OOPS USING JAVA

NAME: PRITAM SHIL

REGISTRATION NUMBER: 1251811300204

ROLL NUMBER: 10021016900018

1. SHIP DATABASE USING JAVA :

PROGRAM:

```
import java.util.Scanner;
public class ship{
    int productId;
    String Pname;
}

class customer extends ship{
    String name;
    customer(String n,int p,String pn){
        name=n;
        productId=p;
        Pname=pn;
    }
    public static void main(String[] args){
        customer s=new customer("Ram",25,"pen");
        System.out.println("customer name:"+s.name+" product id:"+s.productId+" product
name:"+s.Pname);
    }
}
```

OUTPUT :

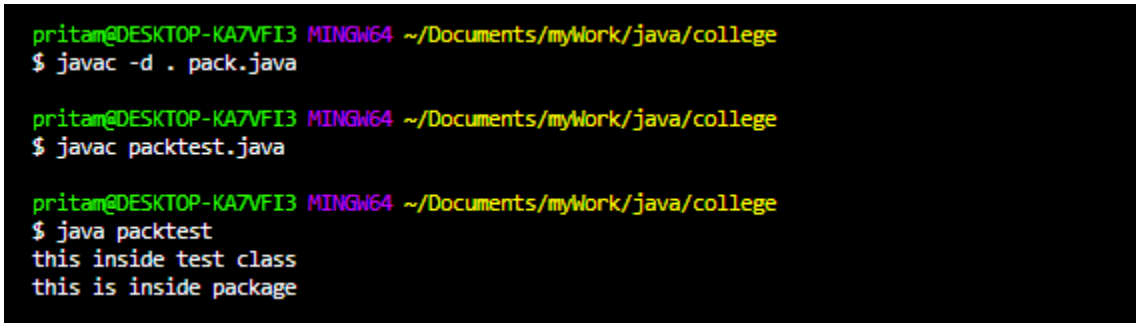
```
pritam@DESKTOP-KA7VFI3 MINGW64 ~/Documents/myWork/java/college
$ java customer
customer name:Ram product id:25 product name:pen
```

2. IMPLEMENT A PACKAGE :

PROGRAM:

```
package pack1;
public class pack {
    public static void test(){
        System.out.println("this is inside package");
    }
    public static void main(String[] args){
        System.out.println("this inside package");
    }
}

import pack1.pack;

public class packtest {
    public static void main(String[] args){
        System.out.println("this inside test class");
        pack p=new pack();
        p.test();
    }
}
```

OUTPUT :



3. IMPLEMENT A STACK :

PROGRAM:

```
import java.util.Scanner;

class stack {

   private int arr[];
   private int top;
   private int capacity;

   stack(int size) {
     arr = new int[size];
     capacity = size;
     top = -1;
   }

   public void push(int x) {
     if (isFull()) {
       System.out.println("Stack OverFlow");

       System.exit(1);
     }

     System.out.println("Inserting " + x);
     arr[++top] = x;
   }

   public int pop() {

     if (isEmpty()) {
       System.out.println("STACK EMPTY");
       System.exit(1);
     }

     return arr[top--];
   }

   public int getSize() {
     return top + 1;
   }

   public Boolean isEmpty() {
     return top == -1;
   }

   public Boolean isFull() {
     return top == capacity - 1;
   }

   public void printStack() {
     for (int i = 0; i <= top; i++) {
       System.out.print(arr[i] + ", ");
```

```
    }
  }

  public static void main(String[] args) {
    Scanner sc=new Scanner(System.in);
    System.out.println("enter the size of the stack");
    int a= sc.nextInt();
    stack stack = new stack(a);
    while (true){
      System.out.println("\n\nEnter your choice");
      System.out.println("1.push into stack");
      System.out.println("2.pop into stack");
      System.out.println("3.exit");
      a=sc.nextInt();
      switch(a) {
        case 1:
          System.out.println("enter the number you want to push");
          int b=sc.nextInt();
          stack.push(b);
          System.out.print("Stack:-> ");
          stack.printStack();
          break;
        case 2:
          stack.pop();
          System.out.println("\nAfter popping out\nStack:->");
          stack.printStack();
         break;
        case 3:
          System.exit(0);
        break;
        default:
         System.out.println("wrong input!!!");
       }
    }
  }
}
```

OUTPUT:

```
pritam@DESKTOP-KA7VFI3 MINGW64 ~/Documents/myWork/java/college
$ java stack
enter the size of the stack
5


Enter your choice
1.push into stack
2.pop into stack
3.exit
1
enter the number you want to push
2
Inserting 2
Stack:-> 2,

Enter your choice
1.push into stack
2.pop into stack
3.exit
1
enter the number you want to push
4
Inserting 4
Stack:-> 2, 4,

Enter your choice
1.push into stack
2.pop into stack
3.exit
2

After popping out
Stack:->
2,

Enter your choice
1.push into stack
2.pop into stack
3.exit
3
```

4. IMPLEMENT A QUEUE:

PROGRAM:

```
import java.util.Scanner;
public class queue {
   int SIZE ;
   int items[] ;
   int front, rear;

   queue(int size) {
      items=new int[size];
      SIZE=size;
      front = -1;
      rear = -1;
   }

   boolean isFull() {
     if (front == 0 && rear == SIZE - 1) {
       return true;
     }
     return false;
   }

   boolean isEmpty() {
     if (front == -1)
       return true;
     else
       return false;
   }

   void enQueue(int element) {

     if (isFull()) {
       System.out.println("Queue is full");
     }
     else {
       if (front == -1) {
         front = 0;
       }

       rear++;
       items[rear] = element;
       System.out.println("Insert " + element);
     }
   }

   int deQueue() {
     int element;

     if (isEmpty()) {
       System.out.println("Queue is empty");
       return (-1);
     }
```

```
  else {
    element = items[front];

    if (front >= rear) {
      front = -1;
      rear = -1;
    }
    else {
      front++;
    }
    System.out.println( element + " Deleted");
    return (element);
  }
}

void display() {
  int i;
  if (isEmpty()) {
    System.out.println("Empty Queue");
  }
  else {
    System.out.println("\nFront index-> " + front);

    System.out.println("Items -> ");
    for (i = front; i <= rear; i++)
      System.out.print(items[i] + "  ");

    System.out.println("\nRear index-> " + rear);
  }
}

public static void main(String[] args) {
    Scanner sc=new Scanner(System.in);
    System.out.println("enter the size of the Queue");
    int a= sc.nextInt();
    queue q = new queue(a);
    while (true){
        System.out.println("\n\nEnter your choice");
        System.out.println("1.EnQueue");
        System.out.println("2.DeQueue");
        System.out.println("3.exit");
        a=sc.nextInt();
        switch(a) {
            case 1:
                System.out.println("enter the number you want to put in the Queue");
                int b=sc.nextInt();
                q.enQueue(b);
                System.out.print("Queue: ");
                q.display();
                break;
            case 2:
                q.deQueue();
                System.out.println("\nAfter EnQueue\nQueue:");
                q.display();
              break;
            case 3:
```

```
                System.exit(0);
              break;
              default:
                System.out.println("wrong input!!!");
            }
        }
    }
}
```

OUTPUT:

```
pritam@DESKTOP-KA7VFI3 MINGW64 ~/Documents/myWork/java/college
$ java queue
enter the size of the Queue
2


Enter your choice
1.EnQueue
2.DeQueue
3.exit
1
enter the number you want to put in the Queue
4
Insert 4
Queue:
Front index-> 0
Items ->
4
Rear index-> 0


Enter your choice
1.EnQueue
2.DeQueue
3.exit
2
4 Deleted

After EnQueue
Queue:
Empty Queue


Enter your choice
1.EnQueue
2.DeQueue
3.exit
3
```

5. IMPLEMENT A STACK USING INTERFACE:

PROGRAM:

```
import java.util.Scanner;
interface stack{
    void printStack();
    Boolean isFull();
    Boolean isEmpty();
    int getSize();
    int pop();
    void push(int x);
}
class stackI {
    private int arr[];
    private int top;
    private int capacity;

    stackI(int size) {
      arr = new int[size];
      capacity = size;
      top = -1;
    }
    public void push(int x) {
      if (isFull()) {
        System.out.println("Stack OverFlow");
        System.exit(1);
      }
      System.out.println("Inserting " + x);
      arr[++top] = x;
    }
    public int pop() {
      if (isEmpty()) {
        System.out.println("STACK EMPTY");
        System.exit(1);
      }
      return arr[top--];
    }
    public int getSize() {
      return top + 1;
    }
    public Boolean isEmpty() {
      return top == -1;
    }
    public Boolean isFull() {
      return top == capacity - 1;
    }
    public void printStack() {
      for (int i = 0; i <= top; i++) {
        System.out.print(arr[i] + ", ");
      }
    }
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        System.out.println("enter the size of the stack");
```

```
        int a= sc.nextInt();
        stackI stack = new stackI(a);
        while (true){
            System.out.println("\n\nEnter your choice");
            System.out.println("1.push into stack");
            System.out.println("2.pop into stack");
            System.out.println("3.exit");
            a=sc.nextInt();
            switch(a) {
                case 1:
                    System.out.println("enter the number you want to push");
                    int b=sc.nextInt();
                    stack.push(b);
                    System.out.print("Stack:-> ");
                    stack.printStack();
                    break;
                case 2:
                    stack.pop();
                    System.out.println("\nAfter popping out\nStack:->");
                    stack.printStack();
                  break;
                case 3:
                    System.exit(0);
                break;
                default:
                  System.out.println("wrong input!!!");
              }
        }
    }
}
```

OUTPUT:

```
pritam@DESKTOP-KA7VFI3 MINGW64 ~/Documents/myWork/java/college
$ java queue
enter the size of the Queue
2


Enter your choice
1.EnQueue
2.DeQueue
3.exit
1
enter the number you want to put in the Queue
4
Insert 4
Queue:
Front index-> 0
Items ->
4
Rear index-> 0


Enter your choice
1.EnQueue
2.DeQueue
3.exit
2
4 Deleted

After EnQueue
Queue:
Empty Queue


Enter your choice
1.EnQueue
2.DeQueue
3.exit
3
```

6. IMPLEMENT A QUEUE USING INTERFACE :

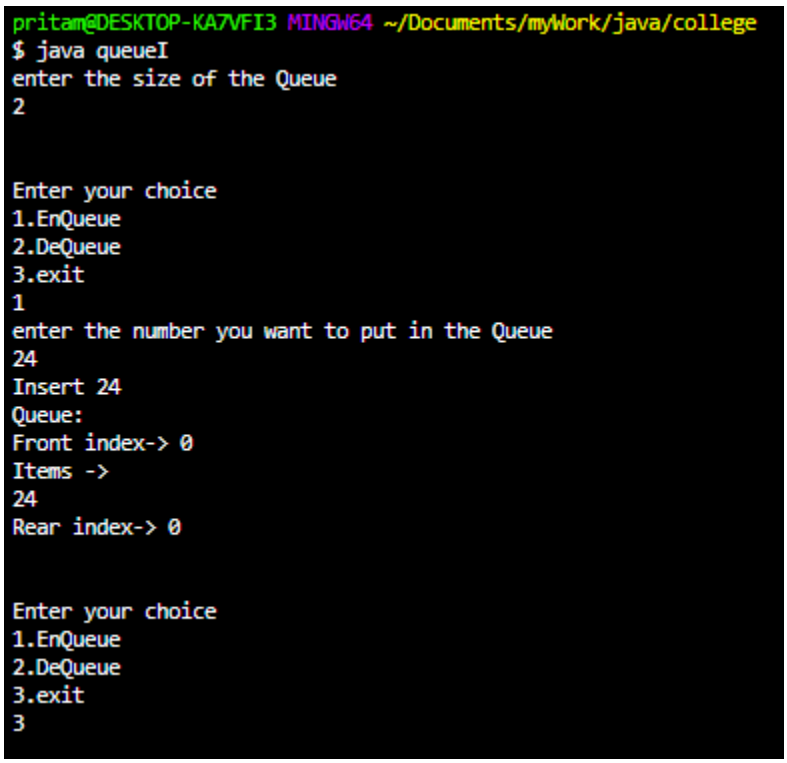
PROGRAM:

```
import java.util.Scanner;
interface queuea{
    boolean isFull();
    boolean isEmpty();
    void enQueue(int element);
    int deQueue();
    void display();
}
class queueI implements queue{
    int SIZE ;
    int items[] ;
    int front, rear;
    public queueI(int size) {
        items=new int[size];
        SIZE=size;
        front = -1;
        rear = -1;
    }
    public boolean isFull() {
      if (front == 0 && rear == SIZE - 1) {
        return true;
      }
      return false;
    }
    public boolean isEmpty() {
      if (front == -1)
        return true;
      else
        return false;
    }
    public void enQueue(int element) {
      if (isFull()) {
        System.out.println("Queue is full");
      }
      else {
        if (front == -1) {
          front = 0;
        }
        rear++;
        items[rear] = element;
        System.out.println("Insert " + element);
      }
    }
    public int deQueue() {
      int element;

      if (isEmpty()) {
        System.out.println("Queue is empty");
        return (-1);
      }
```

```
  else {
    element = items[front];

    if (front >= rear) {
      front = -1;
      rear = -1;
    }
    else {
      front++;
    }
    System.out.println( element + " Deleted");
    return (element);
  }
}
public void display() {
  int i;
  if (isEmpty()) {
    System.out.println("Empty Queue");
  }
  else {
    System.out.println("\nFront index-> " + front);

    System.out.println("Items -> ");
    for (i = front; i <= rear; i++)
      System.out.print(items[i] + "  ");

    System.out.println("\nRear index-> " + rear);
  }
}
public static void main(String[] args) {
    Scanner sc=new Scanner(System.in);
    System.out.println("enter the size of the Queue");
    int a= sc.nextInt();
    queueI q = new queueI(a);
    while (true){
        System.out.println("\n\nEnter your choice");
        System.out.println("1.EnQueue");
        System.out.println("2.DeQueue");
        System.out.println("3.exit");
        a=sc.nextInt();
        switch(a) {
            case 1:
                System.out.println("enter the number you want to put in the Queue");
                int b=sc.nextInt();
                q.enQueue(b);
                System.out.print("Queue: ");
                q.display();
                break;
            case 2:
                q.deQueue();
                System.out.println("\nAfter EnQueue\nQueue:");
                q.display();
              break;
            case 3:
                System.exit(0);
            break;
```

```
            default:
              System.out.println("wrong input!!!");
          }
      }
   }
}
```

OUTPUT:

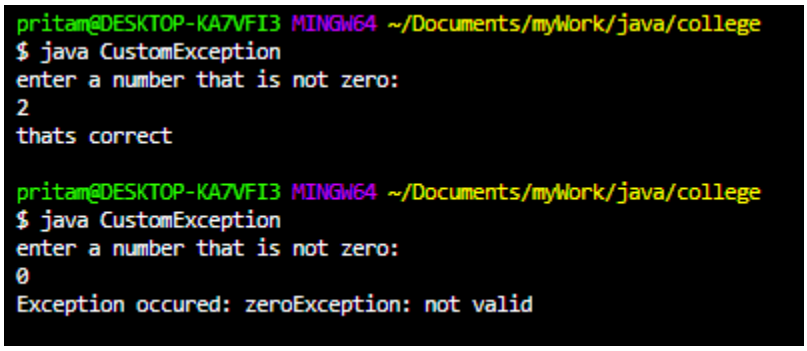


7. CREATE YOUR OWN EXCEPTION :

PROGRAM:

```
import java.util.Scanner;
class zeroException extends Exception{
   zeroException(String s){
                super(s);
        }
}
class CustomException{
        static void validate(int num)throws zeroException{
                if(num==0)
                        throw new zeroException("not valid");
                else
                        System.out.println("thats correct");
        }
        public static void main(String args[]){
                Scanner sc=new Scanner(System.in);
                try{
                        int num;
                        System.out.println("enter a number that is not zero:");
                        num=sc.nextInt();
                        validate(num);
                }
                catch(Exception m){
                        System.out.println("Exception occured: "+m);
                }
        }
}
```

OUTPUT:

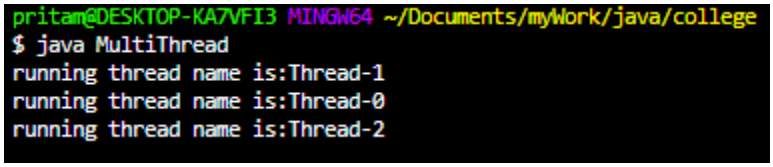


8. CREATE MULTIPLE THREADS:

PROGRAM:

```
class MultiThread extends Thread{
    public void run(){
         System.out.println("running thread name is:"+Thread.currentThread().getName());
     }
    public static void main(String args[]){
        MultiThread m1=new MultiThread();
        MultiThread m2=new MultiThread();
        MultiThread m3=new MultiThread();
         m1.start();
         m2.start();
         m3.start();
    }
}
```

OUTPUT:

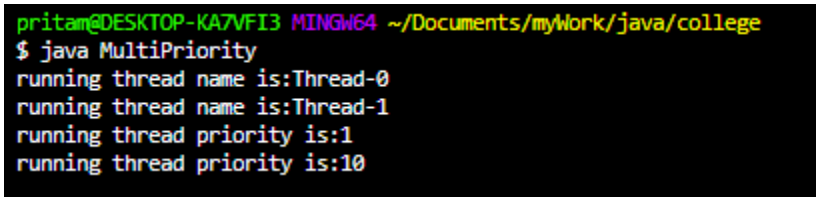


9. WAP TO ASSIGN PRIORITY IN THREADS :

PROGRAM:

```
class MultiPriority extends Thread{
    public void run(){
         System.out.println("running thread name is:"+Thread.currentThread().getName());
         System.out.println("running thread priority is:"+Thread.currentThread().getPriority());
     }
    public static void main(String args[]){
        MultiPriority m1=new MultiPriority();
        MultiPriority m2=new MultiPriority();
        m1.setPriority(Thread.MIN_PRIORITY);
        m2.setPriority(Thread.MAX_PRIORITY);
        m1.start();
        m2.start();
    }
}
```

OUTPUT:



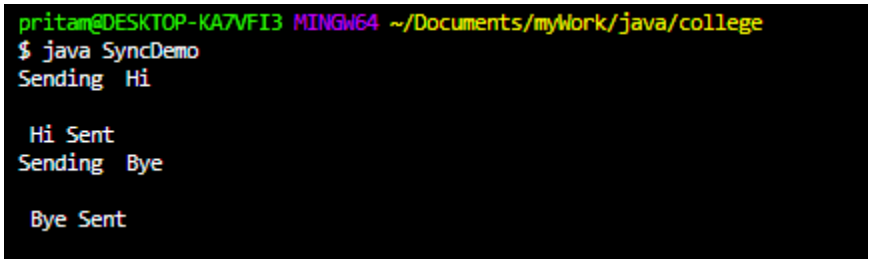
10. WAP TO SYNCHRONIZE MULTIPLE THREADS :

PROGRAM:

```
import java.io.*;
import java.util.*;
class Sender
{
   public void send(String msg)
   {
      System.out.println("Sending\t"  + msg );
      try
      {
         Thread.sleep(1000);
      }
      catch (Exception e)
      {
         System.out.println("Thread  interrupted.");
      }
      System.out.println("\n" + msg + "Sent");
   }
}
class ThreadedSend extends Thread
{
   private String msg;
   Sender  sender;
   ThreadedSend(String m,  Sender obj)
   {
      msg = m;
      sender = obj;
   }
   public void run()
   {
      synchronized(sender)
      {

         sender.send(msg);
      }
   }
}
class SyncDemo
{
   public static void main(String args[])
   {
      Sender send = new Sender();
      ThreadedSend S1 =
         new ThreadedSend( " Hi " , send );
      ThreadedSend S2 =
         new ThreadedSend( " Bye " , send );

      S1.start();
      S2.start();

      try
```

```
            {
                S1.join();
                S2.join();
            }
            catch(Exception e)
            {
                System.out.println("Interrupted");
            }
        }
    }
```

OUTPUT:



11. IMPLEMENT A BANKING DATABASE :

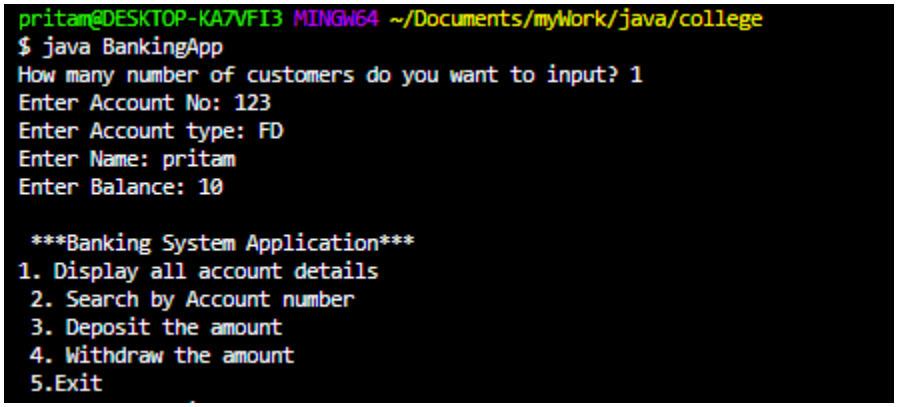
PROGRAM:

```
import java.util.Scanner;
class BankDetails {
    private String accno;
    private String name;
    private String acc_type;
    private long balance;
    Scanner sc = new Scanner(System.in);
    public void openAccount() {
        System.out.print("Enter Account No: ");
        accno = sc.next();
        System.out.print("Enter Account type: ");
        acc_type = sc.next();
        System.out.print("Enter Name: ");
        name = sc.next();
        System.out.print("Enter Balance: ");
        balance = sc.nextLong();
    }
    public void showAccount() {
        System.out.println("Name of account holder: " + name);
        System.out.println("Account no.: " + accno);
        System.out.println("Account type: " + acc_type);
        System.out.println("Balance: " + balance);
    }
    public void deposit() {
        long amt;
        System.out.println("Enter the amount you want to deposit: ");
        amt = sc.nextLong();
        balance = balance + amt;
    }
    public void withdrawal() {
        long amt;
        System.out.println("Enter the amount you want to withdraw: ");
        amt = sc.nextLong();
        if (balance >= amt) {
            balance = balance - amt;
            System.out.println("Balance after withdrawal: " + balance);
        } else {
            System.out.println("Your balance is less than " + amt + "\tTransaction failed...!!" );
        }
    }
    public boolean search(String ac_no) {
        if (accno.equals(ac_no)) {
            showAccount();
            return (true);
        }
        return (false);
    }
}
public class BankingApp {
    public static void main(String arg[]) {
```

```
        Scanner sc = new Scanner(System.in);
        System.out.print("How many number of customers do you want to input? ");
        int n = sc.nextInt();
        BankDetails C[] = new BankDetails[n];
        for (int i = 0; i < C.length; i++) {
            C[i] = new BankDetails();
            C[i].openAccount();
        }
        int ch;
        do {
            System.out.println("\n ***Banking System Application***");
            System.out.println("1. Display all account details \n 2. Search by Account number\n 3.
Deposit the amount \n 4. Withdraw the amount \n 5.Exit ");
            System.out.println("Enter your choice: ");
            ch = sc.nextInt();
                switch (ch) {
                    case 1:
                        for (int i = 0; i < C.length; i++) {
                            C[i].showAccount();
                        }
                        break;
                    case 2:
                        System.out.print("Enter account no. you want to search: ");
                        String ac_no = sc.next();
                        boolean found = false;
                        for (int i = 0; i < C.length; i++) {
                            found = C[i].search(ac_no);
                            if (found) {
                                break;
                            }
                        }
                        if (!found) {
                            System.out.println("Search failed! Account doesn't exist..!!");
                        }
                        break;
                    case 3:
                        System.out.print("Enter Account no. : ");
                        ac_no = sc.next();
                        found = false;
                        for (int i = 0; i < C.length; i++) {
                            found = C[i].search(ac_no);
                            if (found) {
                                C[i].deposit();
                                break;
                            }
                        }
                        if (!found) {
                            System.out.println("Search failed! Account doesn't exist..!!");
                        }
                        break;
                    case 4:
                        System.out.print("Enter Account No : ");
                        ac_no = sc.next();
                        found = false;
                        for (int i = 0; i < C.length; i++) {
                            found = C[i].search(ac_no);
```

```
                    if (found) {
                        C[i].withdrawal();
                        break;
                    }
                }
                if (!found) {
                    System.out.println("Search failed! Account doesn't exist..!!");
                }
                break;
            case 5:
                System.out.println("See you soon...");
                break;
        }
      }
      while (ch != 5);
   }
}
```

OUTPUT:



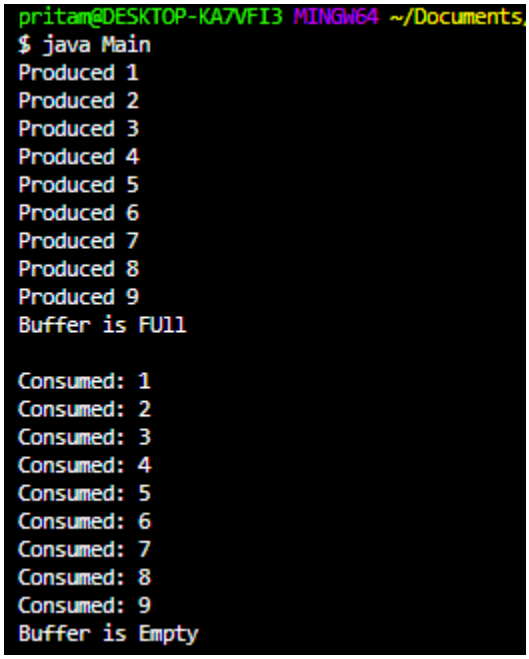
12. WAP TO IMPLEMENT PRODUCER-CONSUMER PROBLEM USING INTERTHREAD COMMUNICATION :

PROGRAM:

```
class Producer extends Thread {
  private final StringBuffer buffer;
  private final int size;
  Producer(int size) {
    this.size = size;
    buffer = new StringBuffer(size);
  }
  public void run() {
    synchronized (buffer) {
      for (int idx = 0; idx < size; idx++) {
        try {
          buffer.append(idx + 1);
          System.out.println("Produced " + (idx + 1));
        } catch (Exception e) {
          e.printStackTrace();
        }
      }
      System.out.println("Buffer is FUll\n");
      buffer.notify();
    }
  }
  public StringBuffer getBuffer() {
    return buffer;
  }
}
class Consumer extends Thread {
  private final Producer producer;
  Consumer(Producer prod) {
    this.producer = prod;
  }
  public void run() {
    synchronized (producer.getBuffer()) {
      try {
        producer.getBuffer().wait();
      } catch (Exception e) {
        e.printStackTrace();
      }
      for (int idx = 0; idx < producer.getBuffer().length(); idx++) {
        System.out.println("Consumed: " + producer.getBuffer().charAt(idx));
      }
      System.out.println("Buffer is Empty\n");
    }
  }
}

public class ProCon {
  public static void main(String[] args) {
```

```
    Producer producer = new Producer(9);
    Consumer consumer = new Consumer(producer);
    Thread thread1 = new Thread(producer);
    Thread thread2 = new Thread(consumer);
    thread2.start();
    thread1.start();
  }
}
```

OUTPUT:



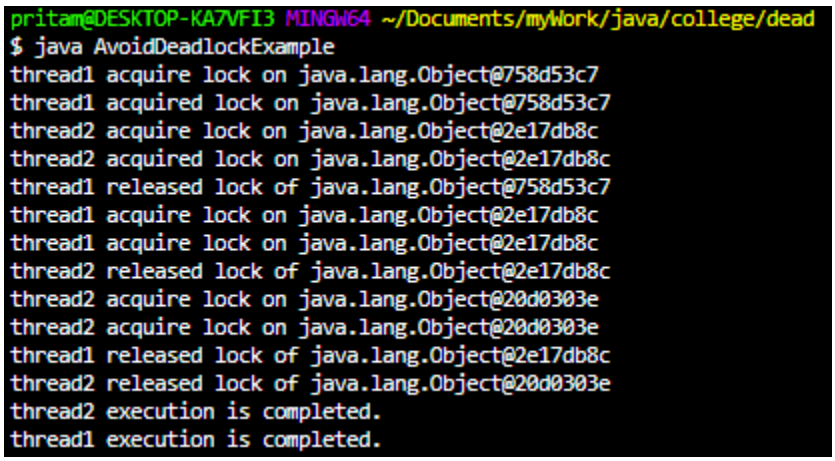
13. WAP TO AVOID DEADLOCK :

PROGRAM:

```
public class AvoidDeadlockExample
{
	public static void main(String[] args) throws InterruptedException
	{
		Object object1 = new Object();
		Object object2 = new Object();
		Object object3 = new Object();
		Thread thread1 = new Thread(new SynchroniseThread(object1, object2),
		"thread1");
		Thread thread2 = new Thread(new SynchroniseThread(object2, object3),
		"thread2");
		thread1.start();
		Thread.sleep(2000);
		thread2.start();
		Thread.sleep(2000);
	}
}
class SynchroniseThread implements Runnable
{
	private Object object1;
	private Object object2;
	public SynchroniseThread(Object o1, Object o2)
{
	this.object1=o1;
	this.object2=o2;
}
public void run()
{
	String name = Thread.currentThread().getName();
	System.out.println(name + " acquire lock on " + object1);
	synchronized (object1)
	{
		System.out.println(name + " acquired lock on " + object1);
		work();
	}
		System.out.println(name + " released lock of " + object1);
		System.out.println(name + " acquire lock on " + object2);
		synchronized (object2)
	{
		System.out.println(name + " acquire lock on " + object2);
		work();
	}
	System.out.println(name + " released lock of " + object2);
	System.out.println(name + " execution is completed.");
}
private void work()
{
	try
{
Thread.sleep(5000);
```

```
            }
            catch (InterruptedException e)
            {
                    e.printStackTrace();
            }
    }
}
```

OUTPUT:



14. WAP TO TAKE INPUT FROM THE KEYBOARD :

PROGRAM:

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
public class Input {
   public static void main(String[] args)
      throws IOException
   {
      BufferedReader reader = new BufferedReader(
         new InputStreamReader(System.in));

      String name = reader.readLine();

      System.out.println(name);
   }
}
```

OUTPUT:

15. WAP TO CREATE FILE AND COUNT VOWELS:

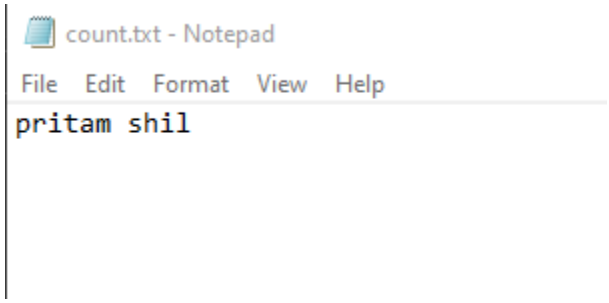
PROGRAM:

```
import java.io.FileWriter;
import java.io.File;
import java.io.IOException;
import java.util.Scanner;
import java.io.FileReader;
public class FileD {
    public static void main(String[] args){
        int vo[]={0,0,0,0,0};
        char vot[]={'a','e','i','o','u'};
        File myfile=new File("count.txt");
        Scanner sc= new Scanner(System.in);
        System.out.print("Enter a line you want to put in the file: ");
        String str= sc.nextLine();
        try{
            myfile.createNewFile();
            FileWriter myWriter = new FileWriter("count.txt");
            myWriter.write(str);
            myWriter.close();
            FileReader fr=new FileReader("count.txt");
            int i;
            while((i=fr.read())!=-1){
                if((char)i=='a'||(char)i=='A'){
                    vo[0]=vo[0]+1;
                }
                if((char)i=='e'||(char)i=='E'){
                    vo[1]=vo[1]+1;
                }
                if((char)i=='I'||(char)i=='i'){
                    vo[2]=vo[2]+1;
                }
                if((char)i=='O'||(char)i=='o'){
                    vo[3]=vo[3]+1;
                }
                if((char)i=='u'||(char)i=='U'){
                    vo[4]=vo[4]+1;
                }
            }
            fr.close();
        } catch (IOException e) {
            System.out.println("An error occurred.");
            e.printStackTrace();
        }
        System.out.println("The number of vowels are:");
        for(int i=0;i<5;i++){
            System.out.print(vot[i]+":"+vo[i]+"\t");
        }
    }
}
```

OUTPUT:

```
pritam@DESKTOP-KA7VFI3 MINGW64 ~/Documents/myWork/java/college
$ java FileD
Enter a line you want to put in the file: pritam shil
The number of vowels are:
a:1     e:0     i:2     o:0     u:0
```

FILE OUTPUT:

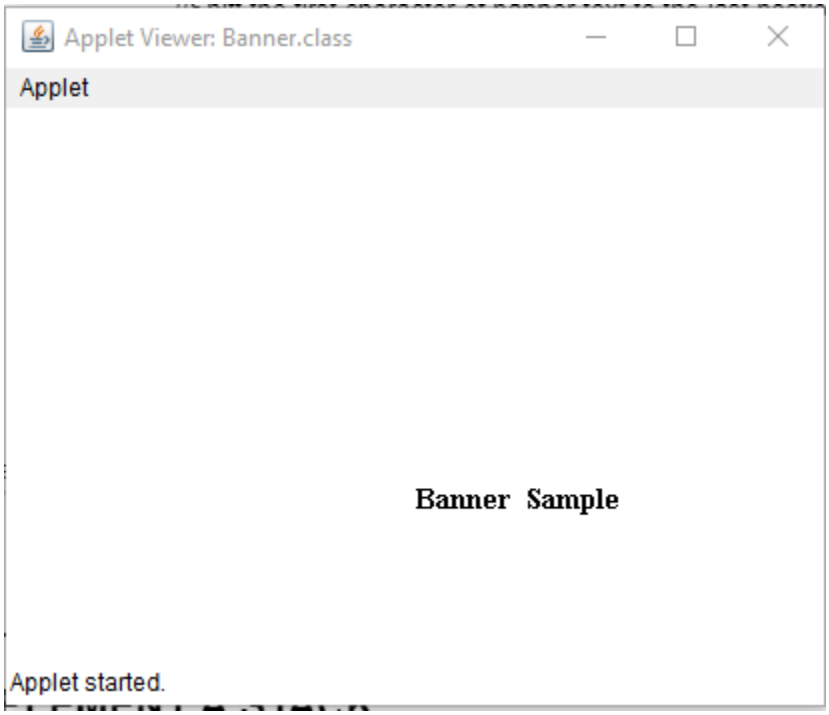


16. WAP FOR APPLET TO DISPLAY BANNER :

PROGRAM:

```
import java.applet.*;
import java.awt.*;
public class Banner extends Applet implements Runnable
{
   String text = " Simple Banner in java";
   Thread t;
   public void init()
   {
      setBackground(Color.white);
   }
   public void start()
   {
         t = new Thread(this);
         t.start();
   }
   public void run()
   {
      while(true)
         {
            try
            {
               repaint();
                  //Delay each thread by 1000ms or 1 seconds
              Thread.sleep(1000);
                  //Shift the first character of banner text to the last postion
              text = text.substring(1) + text.charAt(0);
            }
            catch(Exception e)
            {
            }
         }
   }
   public void paint(Graphics g)
   {
         g.setFont(new Font("TimesRoman",Font.BOLD,15));
         g.drawString(text,200,200);
   }
}
/*
<applet code = Banner.class width=500 height=500>
</applet>
*/
```

OUTPUT:

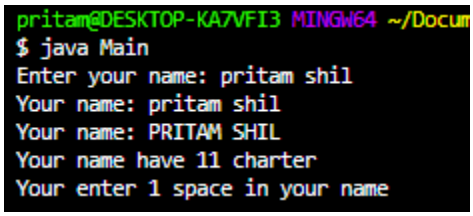

Applet Viewer: Banner.class
Applet
Banner Sample
Applet started.

17. WAP TO IMPLEMENT MINIMUM 3 STRING FUNCTION :

PROGRAM:

```
import java.util.Scanner;
public class Main {
  private static final Scanner scanner = new Scanner(System.in);

  public static String input(String message) {
    System.out.print(message);
    String input = scanner.nextLine();
    return input;
  }

  public static void main(String[] args) {
    String userName = input("Enter your name: ");

    if (userName.isEmpty()) {
      System.out.println("You didn't enter your name");
      return;
    }

    System.out.println("Your name: " + userName.toLowerCase());
    System.out.println("Your name: " + userName.toUpperCase());
    System.out.printf("Your name have %d charter\n", userName.length());
    System.out.printf("Your enter %d space in your name\n", userName.split(" ").length - 1);

    scanner.close();
  }
}
```

OUTPUT:



18. CHANGE THE TEXT TO UPPERCASE OF A FILE :

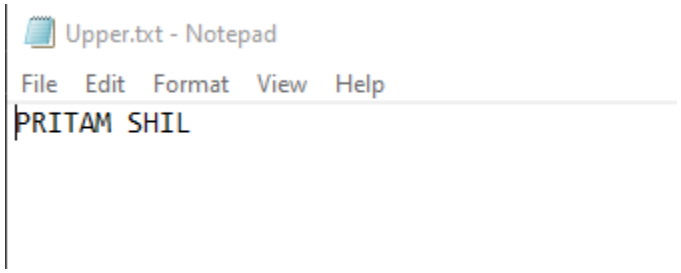
PROGRAM:

```
import java.io.FileWriter;
import java.io.File;
import java.io.IOException;
import java.util.Scanner;
import java.io.FileReader;

public class Upper {
   public static void main(String[] args){
      String txt="";
      File myfile=new File("count.txt");
      Scanner sc= new Scanner(System.in);
      System.out.print("Enter a line you want to put in the file: ");
      String str= sc.nextLine();
      try{
         myfile.createNewFile();
         FileWriter myWriter = new FileWriter("Upper.txt");
         myWriter.write(str);
         myWriter.close();
         FileReader fr=new FileReader("Upper.txt");
         int i;
         while((i=fr.read())!=-1){
            txt+=(char)i;
         }
         fr.close();
         txt=txt.toUpperCase();
         FileWriter myWriter2 = new FileWriter("Upper.txt");
         myWriter2.write(txt);
         myWriter2.close();
      } catch (IOException e) {
         System.out.println("An error occurred.");
         e.printStackTrace();
      }
      for(int i=0;i<5;i++){

      }
   }
}
```

OUTPUT:

```
pritam@DESKTOP-KA7VFI3 MINGW64 ~/Documents/myWork/java/college
$ java Upper
Enter a line you want to put in the file: Pritam shil
```

FILE OUTPUT:



19. DEMONSTRATE INET-ADDRESS URL PROTOCOL:

PROGRAM:

```
import java.net.*;
import java.io.*;
import java.net.InetAddress;

public class Inet {
    public static void main(String[] args) throws Exception {
        InetAddress localhost = InetAddress.getLocalHost();
        System.out.println("My System's IP Address: " + (localhost.getHostAddress()).trim());
    }
}
```

OUTPUT:

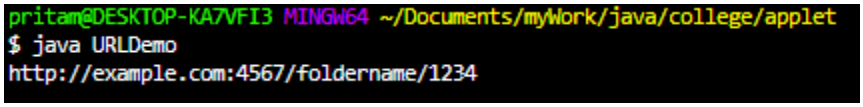
```
pritam@DESKTOP-KA7VFI3 MINGW64 ~/Documents/myWork/java/college/applet
$ java Inet
My System's IP Address: 192.168.42.165
```

20. WAP TO CREATE A URL FROM A PARTICULAR HOST :

PROGRAM:

```
import java.net.*;
import java.net.MalformedURLException;
import java.net.URI;
import java.net.URL;

public class URLDemo{
public static void main(String[] args){
try{
    String protocol = "http";
    String host = "example.com";
    int port = 4567;
    String path = "/foldername/1234";
    String auth = null;
    String fragment = null;
    String query=null;
    URI uri = new URI(protocol, auth, host, port, path, query, fragment);
    URL url = uri.toURL();
    System.out.println(url);
}catch(Exception e){System.out.println(e);}
}
}
```

OUTPUT:

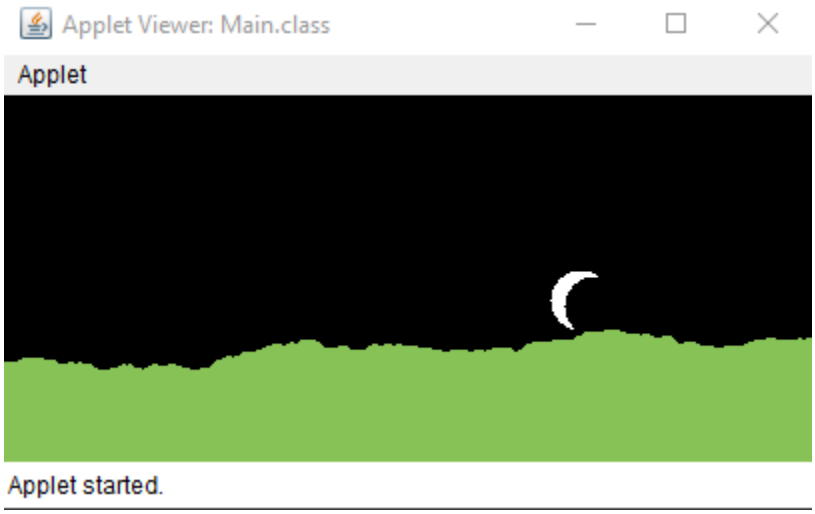


21. APPLET DISPLAY NIGHT VIEW :

PROGRAM:

```
import java.awt.*;
import java.applet.*;

public class Main extends Applet {
  public void init() {
    this.setBackground(Color.BLACK);
  }

  public void paint(Graphics g) {
    {
      int i = 0;
      while (i < 1000) {
        int appletWidth = getSize().width;
        int appletHeight = getSize().height;
        int x = (int) (Math.random() * appletWidth);
        int y = (int) (Math.random() * appletHeight);

        g.setColor(Color.WHITE);
        g.fillOval(x, y, 1, 1);

        i++;
      }
    }

    {
      int appletWidth = getSize().width;
      int appletHeight = getSize().height;
      int x = (int) (0.75 * appletWidth - 30);
      int y = (int) (0.75 * appletHeight - 50);
      g.setColor(Color.WHITE);
      g.fillOval(x, y, 30, 30);
      g.setColor(Color.BLACK);
      g.fillOval(x + 7, y + 3, 30, 30);
    }

    {
      int column = 0;
      int appletWidth = getSize().width;
      int horizonHeight = 50;
      int appletHeight = getSize().height;
      while (column < appletWidth) {
        g.setColor(new Color(.53f, .76f, .34f));
        g.drawRect(column, (appletHeight - horizonHeight), 1, appletHeight);

        column++;
        horizonHeight = horizonHeight + (int) (Math.random() * 3 - 1.5);
      }
    }
  }
}
```

OUTPUT:

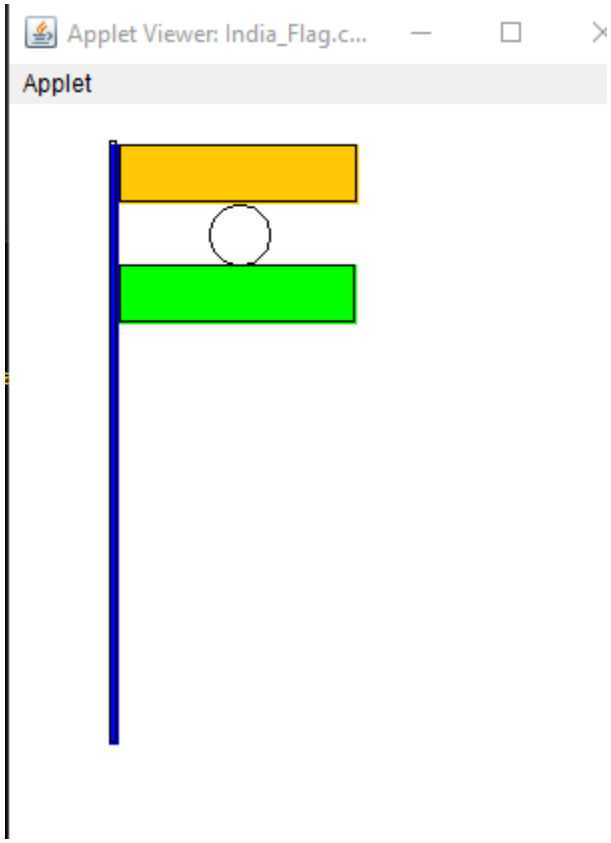


22. CREATE THE INDIAN NATIONAL FLAG :

PROGRAM:

```
import java.awt.*;
import java.applet.*;
public class India_Flag extends Applet
{
        public void paint(Graphics g)
        {
                g.setColor(Color.blue);
                g.fillRect(50,20,5,300);
                g.setColor(Color.black);
                g.drawRect(50,18,3,300);
                g.setColor(Color.orange);
                g.fillRect(55,20,120,30);
                g.setColor(Color.black);
                g.drawRect(55,20,118,28);
                g.setColor(Color.green);
                g.fillRect(55,80,119,30);
                g.setColor(Color.black);
                g.drawRect(55,80,117,28);
                g.setColor(Color.black);
                g.drawOval(100,50,30,30);
        }
}
```

OUTPUT:

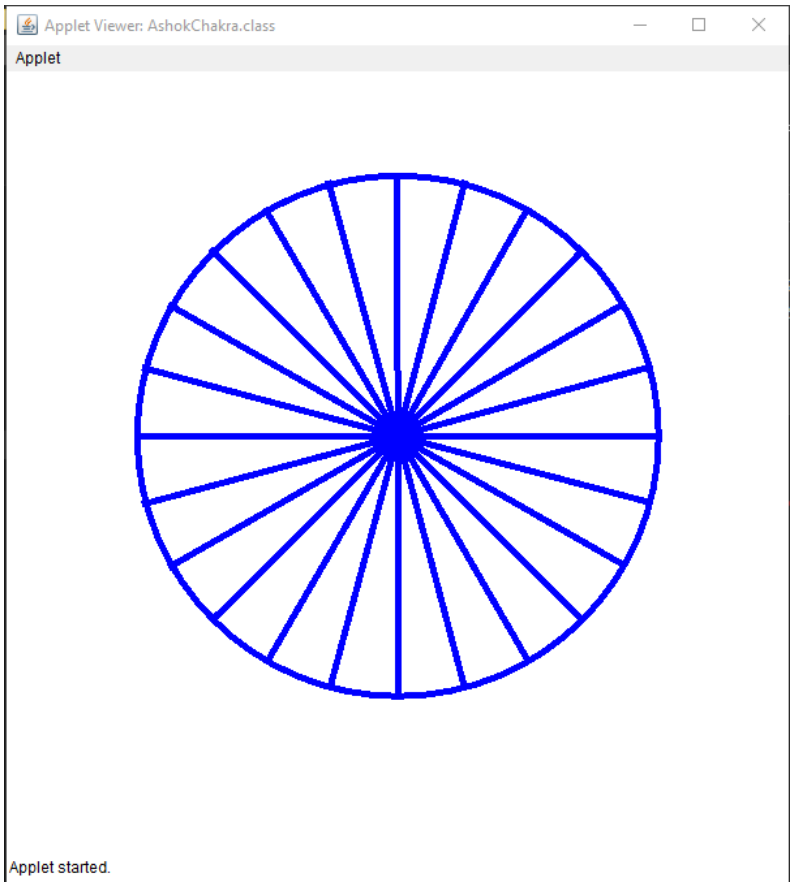


23. CREATE ASHAK CHAKRA :

PROGRAM:

```
import java.applet.*;
import java.awt.*;
import java.awt.geom.Line2D;
import java.awt.geom.Ellipse2D;

public class AshokChakra extends Applet {
  public void drawLine(Graphics graphics, int x1, int y1, int x2, int y2) {
    Graphics2D g2 = (Graphics2D) graphics;
    g2.setColor(Color.blue);
    g2.setStroke(new BasicStroke(5));
    g2.draw(new Line2D.Float(x1, y1, x2, y2));
  }


  public void drawCircle(Graphics graphics, int x1, int y1, int x2, int y2) {
    Graphics2D g2 = (Graphics2D) graphics;
    g2.setColor(Color.blue);
    g2.setStroke(new BasicStroke(5));
    g2.draw(new Ellipse2D.Float(x1, y1, x2, y2));
  }

  public void paint(Graphics graphics) {
    graphics.setColor(Color.blue);
    drawCircle(graphics, 100, 80, 400, 400);

    int n1 = 300;
    int d1 = 280;
    int r = 200;
    double n2, d2, angle, line = 0.0;

    for (int i = 1; i <= 24; i++) {
      angle = line * (3.14 / 180);
      n2 = n1 + (double) r * Math.cos(angle);
      d2 = d1 + (double) r * Math.sin(angle);
      drawLine(graphics, n1, d1, (int) n2, (int) d2);
      line += 360 / (double) 24;
    }
  }
}

/*
 * <applet code="AshokChakra.class" width="600"
 * height="600"></applet>
 */
```

OUTPUT:



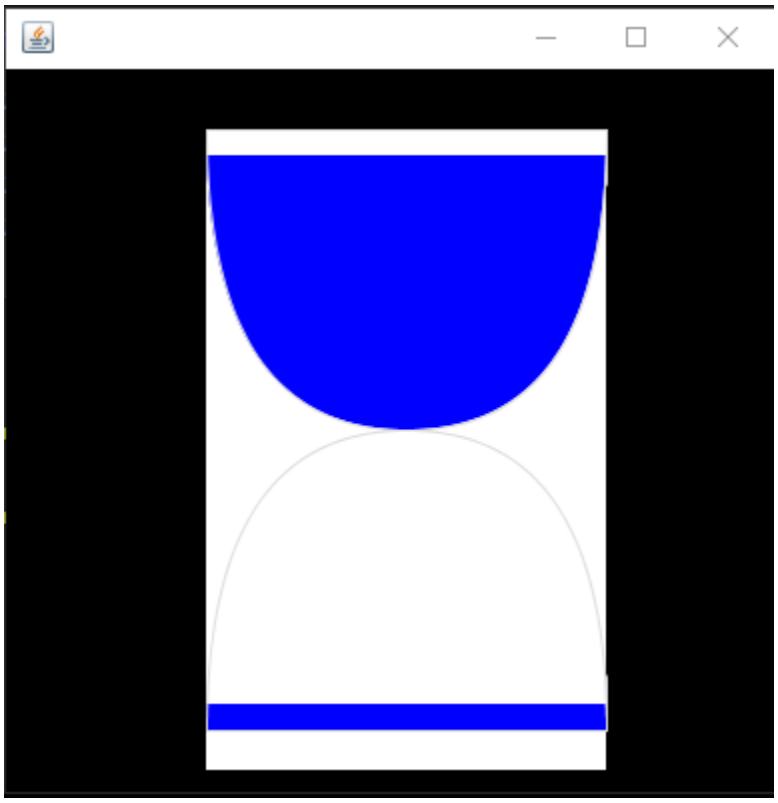
24. CREATE SANDCLOCK :

PROGRAM:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.geom.Area;
import java.awt.geom.GeneralPath;
import java.awt.geom.Rectangle2D;

class Canvas extends JPanel implements ActionListener {
  private int off, elapsed;
  private final int tick;
  private final int h = 150;
  Timer t;

  public Canvas(int hh, int mm) {
    super();
    tick = ((hh * 3600 + mm * 60) * 1000) / h;
    t = new Timer(tick, this);
    t.start();
  }

  @Override
  public void paintComponent(Graphics g) {
    int x = 100, y = 30, w = 200;
    Graphics2D g2 = (Graphics2D) g;
    g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON);

    g2.clearRect(x, y, w, h * 2 + 20);
    g2.setColor(Color.blue);

    GeneralPath polygon = new GeneralPath(GeneralPath.WIND_NON_ZERO, 4);

    polygon.moveTo(x, y);
    polygon.quadTo(x, h + 30, w / 2 + x, h + y);
    polygon.quadTo(x, h + y, x, h * 2 + y);
    polygon.lineTo(w + x, h * 2 + y);
    polygon.quadTo(w + x, h + y, w / 2 + x, h + y);
    polygon.quadTo(w + x, h + y, w + x, y);
    polygon.closePath();


    Rectangle2D mask = new Rectangle2D.Double(x, y + off, w, h - off);
    Rectangle2D mask2 = new Rectangle2D.Double(x, y + 2 * h - off, w, off);

    Area a1 = new Area(mask);
    Area a2 = new Area(polygon);
    a2.intersect(a1);
    g2.fill(a2);
```

```
    a1 = new Area(mask2);
    a2 = new Area(polygon);
    a2.intersect(a1);
    g2.fill(a2);
    g2.setStroke(new BasicStroke(1));
    g2.setColor(new Color(0xDDDDDD));
    g2.draw(polygon);
  }

  @Override
  public void actionPerformed(ActionEvent arg0) {
    off = off + 1;
    elapsed = elapsed + tick;
    repaint();

    if (off == h) {
      t.stop();
    }
  }
}


import java.awt.EventQueue;
import javax.swing.JFrame;
import java.awt.*;
import java.applet.*;

public class Main extends JFrame {
  public static void main(String[] args) {
    new Main();
  }

  public Main() {
    EventQueue.invokeLater(() -> {
      Canvas c = new Canvas(0, 3);
      setContentPane(c);
      pack();
      setSize(400, 400);
      setBackground(Color.BLACK);
      setLocationRelativeTo(null);
      setVisible(true);
    });
  }
}
```

OUTPUT:



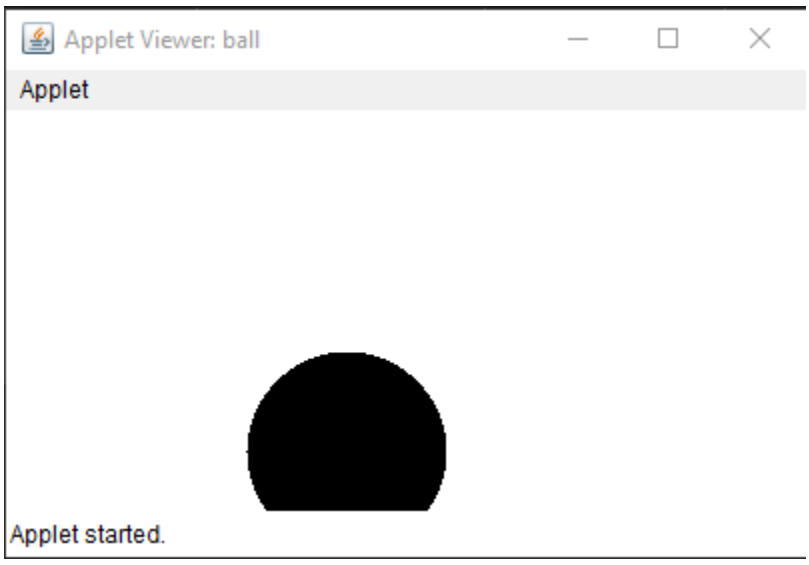
25. CREATE BOUNCEBALL :

PROGRAM:

```
import java.applet.*;
import java.awt.*;

 /* <applet code = "ball" width = 400 height = 200> </applet> */
public class ball extends Applet implements Runnable
 {
 Thread t;
 int x = 0;
 int y = 0;

public void start()
 {
 t = new Thread(this);
 t.start();
 }

public void paint(Graphics g)
  {
    g.fillOval(x,y,100,100);
   }

public void run()
 {
  try
    {
    for(;;)
     {
            for(;;)
        {
            if(y == 120)
               {
                break;
               }
   else if (x == 390)
      {
    x = 0;
    y = 0;
   repaint();
       }
           else
             {
                y = y +3;
                x = x +3;
                Thread.sleep(100);
              repaint();
            }
             }
         for(;;)
           {
            if(y==0)
```

```
                {
                    break;
                    }
               else if (x == 390)
          {
          x = 0;
          y = 0;
               repaint();
    }
    else
               {
        y = y-3;
              x = x +3;
              Thread.sleep(100);
               repaint();
              }
               }
            run();
        }
  }
catch(InterruptedException e)
    {
    }
 }
}
```

OUTPUT:



26. CREATE MULTIPLE BOUNCEBALL:

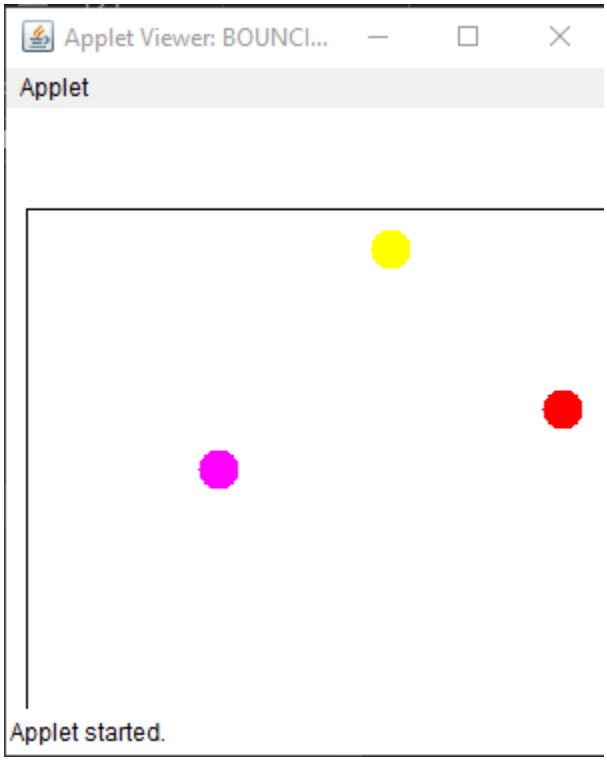
PROGRAM:

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
public class BOUNCINGBALLS extends Applet implements MouseListener, Runnable
{
   Thread t=null;
   int x1=10, x2=10, x3=10, x4=10;
   int y1=300, y2=300, y3=300, y4=300;
 int flagx1,flagy1,flagx2,flagy2;
 int flagx3,flagy3,flagx4,flagy4;

   public void init()
    {
        addMouseListener(this);
    }
   public void mouseExited(MouseEvent me)   {}
   public void mouseReleased(MouseEvent me) {}
   public void mouseEntered(MouseEvent me)  {}
   public void mousePressed(MouseEvent me)  {}
   public void mouseClicked(MouseEvent me)  {}

   public void start()
   {
      t=new Thread(this);
      t.start();
   }

   public void run()
   {
      for(;;)
      {
         try
         {
            repaint();
   if(y1<=50)
   flagx1=0;
   else if(y1>=300)
   flagx1=1;
   if(x1<=10)
   flagy1=0;
   else if(x1>=400)
   flagy1=1;
   if(y2<=50)
   flagx2=0;
   else if(y2>=300)
   flagx2=1;
   if(x2<=10)
   flagy2=0;
   else if(x2>=400)
   flagy2=1;
```

```
  if(y3<=50)
  flagx3=0;
  else if(y3>=300)
  flagx3=1;
  if(x3<=10)
  flagy3=0;
  else if(x3>=400)
  flagy3=1;
  if(y4<=50)
  flagx4=0;
  else if(y4>=300)
  flagx4=1;
  if(x4<=10)
  flagy4=0;
  else if(x4>=400)
  flagy4=1;
            Thread.sleep(10);
         }catch(InterruptedException e){}
      }
  }
  public void paint(Graphics g)
  {
g.drawRect(10,50,410,270);

      g.setColor(Color.blue);
      g.fillOval(x1,y1,20,20);
if(flagx1==1)
 y1-=2;
else if(flagx1==0)
 y1+=2;
if(flagy1==0)
 x1+=4;
else if(flagy1==1)
 x1-=4;
      g.setColor(Color.red);
      g.fillOval(x2,y2,20,20);
if(flagx2==1)
 y2-=4;
else if(flagx2==0)
 y2+=4;
if(flagy2==0)
 x2+=3;
else if(flagy2==1)
 x2-=3;
      g.setColor(Color.yellow);
      g.fillOval(x3,y3,20,20);
if(flagx3==1)
 y3-=6;
else if(flagx3==0)
 y3+=6;
if(flagy3==0)
 x3+=2;
else if(flagy3==1)
 x3-=2;
g.setColor(Color.magenta);
      g.fillOval(x4,y4,20,20);
```

```
 if(flagx4==1)
  y4-=5;
 else if(flagx4==0)
  y4+=5;
 if(flagy4==0)
  x4+=1;
 else if(flagy4==1)
  x4-=1;
   }
}
```

OUTPUT:

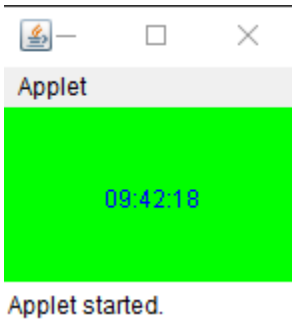


27. CREATE DIGITALCLOCK :

PROGRAM:

```
import java.applet.*;
import java.awt.*;
import java.util.*;
import java.text.*;

public class DigitalClock extends Applet implements Runnable {

  Thread t = null;
  int hours=0, minutes=0, seconds=0;
  String timeString = "";

  public void init() {
    setBackground( Color.green);
  }

  public void start() {
      t = new Thread( this );
      t.start();
  }


  public void run() {
    try {
      while (true) {

        Calendar cal = Calendar.getInstance();
        hours = cal.get( Calendar.HOUR_OF_DAY );
        if ( hours > 12 ) hours -= 12;
        minutes = cal.get( Calendar.MINUTE );
        seconds = cal.get( Calendar.SECOND );

        SimpleDateFormat formatter = new SimpleDateFormat("hh:mm:ss");
        Date date = cal.getTime();
        timeString = formatter.format( date );

        repaint();
        t.sleep( 1000 );
      }
    }
    catch (Exception e) { }
  }


 public void paint( Graphics g ) {
    g.setColor( Color.blue );
    g.drawString( timeString, 50, 50 );
  }
}
```

OUTPUT:



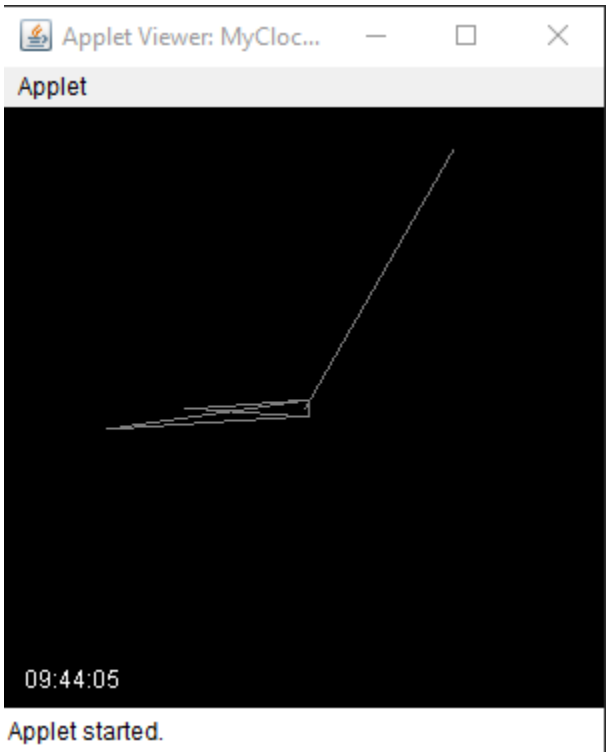
28. CREATE ANALOG CLOCK :

PROGRAM:

```
import java.applet.*;
import java.awt.*;
import java.util.*;
import java.text.*;

public class MyClock extends Applet implements Runnable {

  int width, height;
  Thread t = null;
  boolean threadSuspended;
  int hours=0, minutes=0, seconds=0;
  String timeString = "";

  public void init() {
    width = getSize().width;
    height = getSize().height;
    setBackground( Color.black );
  }

  public void start() {
    if ( t == null ) {
      t = new Thread( this );
      t.setPriority( Thread.MIN_PRIORITY );
      threadSuspended = false;
      t.start();
    }
    else {
      if ( threadSuspended ) {
        threadSuspended = false;
        synchronized( this ) {
          notify();
        }
      }
    }
  }

  public void stop() {
    threadSuspended = true;
  }

  public void run() {
    try {
      while (true) {

        Calendar cal = Calendar.getInstance();
        hours = cal.get( Calendar.HOUR_OF_DAY );
        if ( hours > 12 ) hours -= 12;
        minutes = cal.get( Calendar.MINUTE );
        seconds = cal.get( Calendar.SECOND );
```

```
            SimpleDateFormat formatter
              = new SimpleDateFormat( "hh:mm:ss", Locale.getDefault() );
            Date date = cal.getTime();
            timeString = formatter.format( date );

            if ( threadSuspended ) {
              synchronized( this ) {
                while ( threadSuspended ) {
                  wait();
                }
              }
            }
            repaint();
            t.sleep( 1000 );
          }
        }
        catch (Exception e) { }
    }

    void drawHand( double angle, int radius, Graphics g ) {
        angle -= 0.5 * Math.PI;
        int x = (int)( radius*Math.cos(angle) );
        int y = (int)( radius*Math.sin(angle) );
        g.drawLine( width/2, height/2, width/2 + x, height/2 + y );
    }

    void drawWedge( double angle, int radius, Graphics g ) {
        angle -= 0.5 * Math.PI;
        int x = (int)( radius*Math.cos(angle) );
        int y = (int)( radius*Math.sin(angle) );
        angle += 2*Math.PI/3;
        int x2 = (int)( 5*Math.cos(angle) );
        int y2 = (int)( 5*Math.sin(angle) );
        angle += 2*Math.PI/3;
        int x3 = (int)( 5*Math.cos(angle) );
        int y3 = (int)( 5*Math.sin(angle) );
        g.drawLine( width/2+x2, height/2+y2, width/2 + x, height/2 + y );
        g.drawLine( width/2+x3, height/2+y3, width/2 + x, height/2 + y );
        g.drawLine( width/2+x2, height/2+y2, width/2 + x3, height/2 + y3 );
    }

    public void paint( Graphics g ) {
        g.setColor( Color.gray );
        drawWedge( 2*Math.PI * hours / 12, width/5, g );
        drawWedge( 2*Math.PI * minutes / 60, width/3, g );
        drawHand( 2*Math.PI * seconds / 60, width/2, g );
        g.setColor( Color.white );
        g.drawString( timeString, 10, height-10 );
    }
}
```

OUTPUT:

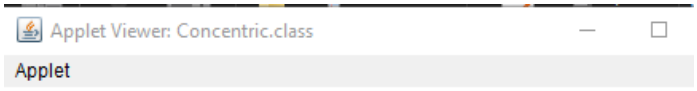


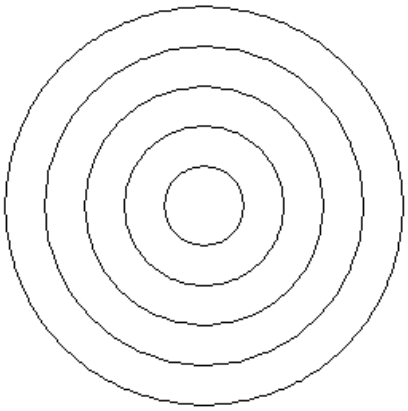
29. CREATE A CONCENTRIC CIRCLE :

PROGRAM:

```
import java.applet.*;
import java.awt.*;
public class Concentric extends Applet
{
   public void init()
   {
         setBackground(Color.blue);
   }
   public void paint(Graphics g)
   {
         g.setColor(Color.black);
         int rad=25;
         int dia=50;
         for(int i=0;i<5;i++)
         {
            g.drawOval(250-(i*rad),250-(i*rad),(i+1)*dia,(i+1)*dia);
         }
   }
}
/*
<applet code = Concentric.class width=500 height=500>
</applet>
*/
```

OUTPUT:



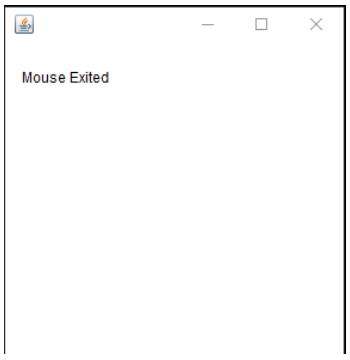


30. MOUSE EVENTS :

PROGRAM:

```
import java.awt.*;
import java.awt.event.*;
public class MouseListenerExample extends Frame implements MouseListener{
    Label l;
    MouseListenerExample(){
        addMouseListener(this);

        l=new Label();
        l.setBounds(20,50,100,20);
        add(l);
        setSize(300,300);
        setLayout(null);
        setVisible(true);
    }
    public void mouseClicked(MouseEvent e) {
        l.setText("Mouse Clicked");
    }
    public void mouseEntered(MouseEvent e) {
        l.setText("Mouse Entered");
    }
    public void mouseExited(MouseEvent e) {
        l.setText("Mouse Exited");
    }
    public void mousePressed(MouseEvent e) {
        l.setText("Mouse Pressed");
    }
    public void mouseReleased(MouseEvent e) {
        l.setText("Mouse Released");
    }
public static void main(String[] args) {
    new MouseListenerExample();
}
}
```

OUTPUT:

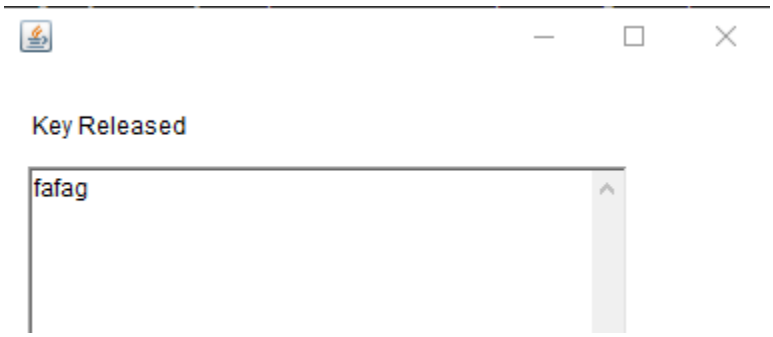


31. KEY EVENTS :

PROGRAM:

```
import java.awt.*;
import java.awt.event.*;
public class KeyListenerExample extends Frame implements KeyListener {
 Label l;
    TextArea area;
    KeyListenerExample() {
        l = new Label();
        l.setBounds (20, 50, 100, 20);
        area = new TextArea();
        area.setBounds (20, 80, 300, 300);
        area.addKeyListener(this);
        add(l);
add(area);
        setSize (400, 400);
        setLayout (null);
        setVisible (true);
    }
    public void keyPressed (KeyEvent e) {
        l.setText ("Key Pressed");
    }
    public void keyReleased (KeyEvent e) {
        l.setText ("Key Released");
    }
    public void keyTyped (KeyEvent e) {
        l.setText ("Key Typed");
    }
    public static void main(String[] args) {
        new KeyListenerExample();
    }
}
```

OUTPUT:

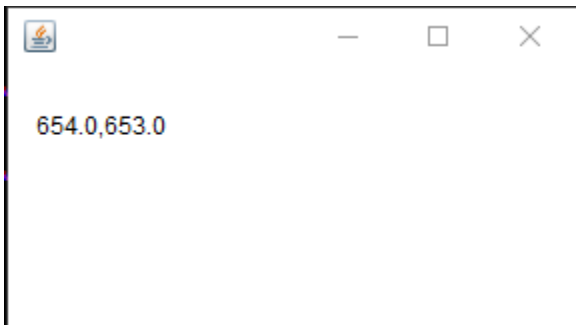


32. MOUSE POSITION :

PROGRAM:

```
import java.awt.*;
import java.awt.event.*;
public class Mousepos extends Frame implements MouseListener{
   Label l;
   Mousepos(){
      addMouseListener(this);

      l=new Label();
      l.setBounds(20,50,100,20);
      add(l);
      setSize(300,300);
      setLayout(null);
      setVisible(true);
   }
   public void mouseClicked(MouseEvent e) {

   }
   public void mouseEntered(MouseEvent e) {

   }
   public void mouseExited(MouseEvent e) {
      Point point = MouseInfo.getPointerInfo().getLocation();
      l.setText(point.getX()+","+point.getY());
   }
   public void mousePressed(MouseEvent e) {

   }
   public void mouseReleased(MouseEvent e) {

   }
public static void main(String[] args) {
   new Mousepos();
}
}
```

OUTPUT:

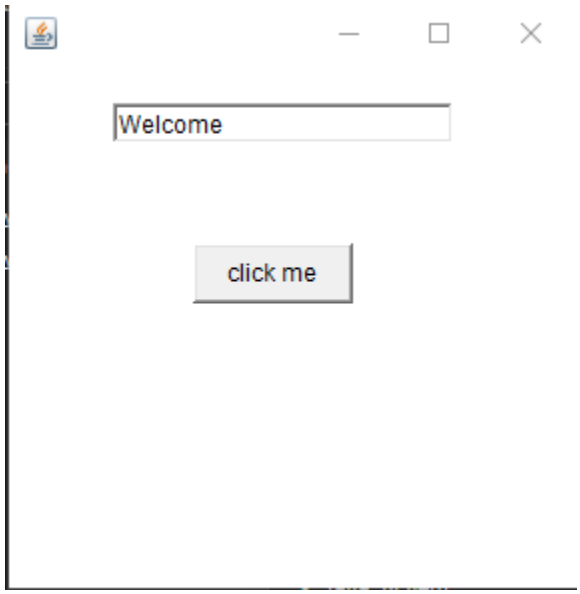


33. MESSAGE AFTER CLICK :

PROGRAM:

```
import java.awt.*;
import java.awt.event.*;
class AEvent extends Frame implements ActionListener{
        TextField tf;
        AEvent(){
                tf=new TextField();
                tf.setBounds(60,50,170,20);
                Button b=new Button("click me");
                b.setBounds(100,120,80,30);

                b.addActionListener(this);

                add(b);add(tf);
                setSize(300,300);
                setLayout(null);
                setVisible(true);
        }
        public void actionPerformed(ActionEvent e){
                tf.setText("Welcome");
        }
        public static void main(String args[]){
                new AEvent();
        }
}
```

OUTPUT:

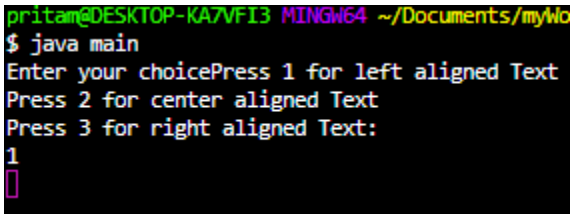


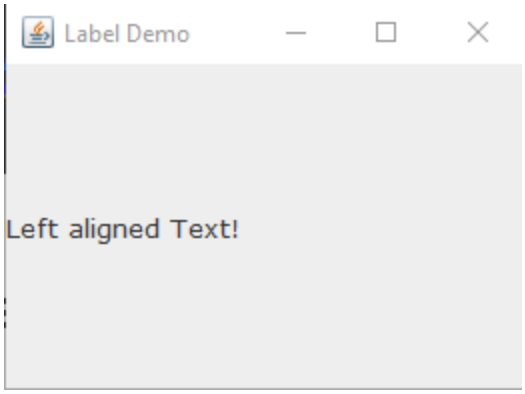
34. APPLET TEXT ALIGNMENT:

PROGRAM:

```
import java.util.*;
import java.awt.Font;
import javax.swing.*;
public class main {
  public static void main(String args[]) {
    Scanner sc= new Scanner(System.in);
    System.out.println("Enter your choice");
    System.out.println("Press 1 for left aligned Text");
    System.out.println("Press 2 for center aligned Text");
    System.out.println("Press 3 for right aligned Text:");
    int a= sc.nextInt();
    JFrame frame = new JFrame("Label Demo");
    JLabel label1;
    label1 = new JLabel("Left aligned Text!", JLabel.LEFT);
    label1.setFont(new Font("Verdana", Font.PLAIN, 13));
    JLabel label2;
    label2 = new JLabel("Center aligned!", JLabel.CENTER);
    label2.setFont(new Font("Verdana", Font.PLAIN, 13));
    JLabel label3;
    label3 = new JLabel("Right aligned!", JLabel.RIGHT);
    label3.setFont(new Font("Verdana", Font.PLAIN, 13));
    frame.add(label3);
    if(a==1){
      frame.add(label1);
    }
    else if(a==2){
      frame.add(label2);
    }
    else if(a==3){
      frame.add(label3);
    }
    frame.setSize(500,300);
    frame.setVisible(true);
  }
}
```

OUTPUT:



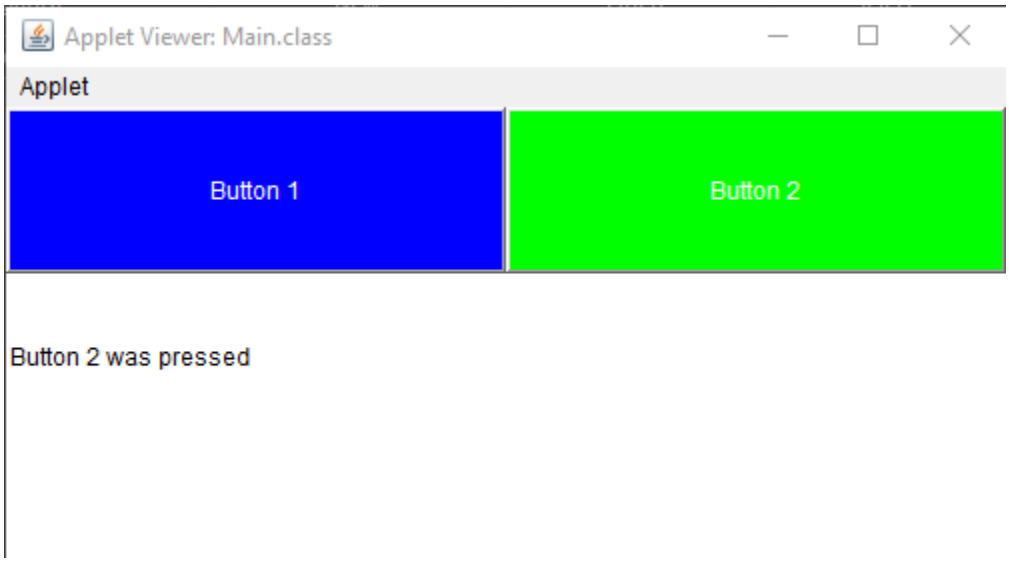

Label Demo
Left aligned Text!

35. DIFFERENT BUTTONS :

PROGRAM:

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;

public class Main extends Applet implements ActionListener {
  Button button1, button2;
  Label status;

  public void init() {
    setLayout(new GridLayout(3, 2, 1, 1));
    Panel p1 = new Panel();
    p1.setLayout(new GridLayout(2, 1));

    button1 = new Button("Button 1");
    button1.setBackground(Color.BLUE);
    button1.setForeground(Color.WHITE);
    button1.addActionListener(this);

    button2 = new Button("Button 2");
    button2.setBackground(Color.GREEN);
    button2.setForeground(Color.WHITE);
    button2.addActionListener(this);

    p1.add(button1);
    p1.add(button2);
    status = new Label("");

    p1.add(status);
    add(p1);
  }

  public void actionPerformed(ActionEvent e) {
    if (e.getSource() == button1) {
      status.setText("Button 1 was pressed");
      System.out.println("Button 1 was pressed");
    } else {
      status.setText("Button 2 was pressed");
      System.out.println("Button 2 was pressed");
    }
  }
}

// <applet code="Main.class" width="500" height="500"></applet>
```

OUTPUT:


Applet Viewer: Main.class
Applet
Button 1
Button 2
Button 2 was pressed

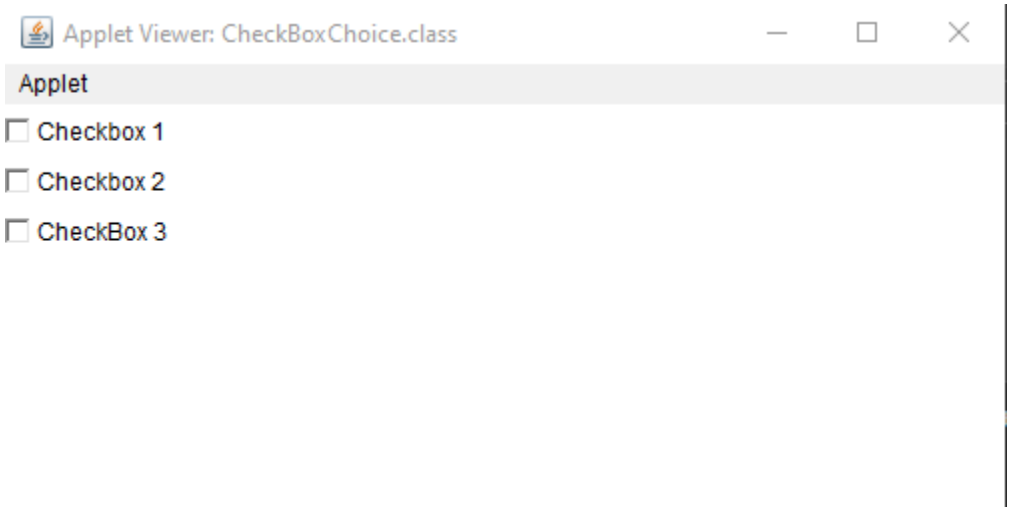
36. CHECKBOX USING APPLETS :

PROGRAM:

```
import java.awt.*;
import java.awt.event.*;
import java.applet.Applet;

public class CheckBoxChoice extends Applet implements ItemListener {
  Checkbox c1, c2, c3;
  Label l;

  public void init() {
    setLayout(new GridLayout(3, 2, 1, 1));
    Panel p1 = new Panel();
    Panel p2 = new Panel();
    p1.setLayout(new GridLayout(3, 1));
    p2.setLayout(new GridLayout(1, 2));

    c1 = new Checkbox("Checkbox 1");
    p1.add(c1);
    c1.addItemListener(this);
    c2 = new Checkbox("Checkbox 2");
    p1.add(c2);
    c2.addItemListener(this);
    c3 = new Checkbox("CheckBox 3");
    p1.add(c3);
    c3.addItemListener(this);
    l = new Label("");
    p2.add(l);

    add(p1);
    add(p2);
  }

  public void itemStateChanged(ItemEvent e) {
    c1.setForeground(Color.BLACK);
    c2.setForeground(Color.BLACK);
    c3.setForeground(Color.BLACK);

    String selected = "";
    if (c1.getState()) {
      selected += "1";
      c1.setForeground(Color.BLUE);
    }
    if (c2.getState()) {
      if (!selected.isEmpty())
        selected += ", ";
      selected += "2";
      c2.setForeground(Color.BLUE);
    }
    if (c3.getState()) {
      if (!selected.isEmpty())
        selected += ", ";
```

```
      selected += "3";
      c3.setForeground(Color.BLUE);
    }
    selected += " Checkbox selected";
    l.setText(selected);
  }
}

// <applet code="CheckBoxChoice.class" width="500"
// height="500"></applet>
```

OUTPUT:



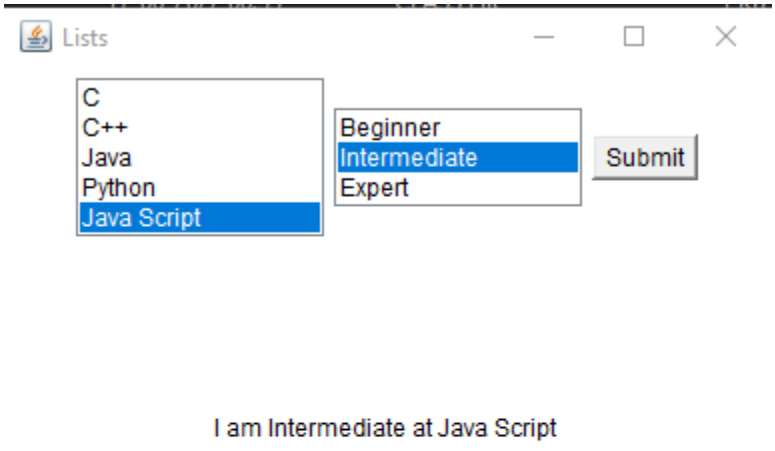
37. CHOICE LIST (CLICK A LIST TO GET THE DETAILS) :

PROGRAM:

```
import java.awt.*;
import java.awt.event.*;

public class Lists {
  private final Frame mainFrame;
  private final Label status;
  private final Panel controlPanel;

  public Lists() {
    mainFrame = new Frame("Lists");
    mainFrame.setSize(400, 400);
    mainFrame.setLayout(new GridLayout(3, 1));
    mainFrame.addWindowListener(new WindowAdapter() {
      public void windowClosing(WindowEvent windowEvent) {
        System.exit(0);
      }
    });
    status = new Label();
    status.setAlignment(Label.CENTER);
    status.setSize(350, 100);
    status.setText("My expertise");

    controlPanel = new Panel();
    controlPanel.setLayout(new FlowLayout());

    mainFrame.add(controlPanel);
    mainFrame.add(status);
    mainFrame.setVisible(true);
  }

  public static void main(String[] args) {
    Lists list = new Lists();
    list.showLists();
  }



  private void showLists() {
    final List languages = new List(5, true);

    languages.add("C");
    languages.add("C++");
    languages.add("Java");
    languages.add("Python");
    languages.add("Java Script");

    final List expertise = new List(3, false);

    expertise.add("Beginner");
    expertise.add("Intermediate");
```

```
    expertise.add("Expert");

    Button showButton = new Button("Submit");

    showButton.addActionListener(new ActionListener() {

      public void actionPerformed(ActionEvent e) {
        StringBuilder data = new StringBuilder("I am " +
expertise.getItem(expertise.getSelectedIndex()) + " at ");
        for (String lang : languages.getSelectedItems()) {
          data.append(lang).append(" ");
        }
        status.setText(data.toString());
      }
    });

    controlPanel.add(languages);
    controlPanel.add(expertise);
    controlPanel.add(showButton);

    mainFrame.setVisible(true);
  }
}
```

OUTPUT:

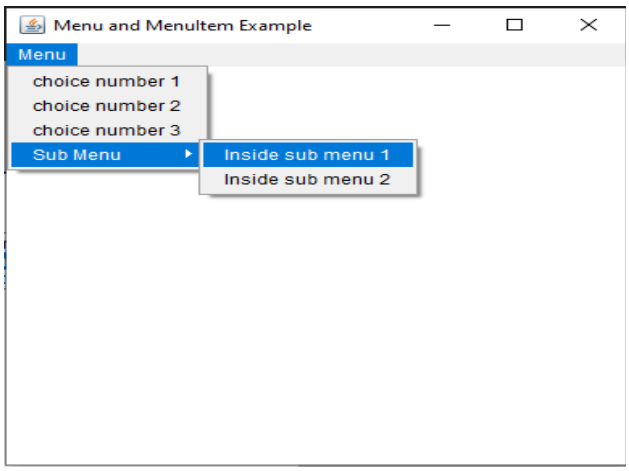


38. MENUBAR (DROPDOWN) :

PROGRAM:

```
import java.awt.*;
class main
{
    main(){
        Frame f= new Frame("Menu and MenuItem Example");
        MenuBar mb=new MenuBar();
        Menu menu=new Menu("Menu");
        Menu submenu=new Menu("Sub Menu");
        MenuItem i1=new MenuItem("choice number 1");
        MenuItem i2=new MenuItem("choice number 2");
        MenuItem i3=new MenuItem("choice number 3");
        MenuItem i4=new MenuItem("Inside sub menu 1");
        MenuItem i5=new MenuItem("Inside sub menu 2");
        menu.add(i1);
        menu.add(i2);
        menu.add(i3);
        submenu.add(i4);
        submenu.add(i5);
        menu.add(submenu);
        mb.add(menu);
        f.setMenuBar(mb);
        f.setSize(400,400);
        f.setLayout(null);
        f.setVisible(true);
}
public static void main(String args[])
{
new main();
}
}
```

OUTPUT:



39. IMPORT AN IMAGE :

PROGRAM:

```
import java.awt.*;
import java.applet.*;
public class main extends Applet {

  Image picture;

  public void init() {
    picture = getImage(getDocumentBase(),"sonoo.png");
  }

  public void paint(Graphics g) {
    g.drawImage(picture, 30,30, this);
  }

  }

/*
<applet code = main.class width=500 height=500>
</applet>
*/
```

OUTPUT:

40. DIALOG BOX :

PROGRAM:

```
import java.awt.*;
import java.awt.event.*;
public class DialogExample {
    private static Dialog d;
    DialogExample() {
        Frame f= new Frame();
        d = new Dialog(f , "Dialog Example", true);
        d.setLayout( new FlowLayout() );
        Button b = new Button ("OK");
        b.addActionListener ( new ActionListener()
        {
            public void actionPerformed( ActionEvent e )
            {
                DialogExample.d.setVisible(false);
            }
        });
        d.add( new Label ("This is a simple dialog box."));
        d.add(b);
        d.setSize(300,300);
        d.setVisible(true);
    }
    public static void main(String args[])
    {
        new DialogExample();
    }
}
```

OUTPUT: