ACSL
American Computer Science League

2016 - 2017 **Contest #1**

ACSL AGRAM
SENIOR DIVISION

PROBLEM: Agram is a card game for 2 players, using the cards from a standard 52-card pack.

The dealer deals five cards to each player. The opponent player leads any card, playing it face up in the middle of the playing area.

For this program the following strategy will be used to determine which card the dealer will play:

> The dealer must play a card of the same suit if he can.
>
> He plays the lowest card in that suit that is of a higher rank than the card the opponent played.
>
> If he does not have such a card, he plays his lowest card in that suit.
>
> If he does not have a card in that suit, he plays the lowest ranking card regardless of suit.
>
> If there is a tie, he plays the card in the following order: clubs, diamonds, hearts and spades.

INPUT: There will be 5 lines of input. Each line will contain the opponent's lead card and the 5 cards held by the dealer. All cards will be represented by 2-character strings in value-suit order. AH represents the ace of hearts. K, Q and J and T will be used for king, queen, jack and 10 respectively. Note that the ace in this game has the lowest rank.

OUTPUT: For each input line print the card the dealer must play according to the strategy listed above.

SAMPLE INPUT	**SAMPLE OUTPUT**
1. 5D, 2D, 6H, 9D, TD, 6H	1. 9D
2. TC, AC, KC, QH, JS, TD	2. KC
3. 3D, 4H, 5C, 6S, 2D, 7H	3. 2D
4. KS, TH, QC, 3D, 9H, 3H	4. 3D
5. AC, AD, KH, AS, KS, QS	5. AD

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

public class Agram {
	static String suits = "CDHS";
	static ArrayList<String> possibleWinners = new ArrayList<String>();
	public static void main(String[] args) {
		Scanner console = new Scanner(System.in);

		for(int i = 1; i <= 6; i++) {
			System.out.print("input "+i+": ");
			possibleWinners.add(console.next());
		}
		String compareSuit = possibleWinners.get(0).substring(1);
		System.out.println();
		int largestIndex = 0;
		int numPossible = 0;
		int compareValue = assignValue(possibleWinners.get(0));
			numPossible = markPossible(compareSuit);
			if(numPossible > 0) {
				largestIndex = findWinner(compareValue);
				clearMarks();
			}
			else {
				markAll();
				largestIndex = findWinner(0);
			}
		clearMarks();
		System.out.println(possibleWinners.get(largestIndex));
		console.close();

	}
	public static int markPossible(String compareSuit) {
		int numPossible = 0;
		for(int i = 1; i < 6; i++) {
			String temp = possibleWinners.get(i);
			if(temp.substring(1).contentEquals(compareSuit)) {
				possibleWinners.set(i, possibleWinners.get(i)
+"W");
				numPossible++;
			}
		}
		return numPossible;
	}
	public static void markAll() {
		for(int i = 0 ; i < possibleWinners.size(); i++) {
			possibleWinners.set(i, possibleWinners.get(i) +"W");
		}
	}
	public static int findWinner(int compareValue) {
		int successes = 0;
		int largestValue = 14;
			int largestIndex = 0;
			for(int i = 1; i < possibleWinners.size(); i++) {
				String temp = possibleWinners.get(i);
				int tempValue = assignValue(temp);
				if(temp.length() > 2) {
```

```
                                        if(tempValue > compareValue) {
                                                if(tempValue < largestValue) {
                                                        largestIndex = i;
                                                        largestValue =
tempValue;
                                                        successes++;
                                                }
                                                else if(tempValue ==
largestValue) {
                                                        if(suitValue(temp) >
suitValue(possibleWinners.get(largestIndex))) {
                                                                largestIndex =
i;
                                                        }
                                                }
                                        }
                                }
                        }
                        if(successes == 0) {
                                return findWinner(0);
                        }
                        return largestIndex;
        }
        public static int assignValue(String temp) {
                int tempValue = 0;
                if(temp.substring(0,1).equals("A")) {
                        tempValue = 1;
                }
                else if(temp.substring(0,1).equals("T")) {
                        tempValue =10;
                }
                else if(temp.substring(0,1).equals("J")) {
                        tempValue =11;
                }
                else if(temp.substring(0,1).equals("Q")) {
                        tempValue =12;
                }
                else if(temp.substring(0,1).equals("K")) {
                        tempValue =13;
                }
                else {
                        tempValue = Integer.parseInt(temp.substring(0,1));
                }
                return tempValue;
        }
        public static void clearMarks() {
                for(int i = 0; i < possibleWinners.size(); i++) {
                        String str = possibleWinners.get(i);
                        if(str.length() > 2) {
                                String temp = str.substring(0,2);
                                possibleWinners.set(i, temp);
                        }
                }
        }
        public static int suitValue(String str) {
                String temp = str.substring(1,2);
                if(temp.equals("C"))
```

```
                return 4;
            else if(temp.equals("D"))
                return 3;
            else if(temp.equals("H"))
                return 2;
            else if(temp.equals("S"))
                return 1;
            else
                return -1;
        }

}
```