C++");
c1.addItem("JAVA");
c1.addItem("Smalltalk");
c1.addItem("Eiffel");
add(c1);
add(l1);
add(t1);
c1.addItemListener(this);
}
public void paint(Graphics g)
{
t1.setText(c1.getSelectedItem());
}
public void itemStateChanged(ItemEvent e)
{
repaint();
}
}
```

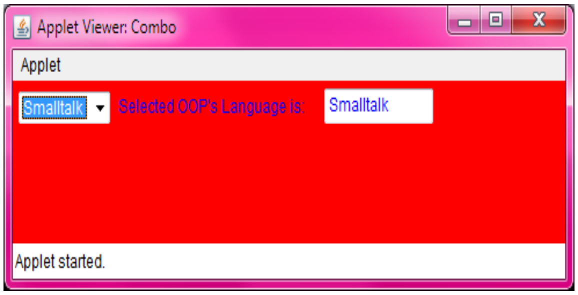
HTML code: (save with any file name.html).

```
<html>
<applet code="Combo" width=500 height=500>
```

```
</applet>
</html>
```

Output:

```
E:\>javac Combo.java
E:\>appletViewer combo.html
```



Smalltalk Selected OOP's Language is: Smalltalk

Applet started.

Viva Questions:

1. What is the difference between checkbox and radio button controls?
2. What is the need of flow layout control in java?
3. List different types of listeners in java?
4. What is the name of method used to set password characters in text field?
5. What is AWT?

EXP NO: 15 Write a java program to Arrange components by using Border Layout control.

AIM: To Arrange components by using Border Layout control in java program.

DESCRIPTION:

The class BorderLayout arranges the components to fit in the five regions: east, west, north, south and center. Each region is can contain only one component and each component in each region is identified by the corresponding constant NORTH, SOUTH, EAST, WEST, and CENTER.

ALGORITHM:

```
Step 1: Start the program.
Step 2: import three packages (java.awt.*; , java.applet.*; ,  java.awt.event.*;)
Step 3: Create a class with public specifier and extends that class with Applet
        Begin
                Declare Button objects b1, b2, b3, b4, b5;
        Create function init() with public access specifier
        Begin
        Set layout to Border layout.
Create 5 button objects for b1, b2, b3, b4, b5 with names North, South, West, East and Center
        Add b1 to North of border layout
        Add b2 to South of border layout.
        Add b3 to West of border layout.
        Add b4 to East of border layout.
        Add b5 to Center of border layout
  End function
Create function paint(Graphics g) with public access specifier
Begin
        Set font to 'arial', bold with size 30.
        Draw the string "BorderLayout" on applet window.
End function.
End class.
Step 4: Stop the program.
```

ALGORITHM for HTML code:

```
        Step 1: Start the program.
        Step 2: write 'html' beginning tag.
        Step 3: write an applet tag with code, width and height attributes.
        Step 4: write end tag for applet tag.
        Step 5: write end tag for html tag.
        Step 6: Stop the program.
```

PROGRAM:

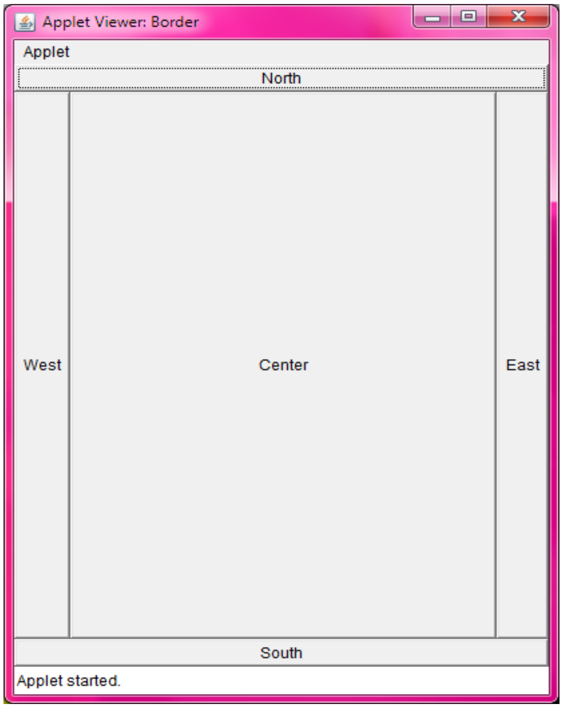
```
import java.applet.*;
import java.awt.*;
public class Border extends Applet
{
Button b1,b2,b3,b4,b5;
public void init()
{
setLayout(new BorderLayout());
b1=new Button("North");
b2=new Button("South");
b3=new Button("West");
b4=new Button("East");
b5=new Button("Center");
add(b1,BorderLayout.NORTH);
add(b2,BorderLayout.SOUTH);
add(b4,BorderLayout.EAST);
add(b3,BorderLayout.WEST);
add(b5,BorderLayout.CENTER);
}
public void paint(Graphics g)
{
g.setFont(new Font("Arial",Font.BOLD,30));
g.drawString("BorderLayout",80,100);
}
}
```

HTML code: (save with any filename.html)

```
<html>
<applet code="Border" width=400 height=500>
</applet>
</html>
```

Output:

```
E:\>javac Border.java
E:\>appletViewer Border.html
```



Applet started.

Viva Questions:

1. What is border layout?
2. What is Panel in AWT?
3. What is container and Grid Bag layout control in AWT?
4. How to set font for Button controls.
5. How to set background colors for AWT controls?

Additional Experiments

1. Write a java program to illustrate function overloading or parametric polymorphism?

Description: One name many forms is the concept of polymorphism. Function overloading achieves compile time polymorphism in java. The concept function overloading depends on number of arguments and type of arguments.

ALGORITHM:

```
Step 1: Start.
Step 2: Create a class
Step 3: Write a function to swap two integer values.
Step 4: Write a function to swap two double values.'
Step 5: Write a function to swap two strings.
Step 6: Write a function to swap two characters.
Step 7: Write 'main()' function.
Step 8: Create object for the class and call swap method with different arguments.
Step 9: Stop.
```

PROGRAM:

```
class Overload
{
void swap(int a,int b)
{
int t=a;
a=b;
b=t;
System.out.println("After swapping:a="+a+"  b="+b);
}
void swap(double a,double b)
{
double t=a;
a=b;
b=t;
System.out.println("After swapping:a="+a+"  b="+b);
}
void swap(String a,String b)
{
String t=a;
a=b;
b=t;
System.out.println("After Swapping:a="+a+"  b="+b);
}
void swap(char a,char b)
{
```

```
char t=a;
a=b;
b=t;
System.out.println("After Swapping:a="+a+"  b="+b);
}
public static void main(String[] args)
{
Overload d=new Overload();
d.swap("abc","xyz");
d.swap(33.22,44.4);
d.swap(23,5);
d.swap('p','q');
}}
```

Output:

```
D:\>javac Overload.java
D:\>java Overload
After Swapping:a=xyz  b=abc
After swapping:a=44.4  b=33.22
After swapping:a=5  b=23
After Swapping:a=q  b=p
```

Viva Questions:

1. What is ‘call by value’?
2. What are the applications of swapping?
3. What is inclusion polymorphism?
4. What is late binding and early binding in java?
5. What is Run time polymorphism?
6. What is call be reference?

2. Write a java program to illustrate 'call by value' and 'call by reference' in java?

Description: In call by value any modifications of formal arguments does not effect on actual arguments, where as in call by reference modifications of formal arguments must effect on actual arguments. Java does not support pointers to achieve call by reference. To achieve call by reference in java by passing object as argument to the function and modify arguments.

Algorithm for 'call by value':

Step 1: Start
Step 2: import utility package to read input through scanner object.
Step 3: create class to swap given two numbers.
Step 4: create swap method to swap given two variables.
Step 5: write main() method for creating object.
Step 6: Create scanner object and read two input values.
Step 7: call swap method with two input arguments by using class object.
Step 8: Stop.

PROGRAM:

```
import java.util.*;
class Swap1
{
void swap(int a,int b)
{
int t=a;
a=b;
b=t;
}
public static void main(String args[])
{
Scanner s=new Scanner(System.in);
System.out.println("Enter two values:");
int x=s.nextInt();
int y=s.nextInt();
Swap1 p=new Swap1();
System.out.println("Before swapping x="+x+"  y="+y);
p.swap(x,y);
System.out.println("After swapping x="+x+"  y="+y);
}
}
```

Output:

```
D:\>javac Swap1.java
D:\>java Swap1
Enter two values:
45
```

```
5
Before swapping x=45  y=5
After swapping x=45  y=5
```

Algorithm for ‘call By Reference’:

```
    Step 1: Start
    Step 2: import utility package to read input through scanner object.
    Step 3: create class to swap given two numbers
    Step 4: Create a method swap containing object as argument.
    Step 5: Swap given two input variables with instance variables.
    Step 6: Write ‘main()’ function.
    Step 7: Create scanner object to read two input values.
    Step 8: Create object for class.
    Step 9: Initialize instance variables with inputs.
    Step 10: Pass instance variables to ‘swap function’ with the help of object.
    Step 11: Stop.
```

PROGRAM:

```
import java.util.*;
class Swapping
{
int a,b;
void swap(Swapping p)
{
int t=p.a;
p.a=p.b;
p.b=t;
}
public static void main(String args[])
{
Swapping p=new Swapping();
Swapping z=new Swapping();
Scanner s=new Scanner(System.in);
System.out.println("Enter two values:");
p.a=s.nextInt();
p.b=s.nextInt();
System.out.println("Before swapping a="+p.a+"  b="+p.b);
z.swap(p);
System.out.println("After swapping a="+p.a+"  b="+p.b);
}
}
```

Output:

```
    D:\>javac Swapping.java
    D:\>java Swapping
```

```
Enter two values:
12
3
Before swapping a=12  b=3
After swapping a=3  b=12
```

Viva Questions:

1. What is the difference between function and method?
2. What is instance variable and data member?
3. What is anonymous object in Java?
4. What is dangling reference of object?
5. How to provide unconditional jump ‘goto’ facility in java?

3. Write a simple Java program to create user defined exceptions by extending Exception class.

Description: Exception is an error which is occurred during the execution of a program. Java contains default handler to handle exceptions. Most of the exceptions are predefined in java. Sometimes user want to creates own exception by using 'Exception' class. 'throw' keyword is used to create user own exception. Exceptions are belongs to import.java.lang.* package.

ALGORITHM:

```
Step 1: Start
Step 2: Create a class by extending 'Exception' class.
Step 3: create constructor with class name and call 'super()' method to pass given string to Exception class.
Step 4: close constructor and class.
Step 5: create another class with 'main' function.
Step 6: create object for the above class with 'throw' keyword and place that object within 'try' block.
Step 7:  write catch statement to handle user exception.
Step 8: close main method and class.
Step 9: Stop.
```

PROGRAM:

```
class Except extends Exception
{
Except(String s)
{
super(s);
}
}
class Exc
{
public static void main(String args[])
{
try
{
throw new Except("User defined Exception:");
}
catch(Exception e)
{
System.out.println(e);
}
}
}
```

Output:

```
D:\>javac Exc.java
D:\>java Exc
Except: User defined Exception
```

Viva Questions:

1. What is the difference between exception and error?
2. All Exceptions belongs to which package in java.
3. What is Daemon thread in java?
4. List some Thread Constructors in Java.
5. What is ‘NullPointerException’ in java?

4. Write a simple Java program to work with Multi-catch statement and finally block?

Description: An exception handling program contains only one ‘try’ block and many number of catch blocks. If any error occurred within try block then it goes to corresponding catch block. Finally block is used to release unused resources in a program. For example file closing operations and thread termination operation are performed in finally block. It is an optional block in exception handling.

ALGORITHM:

Step 1: Start
Step 2: create class with ‘main()’ function.
Step 3: Initialize string variable s=”**GEC**”.
Step 4: Initialize static array.
Step 5: get first command line argument into a.
Step 6: get second command line argument into b.
Step 7: get third command line argument into n.
Step 8: get fourth command line argument into m.
Step 9: Open try block.
Step 10: c=a/b
Step 11: print C.
Step 12: print nth character from string
Step 13: print mth array value from static array
Step 14: write three catch blocks to handle divide error, String index error and array index error.
Step 15: close ‘main ()’ method and class.
Step 16: Stop.

PROGRAM:

```
class Except1
{
public static void main(String args[])
{
String s="GEC";
int z[]={3,4,8};
int a=Integer.parseInt(args[0]);
int b=Integer.parseInt(args[1]);
int n=Integer.parseInt(args[2]);
int m=Integer.parseInt(args[3]);
try
{
int c=a/b;
System.out.println("division is:"+c);
System.out.println("character in string is:"+s.charAt(n));
System.out.println("Given array value is:"+z[m]);
}
catch(ArithmeticException e)
{
System.out.println("Divde error:"+e);
}
catch(ArrayIndexOutOfBoundsException e)
{
System.out.println("array boundary error:"+e);
}
catch(StringIndexOutOfBoundsException e)
{
System.out.println("String boundary error:"+e);
}
}
}
```

Output:

```
D:>javac Except1.java
D:\>java Except1 12  56  2  3
division is:0
character in string is:C
array boundary error:java.lang.ArrayIndexOutOfBoundsException: 3
```

Viva Questions:

1. What is the need of ‘Exception’ Class.?
2. How to handle multiple exceptions with single catch block.
3. What is the need of finally block in exception handling.
4. Is there any relation between throw and throws keyword.
5. It is possible to place more than one try block in java program justify?
6. What are wrapper classes in java?

5. Write a Java program to display Simple Banner using Applet window.

Description: Applet is small program to display output in graphical window. Applet starts by using **init()** method. ‘**init()**’ method contains initialization of controls and applet window properties. In this program we have to use both Threads and applets for displaying Banner. Threads are useful to perform concurrent operations and by using sleep method we have to display character by character from given input string. Threads completely depends on **start()** method and **run()** methods.

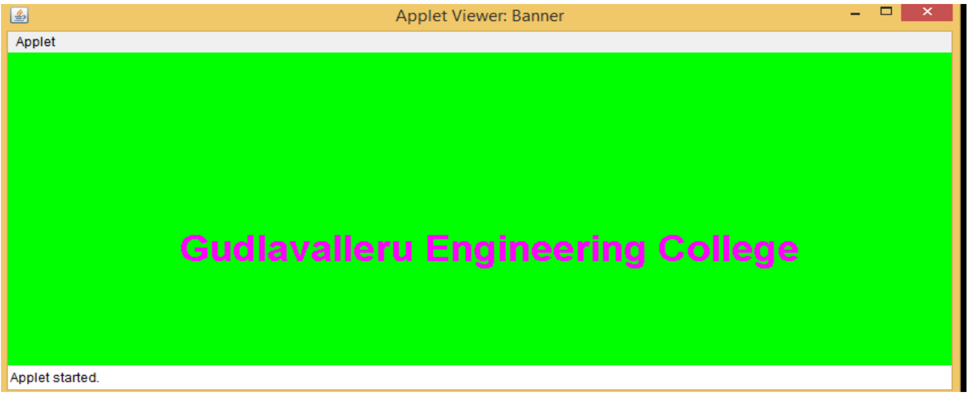
ALGORITHM:

```
Step 1: Start
Step 2: import awt, applet packages which are required for banner applet program.
Step 3: Create public class with name Banner which extends Applet class and implements Runnable interface.
Step 4:  Initialize the string s=”Gudlavalleru Engineering College”.
Step 5:  Declare Thread variable t and Boolean variable b.
Step 6:  Create public init() method to initialize background and foreground colors of an applet window.
Step 7: Create public start method to start thread with run() method.
Step 8:  Create public run method to display banner by using initialized string with the following code.
        Declare character variable ch.
        Write infinite for loop to display banner continuously.
        Open try block
        Call repaint() method.
        Sleep thread with 100 micro seconds.
        Select first character from string and assign it to variable ch.
        Select sub string from second position to its length.
        Concatenate first character to last character.
        Write required catch block for exception.
Step 9: Write public paint method wit graphics argument
Step 10: set font to display given string in applet window.
Step 11: Draw String on given x,y coordinates.
Step 12: Stop.
```

PROGRAM:

```
import java.awt.*;
import java.applet.*;
/*<applet code=Banner width=300 height=500>
</applet>*/
public class Banner extends Applet implements Runnable
{
  String s="Gudlavalleru Engineering College";
  Thread t;
  boolean b;
  public void init()
{
    setBackground(Color.green);
    setForeground(Color.magenta);
}
public void start()
{
    t=new Thread(this);
    b=false;
    t.start();
}
public void run()
{
    char ch;
    for(; ;)
{
    try
{
      repaint();
      Thread.sleep(100);
      ch = s.charAt(0);
      s=s.substring(1,s.length());
      s=s+ch;
    }
    catch(Exception e) {}
    }
  }
  public void paint(Graphics g)
{
        g.setFont(new Font("TimesNewRoman",Font.BOLD,34));
       g.drawString(s, 150, 200);
  }
}
```

OUTPUT:



Gudlavalleru Engineering College

Applet started.

Viva Questions:

1. What is multi threading?
2. How to create object for Runnable interface Thread?
3. What is the size of Boolean variable in java?
4. What are the advantages of an applet?