Name	**Hatim Yusuf Sawai**
UID no.	**2021300108**
Experiment No.	**8**

AIM:	Programs on Interfaces and multiple inheritance in java
Program 1	
PROBLEM STATEMENT:	Write a program that plays the game of hangman. In hangman, the computer begins by selecting a secret word at random from a list of possibilities. It then prints out a row of dashes—one for each letter in the secret word—and asks the user to guess a letter. If the user guesses a letter that appears in the word, the word is redisplayed with all instances of that letter shown in the correct positions, along with any letters guessed correctly on previous turns. If the letter does not appear in the word, the player is charged with an incorrect guess. The player keeps guessing letters until either (1) the player has correctly guessed all the letters in the word or (2) the player has made eight incorrect guesses. To separate the process of choosing a secret word from the rest of the game, define and implement an interface called randword that exports two functions: InitDictionary and ChooseRandomWord. InitDictionary has a list of words, stored into an array declared as a static global variable in the implementation. ChooseRandomWord takes no arguments and returns a word chosen at random from the internally maintained array.

PROGRAM:	import *java.util.Scanner;*
	import *java.util.Arrays;*
	import *java.lang.Math;*
	interface randword {
	String[] words = new String[10];
	public void initDictionary();
	public String ChooseRandomWord();
	}
	class Game *implements* randword {

```
    public void initDictionary() {
        words[0] = "AIRPLANE";
        words[1] = "UMBRELLA";
        words[2] = "FRIENDS";
        words[3] = "ONOMATOPOEIA";
        words[4] = "UNIVERSITY";
        words[5] = "CLICKBAIT";
        words[6] = "TECHNOLOGY";
        words[7] = "YOUTUBE";
        words[8] = "WEBSITE";
        words[9] = "ANIME";
    }
    public String ChooseRandomWord() {
        int index = (int)((Math.random())*10);
        return words[index];
    }
}
class shape {
    String[] figure = {"+----+", "|", "|    |", "|    O", "|   /|\\", "|   / \\"};
    void draw(int n) {
        if(n==7) {
            System.out.println(figure[0]);
        }
        else if(n==6) {
            System.out.println(figure[1]);
            System.out.println(figure[1]);
            System.out.println(figure[0]);
        }
        else if(n==5) {
            for(int i=0;i<4;i++) {
                System.out.println(figure[1]);
            }
            System.out.println(figure[0]);
        }
        else if(n==4) {
            for (int i=0;i<5;i++) {
```

```
            System.out.println(figure[1]);
        }
        System.out.println(figure[0]);
    }
    else if(n==3) {
        System.out.println(figure[0]);
        System.out.println(figure[2]);
        for (int i=0;i<5;i++) {
            System.out.println(figure[1]);
        }
        System.out.println(figure[0]);
    }
    else if(n==2) {
        System.out.println(figure[0]);
        System.out.println(figure[2]);
        System.out.println(figure[3]);
        for (int i=0;i<4;i++) {
            System.out.println(figure[1]);
        }
        System.out.println(figure[0]);
    }
    else if(n==1) {
        System.out.println(figure[0]);
        System.out.println(figure[2]);
        System.out.println(figure[3]);
        System.out.println(figure[4]);
        for (int i=0;i<3;i++) {
            System.out.println(figure[1]);
        }
        System.out.println(figure[0]);
    }
    else if(n==0) {
        System.out.println(figure[0]);
        System.out.println(figure[2]);
        System.out.println(figure[3]);
        System.out.println(figure[4]);
```

```
            System.out.println(figure[5]);
            for (int i=0;i<2;i++) {
                System.out.println(figure[1]);
            }
            System.out.println(figure[0]);
        }
    }
}
public class hangman {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Game g1 = new Game();
        shape s1 = new shape();
        g1.initDictionary();
        String secret = new String();
        String letter = new String();
        int flag,temp;
        while(true) {
            System.out.println("WELCOME TO HANGMAN");
            System.out.println("I will guess a secret word. On each turn, you
guess a letter. If the letter is in the secret word, I will show you where it
appears; if not, a part of your body gets strung up on the scaffold. The
objective is to guess the word before you are hung.\nNote:Only one
instance of a letter will be revealed if the guess is correct!");
            secret = g1.ChooseRandomWord();
            char[] guess = new char[secret.length()];
            char[] secret_char = secret.toCharArray();
            for(int i=0;i<secret.length();i++) {
                guess[i] = '-';
            }
            int i=8;
            while(i>0) {
                System.out.println("The secret word looks like this:");
                System.out.println(guess);
                System.out.printf("You have %d guesses left!\nGuess a letter:
",i);
```

```
            letter = sc.next();
            temp = secret.indexOf(letter.toUpperCase());
            if(temp!=-1) {
               System.out.println("Correct!");
               guess[temp] = secret_char[temp];
               secret = secret.replaceFirst(letter.toUpperCase()," ");
            } else {
               i--;
               System.out.println("Wrong!");
               s1.draw(i);
            }
            if(Arrays.equals(guess,secret_char)==true) {
               System.out.println(guess);
               System.out.println("You win!");
               break;
            }
         }
         if(i==0) {
            System.out.print("You lose!\nThe secret word was: ");
            System.out.println(secret_char);
         }
         System.out.println("Do you want to play again?(yes=1/no=0)");
         flag = sc.nextInt();
         if (flag == 0) {
            break;
         }
      }
      sc.close();
   }
}
```

RESULT:

Win condition:

```
WELCOME TO HANGMAN
I will guess a secret word. On each turn, you guess a letter. If the letter is in the secret word, I will show you where it appears; i
f not, a part of your body gets strung up on the scaffold. The objective is to guess the word before you are hung.
Note:Only one instance of a letter will be revealed if the guess is correct!
The secret word looks like this:
-------
You have 8 guesses left!
Guess a letter: A
Wrong!
+----+
The secret word looks like this:
-------
You have 7 guesses left!
Guess a letter: o
Correct!
The secret word looks like this:
-O-----
You have 7 guesses left!
Guess a letter: u
Correct!
The secret word looks like this:
YOUT-B-
You have 7 guesses left!
Guess a letter: u
Correct!
The secret word looks like this:
YOUTUB-
You have 7 guesses left!
Guess a letter: e
Correct!
YOUTUBE
You win!
Do you want to play again?(yes=1/no=0)
```

Lose Condition:

```
WELCOME TO HANGMAN
I will guess a secret word. On each turn, you guess a letter. If the letter is in the secret word, I will show you where it appears; i
f not, a part of your body gets strung up on the scaffold. The objective is to guess the word before you are hung.
Note:Only one instance of a letter will be revealed if the guess is correct!
The secret word looks like this:
---------
You have 8 guesses left!
Guess a letter: a
Correct!
The secret word looks like this:
------A--
You have 8 guesses left!
Guess a letter: p
Wrong!
+----+
The secret word looks like this:
------A--
You have 7 guesses left!
Guess a letter: d
Wrong!
|
|
+----+
The secret word looks like this:
------A--
```

```
The secret word looks like this:
----KBA-T
You have 1 guesses left!
Guess a letter: e
Wrong!
+----+
|    |
|    O
|   /|\
|   / \
|
|
+----+
You lose!
The secret word was: CLICKBAIT
Do you want to play again?(yes=1/no=0)
0
PS D:\Java Practicals\Experiment_8>
```

Program 2

PROBLEM STATEMENT:

Design and implement an interfaces:

• A interface rankT that allows you to represent the rank of a card. The values of type rankT include the integers between 2 and 10 but should also include the constants Ace, Jack, Queen, and King.

• A interface suitT consisting of the four suits: Clubs, Diamonds, Hearts, and Spades.

• A interface cardT that combines a rank and a suit.

- It has a function NewCard() that creates a card from the rank and suit values.

a) A PrintCard Class has a function CardName() that returns a string identifying the card. The result of CardName begins with a rank indicator (which is one of A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, or K), followed by a one-character suit (C, D, H, or S). Note that the result is usually a two-character string, but contains three characters if the rank is a 10.

b) Using the interfaces initialize a complete deck of 52 cards. It have method ShuffleCard() shuffles cards and then displays the shuffled values, as shown in the following sample run

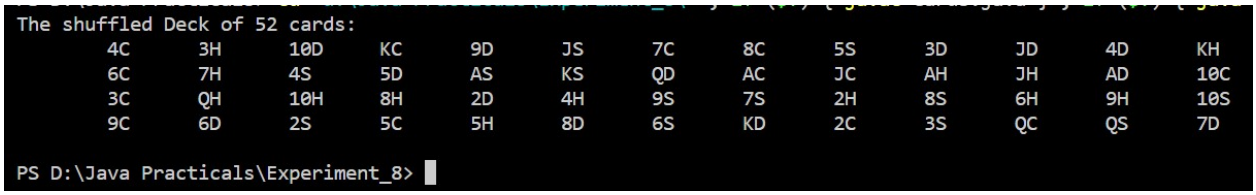
PROGRAM:

```
import java.util.*;
interface RankT {
   List<String> ranks = new ArrayList<>();
}
interface suitT {
   List<String> suits = new ArrayList<>();
}
interface CardT {
   List<String> cards = new ArrayList<>();
   void newCard();
}
class genCard implements RankT,suitT,CardT {
   public void newCard() {

Collections.addAll(ranks,"A","2","3","4","5","6","7","8","9","10","J","Q","K");
      Collections.addAll(suits, "H","C","S","D");
      for(int i=0;i<suits.size();i++) {
         for(int j=0;j<ranks.size();j++) {
            cards.add(ranks.get(j)+suits.get(i));
         }
      }
   }
   void printDeck() {
      Collections.shuffle(cards);
      System.out.println("The shuffled Deck of 52 cards: ");
      for(int i=0;i<cards.size();i++) {
         if (i % 13 == 0 && i != 0) {
            System.out.print("\n");
         }
         System.out.print("\t"+cards.get(i));
      }
      System.out.println("\n");
   }
}
public class Cards {
   public static void main(String[] args) {
```

```
        genCard gc = new genCard();
        gc.newCard();
        gc.printDeck();
    }
}
```

RESULT:



CONCLUSION: In this experiment, we learnt how to use interfaces and make real-life applications like the hangman game or shuffle a deck of cards. We used interfaces to apply the concept of multiple inheritances in java.