5-CARDS INTERFACE

PROGRAM-:

```
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import javax.imageio.*;
import javax.swing.*;
import java.util.*;

class ImageComponent extends JComponent
{
        Image i1,i2,i3,i4,i5,i6,i7,i8,i9,i10,i11,i12,i13;
        ArrayList al=new ArrayList();
        int[] player=new int[5];
        int[] opponent=new int[5];
        int[] deck=new int[42];
        int opencard,joker,playervalue,opponentvalue,opptotal=0,pltotal=0,round=0,reshuffle;
        String s1,s2,s3,s4,s5;
        int xposition,yposition,counter=0;
        int[] values=new int[52];
        String moves="";
        String[]
namestemp={"Aclubs","2clubs","3clubs","4clubs","5clubs","6clubs","7clubs","8clubs","9clubs","
10clubs","Jclubs","Qclubs","Kclubs",
"Adice","2dice","3dice","4dice","5dice","6dice","7dice","8dice","9dice","10dice","Jdice","Qdice",
"Kdice",
"Ahearts","2hearts","3hearts","4hearts","5hearts","6hearts","7hearts","8hearts","9hearts","10h
earts","Jhearts","Qhearts","Khearts",
"Aspade","2spade","3spade","4spade","5spade","6spade","7spade","8spade","9spade","10spad
e","Jspade","Qspade","Kspade",};

        String[]
names={"Aclubs","2clubs","3clubs","4clubs","5clubs","6clubs","7clubs","8clubs","9clubs","10clu
bs","Jclubs","Qclubs","Kclubs",
"Adice","2dice","3dice","4dice","5dice","6dice","7dice","8dice","9dice","10dice","Jdice","Qdice",
"Kdice",
"Ahearts","2hearts","3hearts","4hearts","5hearts","6hearts","7hearts","8hearts","9hearts","10h
earts","Jhearts","Qhearts","Khearts",
"Aspade","2spade","3spade","4spade","5spade","6spade","7spade","8spade","9spade","10spad
e","Jspade","Qspade","Kspade",};
        String[] shuffle=new String[52];
        protected void paintComponent(Graphics g)
        {
```

```
        g.drawImage(i1,xposition,yposition,null);
        g.drawImage(i1,xposition+70,yposition,null);
        g.drawImage(i1,xposition+140,yposition,null);
        g.drawImage(i1,xposition+210,yposition,null);
        g.drawImage(i1,xposition+280,yposition,null);
        g.drawImage(i1,xposition+350,yposition,null);
        g.drawImage(i1,xposition+420,yposition,null);
        g.drawImage(i1,xposition+490,yposition,null);
        g.drawImage(i1,xposition,yposition+30,null);
        g.drawImage(i1,xposition+70,yposition+30,null);
        g.drawImage(i1,xposition+140,yposition+30,null);
        g.drawImage(i1,xposition+210,yposition+30,null);
        g.drawImage(i1,xposition+280,yposition+30,null);
        g.drawImage(i1,xposition+350,yposition+30,null);
        g.drawImage(i1,xposition+420,yposition+30,null);
        g.drawImage(i1,xposition+490,yposition+30,null);
        g.drawImage(i9,xposition+10,yposition,null);
        g.drawImage(i10,xposition+85,yposition,null);
        g.drawImage(i11,xposition+160,yposition,null);
        g.drawImage(i12,xposition+235,yposition,null);
        g.drawImage(i13,xposition+310,yposition,null);
        g.drawImage(i2,xposition+420,yposition,null);
}
String playerscore()
{
        return "Your Score :  "+Integer.toString(playervalue);
}
void addup()
{
        counter++;
        moves+="\n\nMove "+Integer.toString(counter)+"\n=======\n\n";
        moves+="Opponent  :  \n";
        for(int i=0;i<5;i++)
        moves+=(names[opponent[i]])+" "+Integer.toString(values[i+5])+"\n";
        moves+="\nOpponentValue = "+Integer.toString(opponentvalue);
        moves+="\n\nPlayer :\n";
        for(int i=0;i<5;i++)
        moves+=(names[player[i]])+" "+Integer.toString(values[i])+"\n";
        moves+="\nPlayerValue = "+Integer.toString(playervalue);
        moves+="\n\n";
}
boolean opponentbid()
{
        int i;
        i=(int)(Math.random()*100);
        if(opponentvalue==0)
```

```
                {
                        JOptionPane.showMessageDialog(this,"I'm Bidding\nMy Score  :
"+opponentvalue);
                        checkwinner();
                        return true;
                }
                if(opponentvalue<=10)
                {
                        if(i%2==0)
                        {
                                JOptionPane.showMessageDialog(this,"I'm Bidding\nMy Score  :
"+opponentvalue);
                                checkwinner();
                                return true;
                        }
                }
                return false;
        }
        int opponentplay()
        {
                try{

                int i,max,pos=0,count,ct,temp,count2;
                for(i=0;i<5;i++)
                {
                        if(names[opponent[i]].equals("cardback"))
                                values[i+5]=0;
                        else if(names[opponent[i]].substring(0,1).equals("A"))
                                values[i+5]=1;
                        else if(names[opponent[i]].substring(0,1).equals("2"))
                                values[i+5]=2;
                        else if(names[opponent[i]].substring(0,1).equals("3"))
                                values[i+5]=3;
                        else if(names[opponent[i]].substring(0,1).equals("4"))
                                values[i+5]=4;
                        else if(names[opponent[i]].substring(0,1).equals("5"))
                                values[i+5]=5;
                        else if(names[opponent[i]].substring(0,1).equals("6"))
                                values[i+5]=6;
                        else if(names[opponent[i]].substring(0,1).equals("7"))
                                values[i+5]=7;
                        else if(names[opponent[i]].substring(0,1).equals("8"))
                                values[i+5]=8;
                        else if(names[opponent[i]].substring(0,1).equals("9"))
                                values[i+5]=9;
                        else if(names[opponent[i]].substring(0,1).equals("1"))
```

```
                values[i+5]=10;
        else if(names[opponent[i]].substring(0,1).equals("J"))
                values[i+5]=10;
        else if(names[opponent[i]].substring(0,1).equals("Q"))
                values[i+5]=10;
        else if(names[opponent[i]].substring(0,1).equals("K"))
                values[i+5]=10;
}
for(i=0;i<5;i++)
{
        if(names[joker].substring(0,1).equals(names[opponent[i]].substring(0,1)))
                values[i+5]=0;
}
count=0;
boolean[] select=new boolean[5];
select[0]=false;
select[1]=false;
select[2]=false;
select[3]=false;
select[4]=false;
max=0;
ct=0;
for(i=5;i<10;i++)
{
        if(values[i]>=max)
        {
                max=values[i];
                pos=i-5;
        }
}
for(i=5;i<10;i++)
{
        if(values[i]==max)
                ct++;
}
if(ct>0)
for(i=5;i<10;i++)
{
        if(values[i]==max)
        {
                select[i-5]=true;
        }
}
if(max==10)
{
        count=0;
```

```
for(i=0;i<5;i++)
if(names[opponent[i]].substring(0,1).equals("1"))
{
        if(!names[joker].substring(0,1).equals("1"))
        {
        for(int j=0;j<5;j++)
        {
                if(names[opponent[j]].substring(0,1).equals("1"))
                {}
                else select[j]=false;
        }
        count++;
        }
}
for(i=0;i<5&&count==0;i++)
if(names[opponent[i]].substring(0,1).equals("J"))
{
        if(!names[joker].substring(0,1).equals("J"))
        {
        for(int j=0;j<5;j++)
        {
                if(names[opponent[j]].substring(0,1).equals("J"))
                {}
                else select[j]=false;
        }
        count++;
        }
}
for(i=0;i<5&&count==0;i++)
if(names[opponent[i]].substring(0,1).equals("K"))
{
        if(!names[joker].substring(0,1).equals("K"))
        {
        for(int j=0;j<5;j++)
        {
                if(names[opponent[j]].substring(0,1).equals("K"))
                {}
                else select[j]=false;
        }
        count++;}
}
for(i=0;i<5&&count==0;i++)
if(names[opponent[i]].substring(0,1).equals("Q"))
{
        if(!names[joker].substring(0,1).equals("Q"))
        {
```

```
                for(int j=0;j<5;j++)
                {
                        if(names[opponent[j]].substring(0,1).equals("Q"))
                        {}
                        else select[j]=false;
                }
                count++;}
        }
}
String jok=names[joker].substring(0,1);
for(i=5;i<10;i++)
{
        values[i]=opponent[i-5]%13;
        if(values[i]>9)
                values[i]=9;
        values[i]++;
}
for(i=5;i<10;i++)
{
        if(names[opponent[i-5]].substring(0,1).equals(jok))
                values[i]=0;
        if(names[opponent[i-5]].substring(0,1).equals("c"))
                values[i]=0;
}
if(values[10]>max || values[10]==10)
{
        count=0;
        if(ct==0)
        {
                if(select[0]==true)
                {
                        temp=opponent[0];
                        opponent[0]=deck[0];
                        for(i=0;i<39;i++)
                                deck[i]=deck[i+1];
                        deck[39]=opencard;
                        opencard=temp;
                }
                if(select[1]==true)
                {
                        temp=opponent[1];
                        opponent[1]=deck[0];
                        for(i=0;i<39;i++)
                                deck[i]=deck[i+1];
                        deck[39]=opencard;
                        opencard=temp;
```

```
		}
		if(select[2]==true)
		{
			temp=opponent[2];
			opponent[2]=deck[0];
			for(i=0;i<39;i++)
				deck[i]=deck[i+1];
			deck[39]=opencard;
			opencard=temp;
		}
		if(select[3]==true)
		{
			temp=opponent[3];
			opponent[3]=deck[0];
			for(i=0;i<39;i++)
				deck[i]=deck[i+1];
			deck[39]=opencard;
			opencard=temp;
		}
		if(select[4]==true)
		{
			temp=opponent[4];
			opponent[4]=deck[0];
			for(i=0;i<39;i++)
				deck[i]=deck[i+1];
			deck[39]=opencard;
			opencard=temp;
		}
	}
	else if(ct>=1)
	{
		count2=0;
		if(select[0]==true)
		{
			temp=opponent[0];
			opponent[0]=deck[0];
			for(i=0;i<39;i++)
				deck[i]=deck[i+1];
			deck[39]=opencard;
			opencard=temp;
			count2++;
		}
		if(select[1]==true)
		{
			temp=opponent[1];
			opponent[1]=deck[0];
```

```
		for(i=0;i<39;i++)
			deck[i]=deck[i+1];
		deck[39]=opencard;
		opencard=temp;
		if(count2==0)
			count2++;
		else
			names[opponent[1]]="cardback";
	}
	if(select[2]==true)
	{
		temp=opponent[2];
		opponent[2]=deck[0];
		for(i=0;i<39;i++)
			deck[i]=deck[i+1];
		deck[39]=opencard;
		opencard=temp;
		if(count2==0)
			count2++;
		else
			names[opponent[2]]="cardback";
	}
	if(select[3]==true)
	{
		temp=opponent[3];
		opponent[3]=deck[0];
		for(i=0;i<39;i++)
			deck[i]=deck[i+1];
		deck[39]=opencard;
		opencard=temp;
		if(count2==0)
			count2++;
		else
			names[opponent[3]]="cardback";
	}
	if(select[4]==true)
	{
		temp=opponent[4];
		opponent[4]=deck[0];
		for(i=0;i<39;i++)
			deck[i]=deck[i+1];
		deck[39]=opencard;
		opencard=temp;
		if(count2==0)
			count2++;
		else
```

```
                                names[opponent[4]]="cardback";
                        }
                }
                i6=ImageIO.read(new File("Cards/"+names[opencard]+".png"));
        }
        else
        {
                count=1;
                if(ct==0)
                {
                        if(select[0]==true)
                        {
                                temp=opponent[0];
                                opponent[0]=opencard;
                                opencard=temp;
                        }
                        if(select[1]==true)
                        {
                                temp=opponent[1];
                                opponent[1]=opencard;
                                opencard=temp;
                        }
                        if(select[2]==true)
                        {
                                temp=opponent[2];
                                opponent[2]=opencard;
                                opencard=temp;
                        }
                        if(select[3]==true)
                        {
                                temp=opponent[3];
                                opponent[3]=opencard;
                                opencard=temp;
                        }
                        if(select[4]==true)
                        {
                                temp=opponent[4];
                                opponent[4]=opencard;
                                opencard=temp;
                        }
                }
                else if(ct>=1)
                {
                        count2=0;
                        if(select[0]==true)
                        {
```

```
                temp=opponent[0];
                opponent[0]=opencard;
                opencard=temp;
                count2++;
        }
        if(select[1]==true)
        {
                temp=opponent[1];
                opponent[1]=opencard;
                opencard=temp;
                if(count2==0)
                        count2++;
                else
                        names[opponent[1]]="cardback";
        }
        if(select[2]==true)
        {
                temp=opponent[2];
                opponent[2]=opencard;
                opencard=temp;
                if(count2==0)
                        count2++;
                else
                        names[opponent[2]]="cardback";
        }
        if(select[3]==true)
        {
                temp=opponent[3];
                opponent[3]=opencard;
                opencard=temp;
                if(count2==0)
                        count2++;
                else
                        names[opponent[3]]="cardback";
        }
        if(select[4]==true)
        {
                temp=opponent[4];
                opponent[4]=opencard;
                opencard=temp;
                if(count2==0)
                        count2++;
                else
                        names[opponent[4]]="cardback";
        }
}
```

```
            i6=ImageIO.read(new File("Cards/"+names[opencard]+".png"));
        }
        repaint();

        for(i=0;i<5;i++)
        {
            if(names[opponent[i]].equals("cardback"))
                values[i+5]=0;
            else if(names[opponent[i]].substring(0,1).equals("A"))
                values[i+5]=1;
            else if(names[opponent[i]].substring(0,1).equals("2"))
                values[i+5]=2;
            else if(names[opponent[i]].substring(0,1).equals("3"))
                values[i+5]=3;
            else if(names[opponent[i]].substring(0,1).equals("4"))
                values[i+5]=4;
            else if(names[opponent[i]].substring(0,1).equals("5"))
                values[i+5]=5;
            else if(names[opponent[i]].substring(0,1).equals("6"))
                values[i+5]=6;
            else if(names[opponent[i]].substring(0,1).equals("7"))
                values[i+5]=7;
            else if(names[opponent[i]].substring(0,1).equals("8"))
                values[i+5]=8;
            else if(names[opponent[i]].substring(0,1).equals("9"))
                values[i+5]=9;
            else if(names[opponent[i]].substring(0,1).equals("1"))
                values[i+5]=10;
            else if(names[opponent[i]].substring(0,1).equals("J"))
                values[i+5]=10;
            else if(names[opponent[i]].substring(0,1).equals("Q"))
                values[i+5]=10;
            else if(names[opponent[i]].substring(0,1).equals("K"))
                values[i+5]=10;
        }
        for(i=0;i<5;i++)
        {
            if(names[joker].substring(0,1).equals(names[opponent[i]].substring(0,1)))
                values[i+5]=0;
        }
        if(names[opponent[0]].substring(0,1).equals("c"))
            i9=ImageIO.read(new File("Cards/blankwhite.png"));
        if(names[opponent[1]].substring(0,1).equals("c"))
            i10=ImageIO.read(new File("Cards/blankwhite.png"));
        if(names[opponent[2]].substring(0,1).equals("c"))
            i11=ImageIO.read(new File("Cards/blankwhite.png"));
```

```
		if(names[opponent[3]].substring(0,1).equals("c"))
			i12=ImageIO.read(new File("Cards/blankwhite.png"));
		if(names[opponent[4]].substring(0,1).equals("c"))
			i13=ImageIO.read(new File("Cards/blankwhite.png"));
		opponentvalue=0;
		for(i=0;i<5;i++)
			opponentvalue+=values[i+5];
		return count;
		}
		catch(IOException ex)
		{
			return -1;
		}
}
void checkwinner()
{
try
{
		int p,o;
		p=playervalue;
		o=opponentvalue;
		if(playervalue<opponentvalue)
		{
			JOptionPane.showMessageDialog(this,"OK!! You WIN!! ^_^");
			i2=ImageIO.read(new File("Cards/lost.png"));
			o=50;
		}
		else if(playervalue==opponentvalue)
			JOptionPane.showMessageDialog(this,"Phew!! Thats a DRAW!! ^_^");
		else
		{
			JOptionPane.showMessageDialog(this,"Woohoo!! You Lose ^_^");
			i2=ImageIO.read(new File("Cards/win.png"));

			p=50;
		}
		repaint();
		counter++;
		moves+="\n\nMove "+Integer.toString(counter)+"\n=======\n\n";
		counter=0;
		moves+="Opponent  :  \n";
		for(int i=0;i<5;i++)
		moves+=(names[opponent[i]])+" "+Integer.toString(values[i+5])+"\n";
		moves+="\nOpponentValue = "+Integer.toString(opponentvalue);
		moves+="\n\nPlayer :\n";
		for(int i=0;i<5;i++)
```

```
                moves+=(names[player[i]])+" "+Integer.toString(values[i])+"\n";
                moves+="\nPlayerValue = "+Integer.toString(playervalue);
                moves+="\n\n";
                round++;
                System.out.println("Round "+round+"\n=======");
                System.out.println(moves);
                moves="";
                playervalue=p;
                opponentvalue=o;
                finalscores();
                reshuffle();
        }
        catch(IOException e)
        {
        }
        }
        void finalscores()
        {
        try
        {
                opptotal+=opponentvalue;
                pltotal+=playervalue;
                if(opptotal>150)
                {
                        JOptionPane.showMessageDialog(this,"Your Total : "+pltotal+"\nMy Total
: "+opptotal+"\nSo, You WIN!! ^_^");
                        i2=ImageIO.read(new File("Cards/lost.png"));

                        System.exit(0);
                }
                else if(pltotal>150)
                {
                        JOptionPane.showMessageDialog(this,"Your Total : "+pltotal+"\nMy Total
: "+opptotal+"\nSo, You LOSE!! ^_^");
                        i2=ImageIO.read(new File("Cards/win.png"));
                        System.exit(0);
                }
                else
                        JOptionPane.showMessageDialog(this,"Scores Now..\nYour Total :
"+pltotal+"\nMy Total  : "+opptotal);
        }
        catch(IOException e)
        {
        }
        }
```

```
int exchangefromdeck(boolean flag1,boolean flag2,boolean flag3,boolean flag4,boolean flag5)
{
		addup();
		boolean mainflag=true;
		int count,count2,i,cv,temp;
		count=0;cv=0;
		if(flag1==true)
		{
			count++;
			cv=0;
		}
		if(flag2==true)
		{
			count++;
			if(count>1)
			{

if(!names[player[1]].substring(0,1).equals(names[player[cv]].substring(0,1)))
					mainflag=false;
			}
			cv=1;
		}
		if(flag3==true)
		{
			count++;
			if(count>1)
			{

if(names[player[2]].substring(0,1).equals(names[player[cv]].substring(0,1))){}
				else mainflag=false;
			}
			cv=2;
		}
		if(flag4==true)
		{
			count++;
			if(count>1)
			{

if(names[player[3]].substring(0,1).equals(names[player[cv]].substring(0,1))){}
				else mainflag=false;
			}
			cv=3;
		}
		if(flag5==true)
```

```
                {
                        count++;
                        if(count>1)
                        {

if(names[player[4]].substring(0,1).equals(names[player[cv]].substring(0,1)))
                                {}
                                else mainflag=false;
                        }
                        cv=4;
                }
                if(count==0)
                {
                        JOptionPane.showMessageDialog(this,"Select Atleast One Card!!");
                        return 0;
                }
                else if(mainflag==false)
                {
                        JOptionPane.showMessageDialog(this,"CHEATING");
                        return 0;
                }
                else
                {
                        if(count==1)
                        {
                                if(flag1==true)
                                {
                                        temp=player[0];
                                        player[0]=deck[0];
                                        for(i=0;i<39;i++)
                                                deck[i]=deck[i+1];
                                        deck[39]=opencard;
                                        opencard=temp;
                                }
                                if(flag2==true)
                                {
                                        temp=player[1];
                                        player[1]=deck[0];
                                        for(i=0;i<39;i++)
                                                deck[i]=deck[i+1];
                                        deck[39]=opencard;
                                        opencard=temp;
                                }
                                if(flag3==true)
                                {
                                        temp=player[2];
```

```
                player[2]=deck[0];
                for(i=0;i<39;i++)
                        deck[i]=deck[i+1];
                deck[39]=opencard;
                opencard=temp;
        }
        if(flag4==true)
        {
                temp=player[3];
                player[3]=deck[0];
                for(i=0;i<39;i++)
                        deck[i]=deck[i+1];
                deck[39]=opencard;
                opencard=temp;
        }
        if(flag5==true)
        {
                temp=player[4];
                player[4]=deck[0];
                for(i=0;i<39;i++)
                        deck[i]=deck[i+1];
                deck[39]=opencard;
                opencard=temp;
        }
}
else if(count>=2)
{
        count2=0;
        if(flag1==true)
        {
                temp=player[0];
                player[0]=deck[0];
                for(i=0;i<39;i++)
                        deck[i]=deck[i+1];
                deck[39]=opencard;
                opencard=temp;
                count2++;
        }
        if(flag2==true)
        {
                temp=player[1];
                player[1]=deck[0];
                for(i=0;i<39;i++)
                        deck[i]=deck[i+1];
                deck[39]=opencard;
                opencard=temp;
```

```
                if(count2==0)
                        count2++;
                else
                        names[player[1]]="cardback";
        }
        if(flag3==true)
        {
                temp=player[2];
                player[2]=deck[0];
                for(i=0;i<39;i++)
                        deck[i]=deck[i+1];
                deck[39]=opencard;
                opencard=temp;
                if(count2==0)
                        count2++;
                else
                        names[player[2]]="cardback";
        }
        if(flag4==true)
        {
                temp=player[3];
                player[3]=deck[0];
                for(i=0;i<39;i++)
                        deck[i]=deck[i+1];
                deck[39]=opencard;
                opencard=temp;
                if(count2==0)
                        count2++;
                else
                        names[player[3]]="cardback";
        }
        if(flag5==true)
        {
                temp=player[4];
                player[4]=deck[0];
                for(i=0;i<39;i++)
                        deck[i]=deck[i+1];
                deck[39]=opencard;
                opencard=temp;
                if(count2==0)
                        count2++;
                else
                        names[player[4]]="cardback";
        }
}
String jok=names[joker].substring(0,1);
```

```
for(i=5;i<10;i++)
{
        values[i]=opponent[i-5]%13;
        if(values[i]>9)
                values[i]=9;
        values[i]++;
}
for(i=5;i<10;i++)
{
        if(names[opponent[i-5]].substring(0,1).equals(jok))
                values[i]=0;
        if(names[opponent[i-5]].substring(0,1).equals("c"))
                values[i]=0;
}
playervalue=0;
opponentvalue=0;
for(i=0;i<5;i++)
{
        values[i]=player[i]%13;
        if(values[i]>9)
                values[i]=9;
        values[i]++;
}
values[10]=opencard%13;
if(values[10]>9)
        values[10]=9;
values[10]++;
values[11]=0;
for(i=12;i<52;i++)
{
        values[i]=deck[i-12]%13;
        if(values[i]>9)
                values[i]=9;
        values[i]++;
}
for(i=0;i<5;i++)
{
        if(names[player[i]].substring(0,1).equals(jok))
                values[i]=0;
        if(names[player[i]].substring(0,1).equals("c"))
                values[i]=0;
}
for(i=12;i<52;i++)
{
        if(names[deck[i-12]].substring(0,1).equals(jok))
                values[i]=0;
```

```
                        }
                        if(names[opencard].substring(0,1).equals(jok))
                                values[10]=0;
                        for(i=0;i<5;i++)
                        {
                                playervalue+=(values[i]);
                                opponentvalue+=values[i+5];
                        }

                        repaint();
                        return count;
                }
        }
        int exchangeopencard(boolean flag1,boolean flag2,boolean flag3,boolean flag4,boolean
flag5)
        {
                boolean mainflag=true;
                        addup();
                        int count,i,cv;
                        count=0;cv=0;
                        if(flag1==true)
                        {
                                count++;
                                cv=0;
                        }
                        if(flag2==true)
                        {
                                count++;
                                if(count>1)
                                {

        if(!names[player[1]].substring(0,1).equals(names[player[cv]].substring(0,1)))
                                                mainflag=false;
                                }
                                cv=1;
                        }
                        if(flag3==true)
                        {
                                count++;
                                if(count>1)
                                {

        if(names[player[2]].substring(0,1).equals(names[player[cv]].substring(0,1))){}
                                        else mainflag=false;
                                }
                                cv=2;
```

```
                }
                if(flag4==true)
                {
                        count++;
                        if(count>1)
                        {

if(names[player[3]].substring(0,1).equals(names[player[cv]].substring(0,1))){}
                                else mainflag=false;
                        }
                        cv=3;
                }
                if(flag5==true)
                {
                        count++;
                        if(count>1)
                        {

if(names[player[4]].substring(0,1).equals(names[player[cv]].substring(0,1)))
                                {}
                                else mainflag=false;
                        }
                        cv=4;
                }
                if(count==0)
                {
                        JOptionPane.showMessageDialog(this,"Select Atleast One Card!!");
                        return 0;
                }
                else if(mainflag==false)
                {
                        JOptionPane.showMessageDialog(this,"CHEATING");
                        return 0;
                }
                else
                {
                xposition=0;
                yposition=0;
                int temp;
                if(count==1)
                {
                        if(flag1==true)
                        {
                                temp=player[0];
                                player[0]=opencard;
                                opencard=temp;
```

```
                temp=values[0];
                values[0]=values[10];
                values[10]=temp;

        }
        if(flag2==true)
        {
                temp=player[1];
                player[1]=opencard;
                opencard=temp;
                temp=values[1];
                values[1]=values[10];
                values[10]=temp;

        }
        if(flag3==true)
        {
                temp=player[2];
                player[2]=opencard;
                opencard=temp;
                temp=values[2];
                values[2]=values[10];
                values[10]=temp;
        }
        if(flag4==true)
        {
                temp=player[3];
                player[3]=opencard;
                opencard=temp;
                temp=values[3];
                values[3]=values[10];
                values[10]=temp;
        }
        if(flag5==true)
        {
                temp=player[4];
                player[4]=opencard;
                opencard=temp;
                temp=values[4];
                values[4]=values[10];
                values[10]=temp;
        }
}
else if(count>=2)
{
        int count2=0;
```

```
if(flag1==true)
{
        count2++;
        temp=player[0];
        player[0]=opencard;
        opencard=temp;
        temp=values[0];
        values[0]=values[10];
        values[10]=temp;
}
if(flag2==true)
{
        if(count2==0)
        {
        count2++;
        temp=player[1];
        player[1]=opencard;
        opencard=temp;
        temp=values[1];
        values[1]=values[10];
        values[10]=temp;
        }
        else
        {
        names[player[1]]="cardback";
        values[1]=0;
        }
}
if(flag3==true)
{
        if(count2==0)
        {count2++;
        temp=player[2];
        player[2]=opencard;
        opencard=temp;
        temp=values[2];
        values[2]=values[10];
        values[10]=temp;
        }
        else
        {
        names[player[2]]="cardback";
        values[2]=0;
        }
}
if(flag4==true)
```

```
		{
			if(count2==0)
			{count2++;
			temp=player[3];
			player[3]=opencard;
			opencard=temp;
			temp=values[3];
			values[3]=values[10];
			values[10]=temp;
			}
			else
			{
			names[player[3]]="cardback";
			values[3]=0;
			}
		}
		if(flag5==true)
		{
			if(count2==0)
			{count2++;
			temp=player[4];
			player[4]=opencard;
			opencard=temp;
			temp=values[4];
			values[4]=values[10];
			values[10]=temp;
			}
			else
			{
			names[player[4]]="cardback";
			values[4]=0;
			}
		}
	}
	String jok=names[joker].substring(0,1);
	for(i=5;i<10;i++)
	{
		values[i]=opponent[i-5]%13;
		if(values[i]>9)
			values[i]=9;
		values[i]++;
	}
	for(i=5;i<10;i++)
	{
		if(names[opponent[i-5]].substring(0,1).equals(jok))
			values[i]=0;
```

```
            if(names[opponent[i-5]].substring(0,1).equals("c"))
                        values[i]=0;
}
playervalue=0;
opponentvalue=0;
for(i=0;i<5;i++)
{
            values[i]=player[i]%13;
            if(values[i]>9)
                        values[i]=9;
            values[i]++;
}
values[10]=opencard%13;
if(values[10]>9)
            values[10]=9;
values[10]++;
values[11]=0;
for(i=12;i<52;i++)
{
            values[i]=deck[i-12]%13;
            if(values[i]>9)
                        values[i]=9;
            values[i]++;
}
for(i=0;i<5;i++)
{
            if(names[player[i]].substring(0,1).equals(jok))
                        values[i]=0;
            if(names[player[i]].substring(0,1).equals("c"))
                        values[i]=0;
}
for(i=12;i<52;i++)
{
            if(names[deck[i-12]].substring(0,1).equals(jok))
                        values[i]=0;
}
if(names[opencard].substring(0,1).equals(jok))
            values[10]=0;

for(i=0;i<5;i++)
{
            playervalue+=(values[i]);
            opponentvalue+=values[i+5];
}
repaint();
return count;
```

```
                }
}
void reshuffle()
{
        int i;
        for(i=0;i<52;i++)
                names[i]=namestemp[i];
        reshuffle=0;
        Collections.shuffle(al);
        for(i=0;i<5;i++)
        {
                player[i]=(Integer)al.get(i);
                values[i]=player[i]%13;
                if(values[i]>9)
                        values[i]=9;
                values[i]++;
        }
        for(i=5;i<10;i++)
        {
                opponent[i-5]=(Integer)al.get(i);
                values[i]=opponent[i-5]%13;
                if(values[i]>9)
                        values[i]=9;
                values[i]++;
        }
        opencard=(Integer)al.get(10);
        values[10]=opencard%13;
        if(values[10]>9)
                values[10]=9;
        values[10]++;
        for(i=12;i<52;i++)
        {
                deck[i-12]=(Integer)al.get(i);
        }
        joker=(Integer)al.get(11);
        i=0;
        while(joker%13<6)
        {
                joker=deck[0];
                for(i=0;i<39;i++)
                        deck[i]=deck[i+1];
                deck[39]=joker;
        }
        values[11]=0;
        for(i=12;i<52;i++)
        {
```

```
		values[i]=deck[i-12]%13;
		if(values[i]>9)
			values[i]=9;
		values[i]++;
	}
	String jok=names[joker].substring(0,1);
	for(i=0;i<5;i++)
	{
		if(names[player[i]].substring(0,1).equals(jok))
			values[i]=0;
	}
	for(i=5;i<10;i++)
	{
		if(names[opponent[i-5]].substring(0,1).equals(jok))
			values[i]=0;
	}
	for(i=12;i<52;i++)
	{
		if(names[deck[i-12]].substring(0,1).equals(jok))
			values[i]=0;
	}
	if(names[opencard].substring(0,1).equals(jok))
		values[10]=0;
	playervalue=0;
	for(i=0;i<5;i++)
		playervalue+=values[i];
	opponentvalue=0;
	for(i=5;i<10;i++)
		opponentvalue+=values[i];
	try
	{
		xposition=0;
		yposition=0;
		i2=ImageIO.read(new File("Cards/thinking.png"));
		i8=ImageIO.read(new File("Cards/cardback.png"));
		i9=ImageIO.read(new File("Cards/cardback.png"));
		i10=ImageIO.read(new File("Cards/cardback.png"));
		i11=ImageIO.read(new File("Cards/cardback.png"));
		i12=ImageIO.read(new File("Cards/cardback.png"));
		i13=ImageIO.read(new File("Cards/cardback.png"));
		repaint();
	}
	catch(IOException e)
	{
		System.out.println("IO EXCEPTION");
	}
```

```
}
ImageComponent(int x,int y)
{
	int i;
	reshuffle=0;
	for(i=0;i<52;i++)
		al.add(new Integer(i));
	Collections.shuffle(al);
	for(i=0;i<5;i++)
	{
		player[i]=(Integer)al.get(i);
		values[i]=player[i]%13;
		if(values[i]>9)
			values[i]=9;
		values[i]++;
	}
	for(i=5;i<10;i++)
	{
		opponent[i-5]=(Integer)al.get(i);
		values[i]=opponent[i-5]%13;
		if(values[i]>9)
			values[i]=9;
		values[i]++;
	}
	opencard=(Integer)al.get(10);
	values[10]=opencard%13;
	if(values[10]>9)
		values[10]=9;
	values[10]++;
	for(i=12;i<52;i++)
	{
		deck[i-12]=(Integer)al.get(i);
	}
	joker=(Integer)al.get(11);
	i=0;
	int temp;
	while(joker%13<6)
	{
		temp=joker;
		joker=deck[0];
		for(i=0;i<39;i++)
			deck[i]=deck[i+1];
		deck[39]=joker;
	}
	values[11]=0;
	for(i=12;i<52;i++)
```

```
{
        values[i]=deck[i-12]%13;
        if(values[i]>9)
                values[i]=9;
        values[i]++;
}
String jok=names[joker].substring(0,1);
for(i=0;i<5;i++)
{
        if(names[player[i]].substring(0,1).equals(jok))
                values[i]=0;
}
for(i=5;i<10;i++)
{
        if(names[opponent[i-5]].substring(0,1).equals(jok))
                values[i]=0;
}
for(i=12;i<52;i++)
{
        if(names[deck[i-12]].substring(0,1).equals(jok))
                values[i]=0;
}
if(names[opencard].substring(0,1).equals(jok))
        values[10]=0;
playervalue=0;
for(i=0;i<5;i++)
        playervalue+=values[i];
opponentvalue=0;
for(i=5;i<10;i++)
        opponentvalue+=values[i];
try
{
        xposition=x;
        yposition=y;
        i1=ImageIO.read(new File("Cards/blankwhite.png"));
        i2=ImageIO.read(new File("Cards/thinking.png"));
        i8=ImageIO.read(new File("Cards/cardback.png"));
        i9=ImageIO.read(new File("Cards/cardback.png"));
        i10=ImageIO.read(new File("Cards/cardback.png"));
        i11=ImageIO.read(new File("Cards/cardback.png"));
        i12=ImageIO.read(new File("Cards/cardback.png"));
        i13=ImageIO.read(new File("Cards/cardback.png"));

}
catch(IOException e)
{
```

```
                        System.out.println("IO EXCEPTION");
                }
        }
}

public class cards extends JFrame implements ActionListener
{
        JPanel p=new JPanel();
        JPanel p1=new JPanel();
        JPanel p2=new JPanel();
        JPanel p3=new JPanel();
        JPanel p4=new JPanel();
        JMenuBar me=new JMenuBar();
        JTextField tf=new JTextField("",30);
        JMenu menu=new JMenu("MENU");
        JMenuItem aboutgame=new JMenuItem("How to play");
        JMenuItem author=new JMenuItem("About Author");
        JMenuItem quit=new JMenuItem("Exit");
        JButton b1=new JButton("");
        JButton b2=new JButton("");
        JButton b3=new JButton("");
        JButton b4=new JButton("");
        JButton b5=new JButton("");
        JButton b6=new JButton("");
        JButton b7=new JButton("");
        JButton b8=new JButton("");
        JButton b9=new JButton("");
        JButton b10=new JButton("");
        JCheckBox r1=new JCheckBox("Card 1");
        JCheckBox r2=new JCheckBox("Card 2");
        JCheckBox r3=new JCheckBox("Card 3");
        JCheckBox r4=new JCheckBox("Card 4");
        JCheckBox r5=new JCheckBox("Card 5");
        ImageComponent im;
        JLabel l1=new JLabel(" Open Card");
        JLabel l2=new JLabel("      Deck");
        JLabel l3=new JLabel("     Joker");
        JLabel l4=new JLabel("    BID!!!");
        JLabel l5=new JLabel("    Deselect");
        JLabel l6=new JLabel("Opponent Cards -:");
        JLabel l7=new JLabel("Player Cards -:");

        GridLayout g1=new GridLayout(1,5,10,10);
        GridBagConstraints g=new GridBagConstraints();
        GridBagConstraints c=new GridBagConstraints();
        String text;
```

```
	public cards()
	{
		setTitle("5 Cards");
		setSize(500,530);
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		setVisible(true);
		me.add(menu);
		menu.add(aboutgame);
		menu.add(author);
		menu.add(quit);
		aboutgame.setActionCommand("aboutgame");
		aboutgame.addActionListener(this);
		author.setActionCommand("author");
		author.addActionListener(this);
		quit.setActionCommand("quit");
		quit.addActionListener(this);

	aboutgame.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.
VK_G, java.awt.event.InputEvent.CTRL_MASK));

	author.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK_
A, java.awt.event.InputEvent.CTRL_MASK));

	quit.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK_Q,
java.awt.event.InputEvent.CTRL_MASK));
		setJMenuBar(me);

		r1.addActionListener(this);
		r2.addActionListener(this);
		r3.addActionListener(this);
		r4.addActionListener(this);
		r5.addActionListener(this);

		p2.setLayout(new GridBagLayout());

		g.fill     = GridBagConstraints.BOTH;
		g.weightx   = 0.5;
		g.weighty   = 0.5;
		g.gridwidth = 1;
		g.anchor    = GridBagConstraints.CENTER;
		g.insets    = new Insets(0,10,5,10);
		g.gridx     = 0;
		g.gridy     = 0;
		p2.add(b1,g);

		g.fill     = GridBagConstraints.BOTH;
```

```
g.weightx   = 0.5;
g.weighty   = 0.5;
g.gridwidth = 1;
g.anchor    = GridBagConstraints.CENTER;
g.insets    = new Insets(0,10,5,10);
g.gridx     = 1;
g.gridy     = 0;
p2.add(b2,g);

g.fill      = GridBagConstraints.BOTH;
g.weightx   = 0.5;
g.weighty   = 0.5;
g.gridwidth = 1;
g.anchor    = GridBagConstraints.CENTER;
g.insets    = new Insets(0,10,5,10);
g.gridx     = 2;
g.gridy     = 0;
p2.add(b3,g);

g.fill      = GridBagConstraints.BOTH;
g.weightx   = 0.5;
g.weighty   = 0.5;
g.gridwidth = 1;
g.anchor    = GridBagConstraints.CENTER;
g.insets    = new Insets(0,10,5,10);
g.gridx     = 3;
g.gridy     = 0;
p2.add(b9,g);

g.fill      = GridBagConstraints.BOTH;
g.weightx   = 0.5;
g.weighty   = 0.5;
g.gridwidth = 1;
g.anchor    = GridBagConstraints.CENTER;
g.insets    = new Insets(0,10,5,10);
g.gridx     = 4;
g.gridy     = 0;
p2.add(b10,g);

g.fill      = GridBagConstraints.BOTH;
g.weightx   = 0.5;
g.weighty   = 0.5;
g.gridwidth = 1;
g.anchor    = GridBagConstraints.CENTER;
g.insets    = new Insets(0,10,5,10);
g.gridx     = 0;
```

```
g.gridy    = 1;
p2.add(l1,g);

g.fill     = GridBagConstraints.BOTH;
g.weightx  = 0.5;
g.weighty  = 0.5;
g.gridwidth = 1;
g.anchor   = GridBagConstraints.CENTER;
g.insets   = new Insets(0,10,5,10);
g.gridx    = 1;
g.gridy    = 1;
p2.add(l2,g);

g.fill     = GridBagConstraints.BOTH;
g.weightx  = 0.5;
g.weighty  = 0.5;
g.gridwidth = 1;
g.anchor   = GridBagConstraints.CENTER;
g.insets   = new Insets(0,10,5,10);
g.gridx    = 2;
g.gridy    = 1;
p2.add(l3,g);

g.fill     = GridBagConstraints.BOTH;
g.weightx  = 0.5;
g.weighty  = 0.5;
g.gridwidth = 1;
g.anchor   = GridBagConstraints.CENTER;
g.insets   = new Insets(0,10,5,10);
g.gridx    = 3;
g.gridy    = 1;
p2.add(l4,g);

g.fill     = GridBagConstraints.BOTH;
g.weightx  = 0.5;
g.weighty  = 0.5;
g.gridwidth = 1;
g.anchor   = GridBagConstraints.CENTER;
g.insets   = new Insets(0,10,5,10);
g.gridx    = 4;
g.gridy    = 1;
p2.add(l5,g);

g.fill     = GridBagConstraints.BOTH;
g.weightx  = 0.5;
g.weighty  = 0.5;
```

```
                g.gridwidth = 5;
                g.anchor    = GridBagConstraints.CENTER;
                g.insets    = new Insets(0,10,5,10);
                g.gridx     = 0;
                g.gridy     = 4;
                p2.add(l6,g);

//-------------------------------------------------------//
                p4.setLayout(new GridBagLayout());

                c.fill      = GridBagConstraints.BOTH;
                c.weightx   = 0.5;
                c.weighty   = 0.5;
                c.gridwidth = 5;
                c.anchor    = GridBagConstraints.CENTER;
                c.insets    = new Insets(0,10,5,10);
                c.gridx     = 0;
                c.gridy     = 0;

                p4.add(tf,c);

                c.fill      = GridBagConstraints.BOTH;
                c.weightx   = 0.5;
                c.weighty   = 0.5;
                c.gridwidth = 5;
                c.anchor    = GridBagConstraints.CENTER;
                c.insets    = new Insets(0,10,5,10);
                c.gridx     = 0;
                c.gridy     = 1;

                p4.add(l7,c);

                c.fill      = GridBagConstraints.BOTH;
                c.weightx   = 0.5;
                c.weighty   = 0.5;
                c.gridwidth = 1;
                c.anchor    = GridBagConstraints.CENTER;
                c.insets    = new Insets(0,10,5,10);
                c.gridx     = 0;
                c.gridy     = 2;
                p4.add(b4,c);

                c.fill      = GridBagConstraints.BOTH;
                c.weightx   = 0.5;
                c.weighty   = 0.5;
                c.gridwidth = 1;
```

```
c.anchor    = GridBagConstraints.CENTER;
c.insets    = new Insets(0,10,5,10);
c.gridx     = 1;
c.gridy     = 2;
p4.add(b5,c);

c.fill      = GridBagConstraints.BOTH;
c.weightx   = 0.5;
c.weighty   = 0.5;
c.gridwidth = 1;
c.anchor    = GridBagConstraints.CENTER;
c.insets    = new Insets(0,10,5,10);
c.gridx     = 2;
c.gridy     = 2;
p4.add(b6,c);

c.fill      = GridBagConstraints.BOTH;
c.weightx   = 0.5;
c.weighty   = 0.5;
c.gridwidth = 1;
c.anchor    = GridBagConstraints.CENTER;
c.insets    = new Insets(0,10,5,10);
c.gridx     = 3;
c.gridy     = 2;
p4.add(b7,c);

c.fill      = GridBagConstraints.BOTH;
c.weightx   = 0.5;
c.weighty   = 0.5;
c.gridwidth = 1;
c.anchor    = GridBagConstraints.CENTER;
c.insets    = new Insets(0,10,5,10);
c.gridx     = 4;
c.gridy     = 2;
p4.add(b8,c);


add(p2,BorderLayout.NORTH);
add(p4,BorderLayout.SOUTH);
im=new ImageComponent(0,0);
add(im);
setResizable(false);
tf.setText(im.playerscore());
tf.setBounds(10,10,30,50);
b1.setIcon(new ImageIcon("Cards/"+im.names[im.opencard]+".png"));
b2.setIcon(new ImageIcon("Cards/"+"cardback"+".png"));
```

```
            b3.setIcon(new ImageIcon("Cards/"+im.names[im.joker]+".png"));
            b4.setIcon(new ImageIcon("Cards/"+im.names[im.player[0]]+".png"));
            b5.setIcon(new ImageIcon("Cards/"+im.names[im.player[1]]+".png"));
            b6.setIcon(new ImageIcon("Cards/"+im.names[im.player[2]]+".png"));
            b7.setIcon(new ImageIcon("Cards/"+im.names[im.player[3]]+".png"));
            b8.setIcon(new ImageIcon("Cards/"+im.names[im.player[4]]+".png"));
            b9.setIcon(new ImageIcon("Cards/bid.png"));
            b10.setIcon(new ImageIcon("Cards/deselect.png"));
            b1.setActionCommand("opencard");
            b2.setActionCommand("deck");
            b3.setActionCommand("joker");
            b4.setActionCommand("1");
            b5.setActionCommand("2");
            b6.setActionCommand("3");
            b7.setActionCommand("4");
            b8.setActionCommand("5");
            b9.setActionCommand("bid");
            b10.setActionCommand("deselect");
            b1.addActionListener(this);
            b2.addActionListener(this);
            b3.addActionListener(this);
            b4.addActionListener(this);
            b5.addActionListener(this);
            b6.addActionListener(this);
            b7.addActionListener(this);
            b8.addActionListener(this);
            b9.addActionListener(this);
            b10.addActionListener(this);
            p2.setBackground(Color.WHITE);
            p4.setBackground(Color.WHITE);
        }
        public void actionPerformed(ActionEvent event)
        {
            boolean enable;
            String input="";
            int retopencard,retcnt,opp;
            if("aboutgame".equals(event.getActionCommand()))
            {
                JOptionPane.showMessageDialog(this,"This is a 2 player game, one being
you and the other being the system. \nYou will be given 5 cards. Each card has a value associated
with it. \nFor example, 2 from any suit has a value 2. Similarly, 8 from \nany suit has a value 8. A
point to be noted here is that 10, J, Q \nand K of all suits have a value of 10 only. Also, the joker
card is set \na value of 0. So, if 8 is the joker, then all the 8's get a value of \n0. The player's basic
motive is to reduce the value of cards he/she\n has. The sum of all the 5 cards must be as low as
possible. The \nplayer can select one card (or more than one card if they have the same \nname,
like 2 or more J's, 2 or more 10's etc.) and replace them \nwith one card from the deck or the
```

```
open card and place their selected \ncard(s) as the open card. Now, the turn goes to the
opponent and he \nselects either from open card or deck and places his replacement \ncard on
the open card. At all times, both the player and the opponent \nhas 5 or lesser cards only! When
the player feels that his total \nvalue is considerably less, he/she can cast a bid. If the opponent's
\ntotal is less than the player's score, the opponent wins and 50 points \nget added to the
player. Otherwise, 50 points is added to the \nopponent. (Even the opponent will bid. It is not
like the game will \nstop only after the player bids). Then, the cards are reshuffled and \na new
game is started. The game stops when either player or the opponent \nreaches 150 points. The
one who reaches first, is the loser.");
                }
                else if("author".equals(event.getActionCommand()))
                {
                        JOptionPane.showMessageDialog(this,"This game is designed by none
other than Pramodh, \nCSE department (at present 3rd year), \nSSN College Of Engineering");
                }
                else if("quit".equals(event.getActionCommand()))
                {
                        System.exit(0);
                }
                if("1".equals(event.getActionCommand()))
                {
                        input="";
                        r1.setSelected(true);
                        if(r1.isSelected())
                                input+="1 ";
                        if(r2.isSelected())
                                input+="2 ";
                        if(r3.isSelected())
                                input+="3 ";
                        if(r4.isSelected())
                                input+="4 ";
                        if(r5.isSelected())
                                input+="5 ";
                        tf.setText("Card(s) Selected  : "+input);
                }
                if("2".equals(event.getActionCommand()))
                {
                        input="";
                        r2.setSelected(true);
                        if(r1.isSelected())
                                input+="1 ";
                        if(r2.isSelected())
                                input+="2 ";
                        if(r3.isSelected())
                                input+="3 ";
                        if(r4.isSelected())
```

```
                input+="4 ";
        if(r5.isSelected())
                input+="5 ";
        tf.setText("Card(s) Selected  : "+input);
}
if("3".equals(event.getActionCommand()))
{
        input="";
        r3.setSelected(true);
        if(r1.isSelected())
                input+="1 ";
        if(r2.isSelected())
                input+="2 ";
        if(r3.isSelected())
                input+="3 ";
        if(r4.isSelected())
                input+="4 ";
        if(r5.isSelected())
                input+="5 ";
        tf.setText("Card(s) Selected  : "+input);
}
if("4".equals(event.getActionCommand()))
{
        input="";
        r4.setSelected(true);
        if(r1.isSelected())
                input+="1 ";
        if(r2.isSelected())
                input+="2 ";
        if(r3.isSelected())
                input+="3 ";
        if(r4.isSelected())
                input+="4 ";
        if(r5.isSelected())
                input+="5 ";
        tf.setText("Card(s) Selected  : "+input);
}
if("5".equals(event.getActionCommand()))
{
        input="";
        r5.setSelected(true);
        if(r1.isSelected())
                input+="1 ";
        if(r2.isSelected())
                input+="2 ";
        if(r3.isSelected())
```

```
                    input+="3 ";
                if(r4.isSelected())
                    input+="4 ";
                if(r5.isSelected())
                    input+="5 ";
                tf.setText("Card(s) Selected  : "+input);
            }
            if("joker".equals(event.getActionCommand()))
            {
                JOptionPane.showMessageDialog(this,"You Can't Replace With Joker!!");
            }
            if("opencard".equals(event.getActionCommand()))
            {

        retopencard=im.exchangeopencard(r1.isSelected(),r2.isSelected(),r3.isSelected(),r4.isSele
cted(),r5.isSelected());
                if(retopencard>=2)
                {
                    retcnt=0;
                    if(r1.isSelected())
                    {
                        retcnt++;
                        r1.setSelected(false);
                    }
                    if(r2.isSelected())
                    {
                        r2.setSelected(false);
                        if(retcnt==0)
                            retcnt++;
                        else
                            r2.setEnabled(false);
                    }
                    if(r3.isSelected())
                    {
                        r3.setSelected(false);
                        if(retcnt==0)
                            retcnt++;
                        else
                            r3.setEnabled(false);
                    }
                    if(r4.isSelected())
                    {       r4.setSelected(false);
                        if(retcnt==0)
                            retcnt++;
                        else
                            r4.setEnabled(false);
```

```
                }
                if(r5.isSelected())
                {       r5.setSelected(false);
                        if(retcnt==0)
                                retcnt++;
                        else
                                r5.setEnabled(false);
                }
        }
        if(r1.isSelected())
                r1.setSelected(false);
        if(r2.isSelected())
                r2.setSelected(false);
        if(r3.isSelected())
                r3.setSelected(false);
        if(r4.isSelected())
                r4.setSelected(false);
        if(r5.isSelected())
                r5.setSelected(false);
        if(retopencard!=0)
        {
        opp=im.opponentplay();
        if(opp==0)
                tf.setText(im.playerscore()+"\tOpponent Exchanged from Deck");
        else if(opp==1)
                tf.setText(im.playerscore()+"\tOpponent Exchanged from
OpenCard");
        b1.setIcon(new ImageIcon("Cards/"+im.names[im.opencard]+".png"));
        b2.setIcon(new ImageIcon("Cards/"+"cardback"+".png"));
        b3.setIcon(new ImageIcon("Cards/"+im.names[im.joker]+".png"));
        b4.setIcon(new ImageIcon("Cards/"+im.names[im.player[0]]+".png"));
        b5.setIcon(new ImageIcon("Cards/"+im.names[im.player[1]]+".png"));
        b6.setIcon(new ImageIcon("Cards/"+im.names[im.player[2]]+".png"));
        b7.setIcon(new ImageIcon("Cards/"+im.names[im.player[3]]+".png"));
        b8.setIcon(new ImageIcon("Cards/"+im.names[im.player[4]]+".png"));
        enable=im.opponentbid();
        if(enable==true)
        {
                r1.setEnabled(true);
                r2.setEnabled(true);
                r3.setEnabled(true);
                r4.setEnabled(true);
                r5.setEnabled(true);
        }
        input="";
        b1.setIcon(new ImageIcon("Cards/"+im.names[im.opencard]+".png"));
```

```
                b2.setIcon(new ImageIcon("Cards/"+"cardback"+".png"));
                b3.setIcon(new ImageIcon("Cards/"+im.names[im.joker]+".png"));
                b4.setIcon(new ImageIcon("Cards/"+im.names[im.player[0]]+".png"));
                b5.setIcon(new ImageIcon("Cards/"+im.names[im.player[1]]+".png"));
                b6.setIcon(new ImageIcon("Cards/"+im.names[im.player[2]]+".png"));
                b7.setIcon(new ImageIcon("Cards/"+im.names[im.player[3]]+".png"));
                b8.setIcon(new ImageIcon("Cards/"+im.names[im.player[4]]+".png"));
                if(im.names[im.player[0]].equals("cardback"))
                        b4.setEnabled(false);
                if(im.names[im.player[1]].equals("cardback"))
                        b5.setEnabled(false);
                if(im.names[im.player[2]].equals("cardback"))
                        b6.setEnabled(false);
                if(im.names[im.player[3]].equals("cardback"))
                        b7.setEnabled(false);
                if(im.names[im.player[4]].equals("cardback"))
                        b8.setEnabled(false);
                if(im.reshuffle==0)
                {
                        im.reshuffle=1;
                        if(r1.isSelected())
                                r1.setEnabled(true);
                        if(r2.isSelected())
                                r2.setEnabled(true);
                        if(r3.isSelected())
                                r3.setEnabled(true);
                        if(r4.isSelected())
                                r4.setEnabled(true);
                        if(r5.isSelected())
                                r5.setEnabled(true);
                        b4.setEnabled(true);
                        b5.setEnabled(true);
                        b6.setEnabled(true);
                        b7.setEnabled(true);
                        b8.setEnabled(true);
                        tf.setText(im.playerscore());
                }

                }

        }
        if("deck".equals(event.getActionCommand()))
        {

    retopencard=im.exchangefromdeck(r1.isSelected(),r2.isSelected(),r3.isSelected(),r4.isSele
cted(),r5.isSelected());
```

```
if(retopencard>=2)
{
        retcnt=0;
        if(r1.isSelected())
        {
                retcnt++;
                r1.setSelected(false);
        }
        if(r2.isSelected())
        {
                r2.setSelected(false);
                if(retcnt==0)
                        retcnt++;
                else
                        r2.setEnabled(false);
        }
        if(r3.isSelected())
        {       r3.setSelected(false);
                if(retcnt==0)
                        retcnt++;
                else
                        r3.setEnabled(false);
        }
        if(r4.isSelected())
        {       r4.setSelected(false);
                if(retcnt==0)
                        retcnt++;
                else
                        r4.setEnabled(false);
        }
        if(r5.isSelected())
        {       r5.setSelected(false);
                if(retcnt==0)
                        retcnt++;
                else
                r5.setEnabled(false);
        }
}
if(r1.isSelected())
        r1.setSelected(false);
if(r2.isSelected())
        r2.setSelected(false);
if(r3.isSelected())
        r3.setSelected(false);
if(r4.isSelected())
        r4.setSelected(false);
```

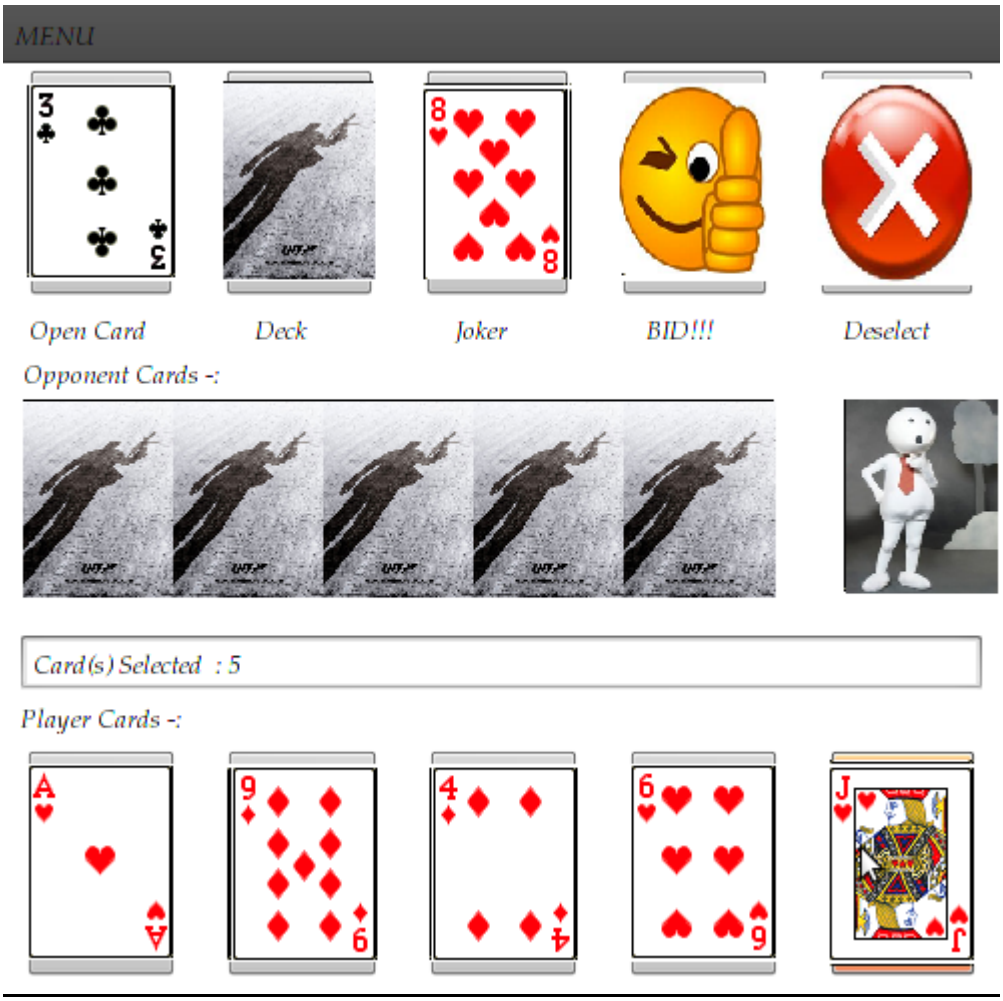
```
if(r5.isSelected())
        r5.setSelected(false);
if(retopencard!=0)
{opp=im.opponentplay();
if(opp==0)
        tf.setText(im.playerscore()+"\tOpponent Exchanged from Deck");
else if(opp==1)
        tf.setText(im.playerscore()+"\tOpponent Exchanged from
OpenCard");
b1.setIcon(new ImageIcon("Cards/"+im.names[im.opencard]+".png"));
b2.setIcon(new ImageIcon("Cards/"+"cardback"+".png"));
b3.setIcon(new ImageIcon("Cards/"+im.names[im.joker]+".png"));
b4.setIcon(new ImageIcon("Cards/"+im.names[im.player[0]]+".png"));
b5.setIcon(new ImageIcon("Cards/"+im.names[im.player[1]]+".png"));
b6.setIcon(new ImageIcon("Cards/"+im.names[im.player[2]]+".png"));
b7.setIcon(new ImageIcon("Cards/"+im.names[im.player[3]]+".png"));
b8.setIcon(new ImageIcon("Cards/"+im.names[im.player[4]]+".png"));

enable=im.opponentbid();
if(enable==true)
{
        r1.setEnabled(true);
        r2.setEnabled(true);
        r3.setEnabled(true);
        r4.setEnabled(true);
        r5.setEnabled(true);
}
input="";
b1.setIcon(new ImageIcon("Cards/"+im.names[im.opencard]+".png"));
b2.setIcon(new ImageIcon("Cards/"+"cardback"+".png"));
b3.setIcon(new ImageIcon("Cards/"+im.names[im.joker]+".png"));
b4.setIcon(new ImageIcon("Cards/"+im.names[im.player[0]]+".png"));
b5.setIcon(new ImageIcon("Cards/"+im.names[im.player[1]]+".png"));
b6.setIcon(new ImageIcon("Cards/"+im.names[im.player[2]]+".png"));
b7.setIcon(new ImageIcon("Cards/"+im.names[im.player[3]]+".png"));
b8.setIcon(new ImageIcon("Cards/"+im.names[im.player[4]]+".png"));
if(im.names[im.player[0]].equals("cardback"))
        b4.setEnabled(false);
if(im.names[im.player[1]].equals("cardback"))
        b5.setEnabled(false);
if(im.names[im.player[2]].equals("cardback"))
        b6.setEnabled(false);
if(im.names[im.player[3]].equals("cardback"))
        b7.setEnabled(false);
if(im.names[im.player[4]].equals("cardback"))
        b8.setEnabled(false);
```

```
            if(im.reshuffle==0)
            {
                    im.reshuffle=1;
                    if(r1.isSelected())
                            r1.setEnabled(true);
                    if(r2.isSelected())
                            r2.setEnabled(true);
                    if(r3.isSelected())
                            r3.setEnabled(true);
                    if(r4.isSelected())
                            r4.setEnabled(true);
                    if(r5.isSelected())
                            r5.setEnabled(true);
                    b4.setEnabled(true);
                    b5.setEnabled(true);
                    b6.setEnabled(true);
                    b7.setEnabled(true);
                    b8.setEnabled(true);
                    tf.setText(im.playerscore());
            }
            }

    }
    if("bid".equals(event.getActionCommand()))
    {
            im.checkwinner();
            r1.setEnabled(true);
            r2.setEnabled(true);
            r3.setEnabled(true);
            r4.setEnabled(true);
            r5.setEnabled(true);
            b4.setEnabled(true);
            b5.setEnabled(true);
            b6.setEnabled(true);
            b7.setEnabled(true);
            b8.setEnabled(true);
            b1.setIcon(new ImageIcon("Cards/"+im.names[im.opencard]+".png"));
            b2.setIcon(new ImageIcon("Cards/"+"cardback"+".png"));
            b3.setIcon(new ImageIcon("Cards/"+im.names[im.joker]+".png"));
            b4.setIcon(new ImageIcon("Cards/"+im.names[im.player[0]]+".png"));
            b5.setIcon(new ImageIcon("Cards/"+im.names[im.player[1]]+".png"));
            b6.setIcon(new ImageIcon("Cards/"+im.names[im.player[2]]+".png"));
            b7.setIcon(new ImageIcon("Cards/"+im.names[im.player[3]]+".png"));
            b8.setIcon(new ImageIcon("Cards/"+im.names[im.player[4]]+".png"));
            tf.setText("Next Game\t"+im.playerscore());
    }
```

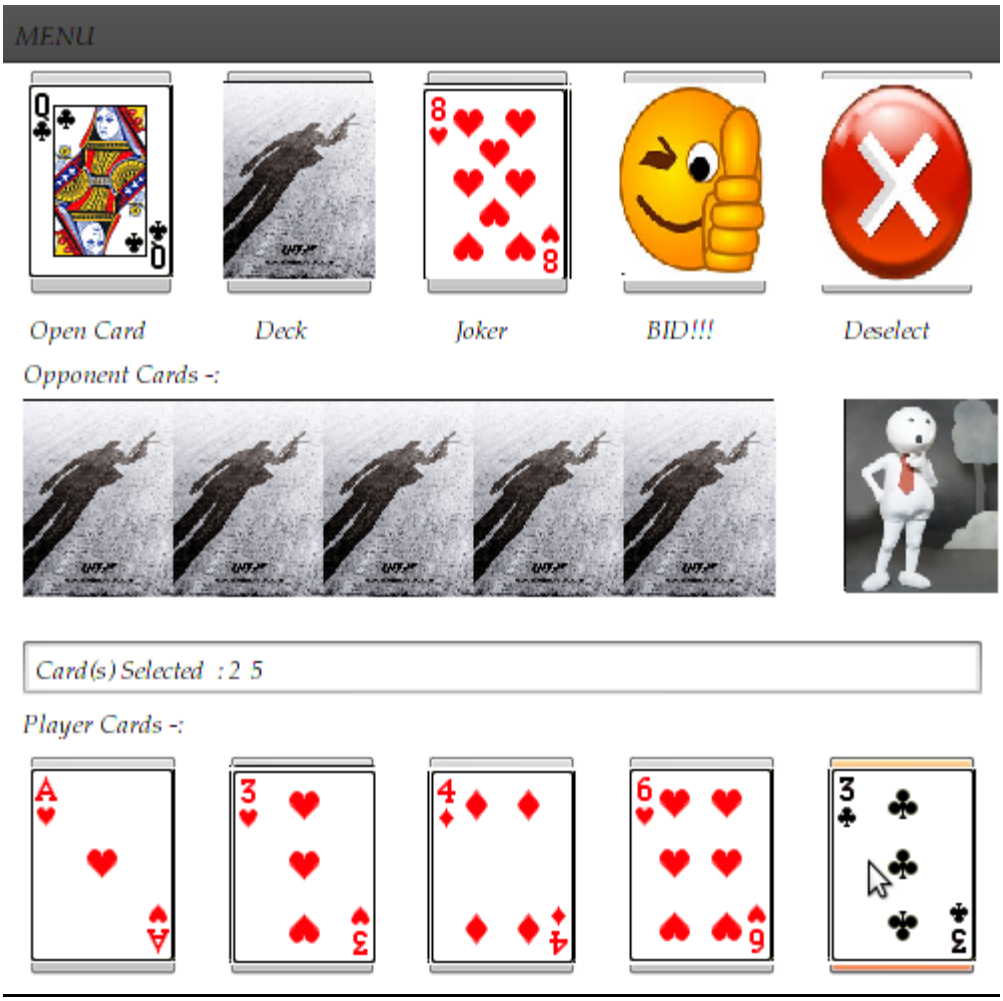
```
                if("deselect".equals(event.getActionCommand()))
                {
                        if(r1.isSelected())
                        r1.setSelected(false);
                        if(r2.isSelected())
                        r2.setSelected(false);
                        if(r3.isSelected())
                        r3.setSelected(false);
                        if(r4.isSelected())
                        r4.setSelected(false);
                        if(r5.isSelected())
                        r5.setSelected(false);
                        tf.setText(im.playerscore());
                }
        }
        public static void main(String[] args)
        {
                try
                {

        UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
                }
                catch (Exception unused)
                {
                }
                cards c=new cards();
                c.setBackground(Color.WHITE);
                c.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                c.setVisible(true);

        }
}
```

OUTPUT-:

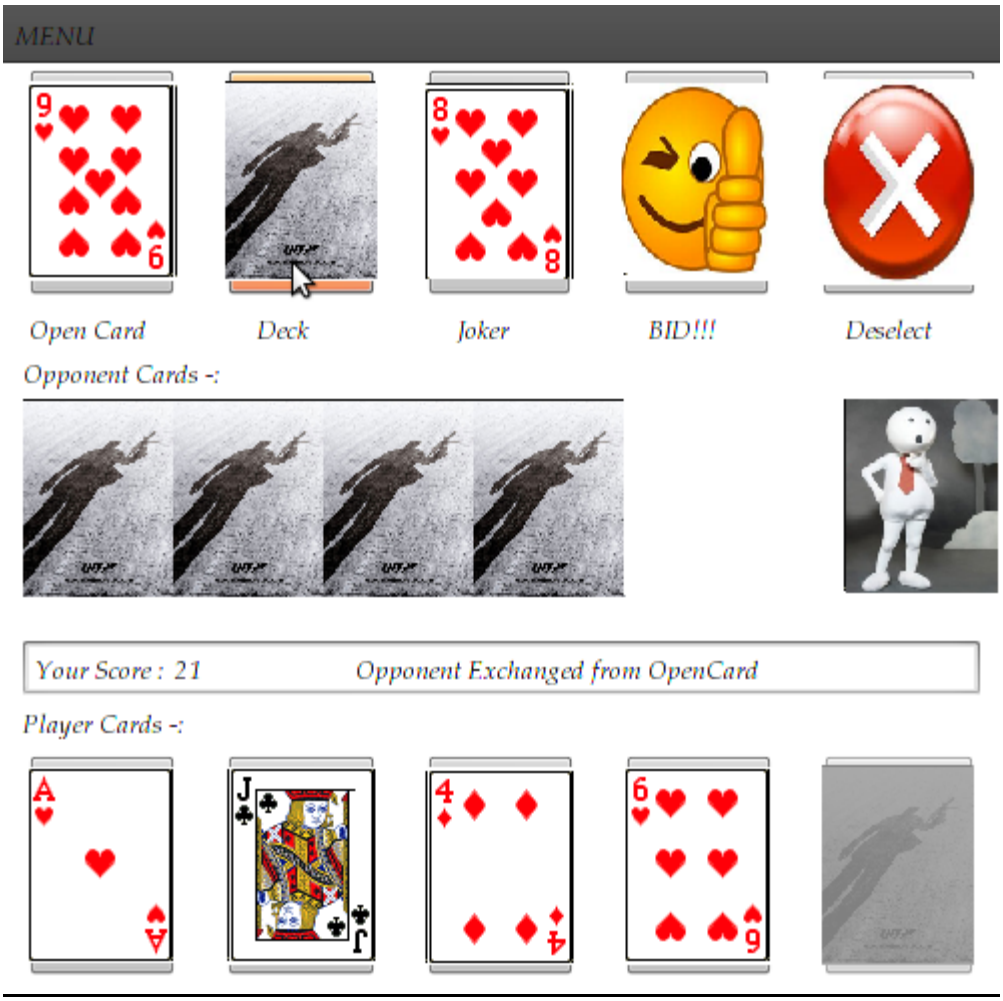
Selecting a Card-:



Replacing With Open Card-:



After Replacing -:



Selecting 2 cards that are not equal-:

Error Message Printed-:

Bidding by Player-:

Player Lost-:


5 Cards
MENU
Open Card
Deck
Joker
BID!!!
Deselect
Opponent Cards -:
Message
Woohoo!! You Lose ^_^
OK
Your Score : 21
Player Cards -:

Scores after game is over (Total game ends when either player or opponent reaches 150)-: