Q1

i)

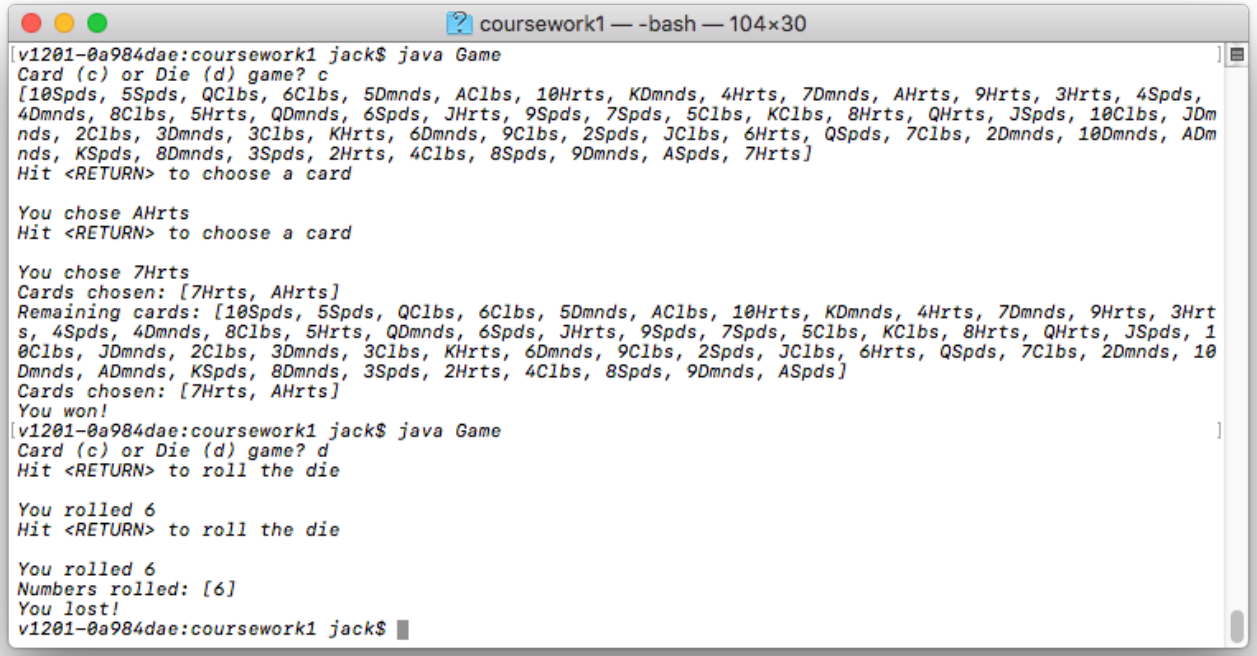
Changes made to Ex1SourceProg

LinearCongruentialGenerator.java

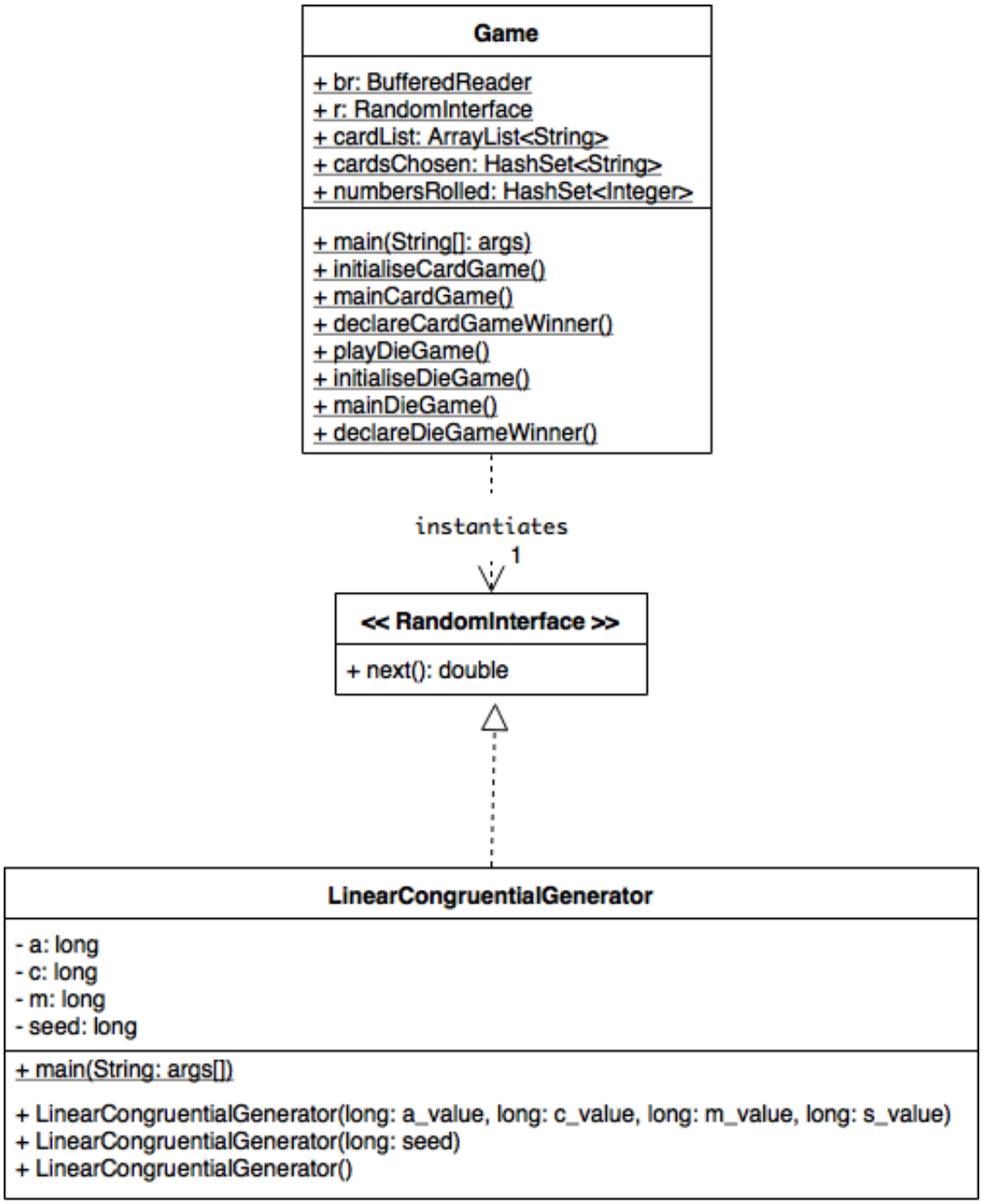
```
// From
public class LinearCongruentialGenerator implements IncompatibleRandomInterface {

// To
public class LinearCongruentialGenerator implements RandomInterface {
```

```
// Form
public static void main(String args[]) {
    IncompatibleRandomInterface r=new LinearCongruentialGenerator();
    for (int i=0; i<10; i++) System.out.println(r.getNextNumber());
}

// To
public static void main(String args[]) {
        RandomInterface r = new LinearCongruentialGenerator();
        for (int i = 0; i < 10; i++) {
            System.out.println(r.next());
        }
}
```

```
// From
public double getNextNumber() {
    seed = (a * seed + c) % m;
    return (double) seed/m;
}

// To
public double next() {
        seed = (a * seed + c) % m;
        return (double) seed / m;
}
```


coursework1 — -bash — 104×30
v1201-0a984dae:coursework1 jack$ java Game
Card (c) or Die (d) game? c
[10Spds, 5Spds, QClbs, 6Clbs, 5Dmnds, AClbs, 10Hrts, KDmnds, 4Hrts, 7Dmnds, AHrts, 9Hrts, 3Hrts, 4Spds, 4Dmnds, 8Clbs, 5Hrts, QDmnds, 6Spds, JHrts, 9Spds, 7Spds, 5Clbs, KClbs, 8Hrts, QHrts, JSpds, 10Clbs, JDmnds, 2Clbs, 3Dmnds, 3Clbs, KHrts, 6Dmnds, 9Clbs, 2Spds, JClbs, 6Hrts, QSpds, 7Clbs, 2Dmnds, 10Dmnds, ADmnds, KSpds, 8Dmnds, 3Spds, 2Hrts, 4Clbs, 8Spds, 9Dmnds, ASpds, 7Hrts]
Hit <RETURN> to choose a card
You chose AHrts
Hit <RETURN> to choose a card
You chose 7Hrts
Cards chosen: [7Hrts, AHrts]
Remaining cards: [10Spds, 5Spds, QClbs, 6Clbs, 5Dmnds, AClbs, 10Hrts, KDmnds, 4Hrts, 7Dmnds, 9Hrts, 3Hrts, 4Spds, 4Dmnds, 8Clbs, 5Hrts, QDmnds, 6Spds, JHrts, 9Spds, 7Spds, 5Clbs, KClbs, 8Hrts, QHrts, JSpds, 10Clbs, JDmnds, 2Clbs, 3Dmnds, 3Clbs, KHrts, 6Dmnds, 9Clbs, 2Spds, JClbs, 6Hrts, QSpds, 7Clbs, 2Dmnds, 10Dmnds, ADmnds, KSpds, 8Dmnds, 3Spds, 2Hrts, 4Clbs, 8Spds, 9Dmnds, ASpds]
Cards chosen: [7Hrts, AHrts]
You won!
v1201-0a984dae:coursework1 jack$ java Game
Card (c) or Die (d) game? d
Hit <RETURN> to roll the die
You rolled 6
Hit <RETURN> to roll the die
You rolled 6
Numbers rolled: [6]
You lost!
v1201-0a984dae:coursework1 jack$

ii)



iii)

The purpose of this Java program is to play 2 games (card game or die game). It takes a user input from the user to select the game and then runs through each game. The card game simulates the user selecting 2 cards and if one of them match 1 of 4 cards in a list the player wins. The die game simulates the user rolling a dice. If the user rolls a 1 they win. Each of the games uses the same method in LinearCongruentialGenerator (next()) to select the card or number rolled. In addition both games are called in a similar way. This includes a initialiser, main method and a method to declare the winner.

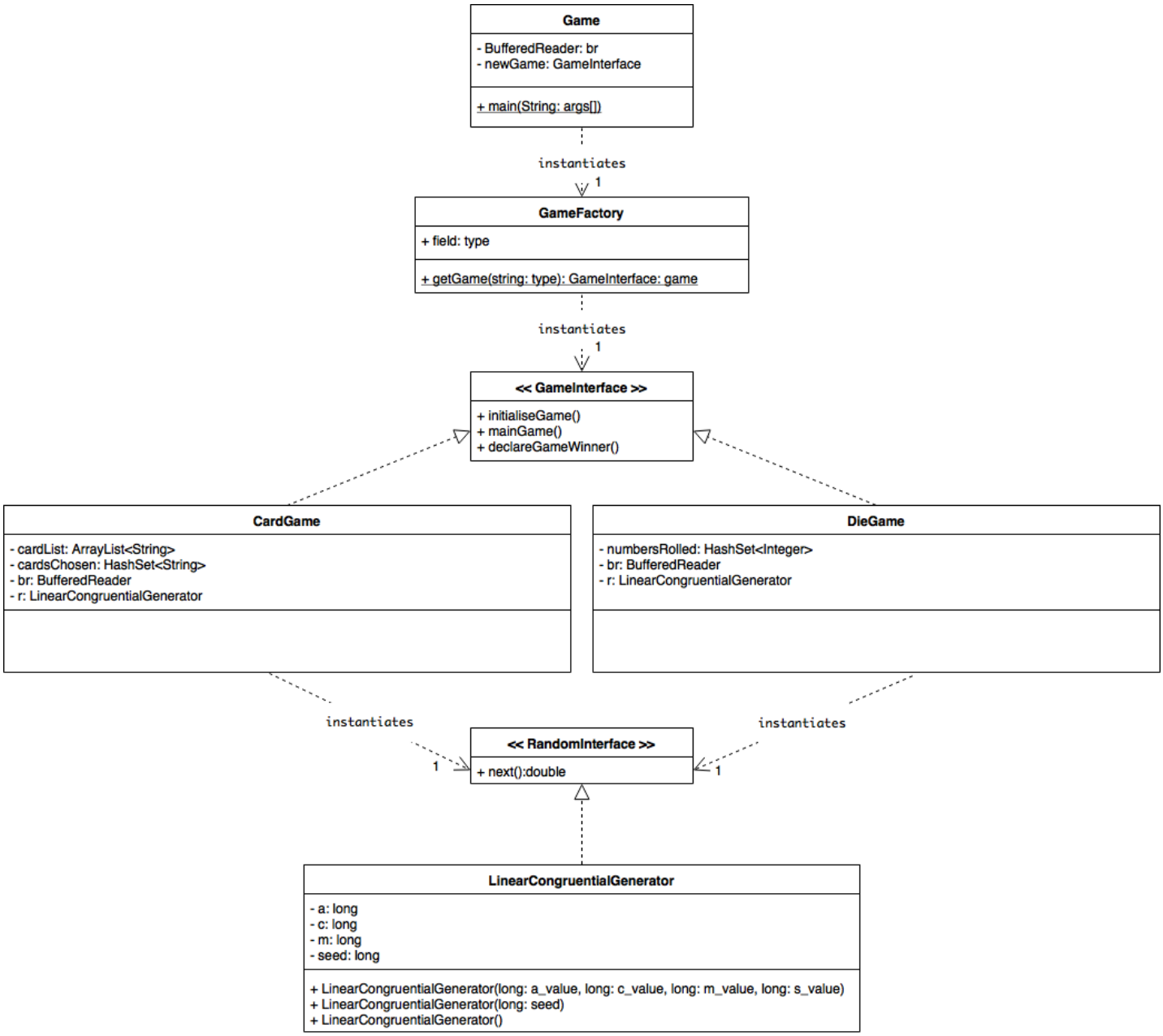
Q2

i)

Class / Interface	Explanation
Game	The game class will be the client in which the user will be able to interact with the program
GameFactory	GameFactory will call the GameInterface creating card game and dice game
GameInterface	The interface GameInterface will state the methods used for both CardGame and DieGame
CardGame	The card game class will create a new card game and return it
DieGame	The dice game class will create a new dice game and return it
LinearCongruentialGenerator	This class will generate a random number based on the current time
RandomInterface	The interface RandomInterface will state the methods used for random number generation classes

To improve the game program there will be 5 classes and 2 interfaces. The class Game will be what the user interacts with and is what they will use to start the application. From this the GameFactory class will call the process to create the game the user specified. GameInterface will be used to define a game. This is because both CardGame and DieGame have very similar functions and there is also the possibility of creating other similar games with the interface. The final class is LinearCongreantialGenerator. This class is called by each of the games and will return a number generated by the current time when the next() function is called. The next function is defined in the interface RandomInterface. This will allow a separate random class to be created implementing random number generation using a different method.

ii)



iii)

Game.java

```java
import java.io.*;

public class Game {

    public static void main(String[] args) throws Exception {

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        System.out.print("Card (c) or Die (d) game? ");
        String ans = br.readLine();

        GameInterface newGame = GameFactory.getGame(ans);

        if (newGame == null) {
            System.out.println("Input not understood"); // Game does not exist
        } else {
            try { // Game exists, starts game
                newGame.initialiseGame();
                newGame.mainGame();
                newGame.declareGameWinner();
            } catch (Exception e) {
                System.out.println("Unexpected error. See " + e + " for more");
            }
        }
    }
}
```

GameFactory.java

```java
public class GameFactory {

    public static GameInterface getGame(String type){

        if (type.equalsIgnoreCase("c")) {
            GameInterface game = new CardGame();
            return game;
        } else if (type.equalsIgnoreCase("d")) {
            GameInterface game = new DieGame();
            return game;
        }

        return null;
    }
}
```

GameInterface.java

```java
public interface GameInterface {

    public void mainGame() throws Exception;

    public void initialiseGame() throws Exception;

    public void declareGameWinner();

}
```

DieGame.java

```
import java.util.*;
import java.io.*;

public class DieGame implements GameInterface {

    private HashSet<Integer> numbersRolled = new HashSet<Integer>();

    private BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));

    private RandomInterface r = new LinearCongruentialGenerator();

    public void mainGame() throws Exception {

        for (int i = 0; i < 2; i++) {
            System.out.println("Hit <RETURN> to roll the die");
            br.readLine();
            int dieRoll = (int)(r.next() * 6) + 1;

            System.out.println("You rolled " + dieRoll);
            numbersRolled.add(new Integer(dieRoll));
        }

        // Display the numbers rolled
        System.out.println("Numbers rolled: " + numbersRolled);

    }

    public void initialiseGame() throws Exception {
        return;
    }

    public void declareGameWinner() {
        if (numbersRolled.contains(new Integer(1))) {
            System.out.println("You won!");
        }
        else System.out.println("You lost!");
    }
}
```

CardGame.java

```
import java.util.*;
import java.io.*;

public class CardGame implements GameInterface {
```

```
    private ArrayList<String> cardList;
    private HashSet<String> cardsChosen = new HashSet<String>();

    private BufferedReader br = new BufferedReader(
                                new InputStreamReader(System.in));

    private RandomInterface r = new LinearCongruentialGenerator();

    public void mainGame() throws Exception {

        for (int i = 0; i < 2; i++) {
            System.out.println("Hit <RETURN> to choose a card");
            br.readLine();

            int cardChoice = (int) (r.next() * cardList.size());
            System.out.println("You chose " + cardList.get(cardChoice));
            cardsChosen.add(cardList.remove(cardChoice));
        }

        // Display the cards chosen and remaining cards
        System.out.println("Cards chosen: " + cardsChosen);
        System.out.println("Remaining cards: " + cardList);

    }

    public void initialiseGame() throws Exception {
        // The initialisation phase:

        // Create a list of cards ... and shuffle them
        String cards[] = {"AHrts", "2Hrts", "3Hrts", "4Hrts", "5Hrts", "6Hrts",
                          "7Hrts", "8Hrts", "9Hrts", "10Hrts", "JHrts",
                          "QHrts", "KHrts",
                          "ADmnds", "2Dmnds", "3Dmnds", "4Dmnds", "5Dmnds",
                          "6Dmnds", "7Dmnds", "8Dmnds", "9Dmnds", "10Dmnds",
                          "JDmnds", "QDmnds", "KDmnds",
                          "ASpds", "2Spds", "3Spds", "4Spds", "5Spds", "6Spds",
                          "7Spds", "8Spds", "9Spds", "10Spds", "JSpds",
                          "QSpds", "KSpds",
                          "AClbs", "2Clbs", "3Clbs", "4Clbs", "5Clbs", "6Clbs",
                          "7Clbs", "8Clbs", "9Clbs", "10Clbs", "JClbs",
                          "QClbs", "KClbs"};
        cardList = new ArrayList<String> (Arrays.asList(cards));
        // Taking advantage of "generics" to tell the compiler all the elements
will be Strings

        // Shuffle them
        for (int i = 0; i < 100; i++) {
```

```
            // choose two random cards at random and swap them, 100 times
            int firstIndex = ((int) (r.next() * 52));
            int secondIndex = ((int) (r.next() * 52));
            String temp = (String) cardList.get(firstIndex);
            cardList.set(firstIndex, cardList.get(secondIndex));
            cardList.set(secondIndex, temp);
        }

        // Print out the result
        System.out.println(cardList);
    }

    public void declareGameWinner() {
        System.out.println("Cards chosen: " + cardsChosen);
        if (cardsChosen.contains("AHrts") || cardsChosen.contains("ADmnds") ||
            cardsChosen.contains("ASpds") || cardsChosen.contains("AClbs")) {
            System.out.println("You won!");
        }
        else System.out.println("You lost!");
    }
}
```

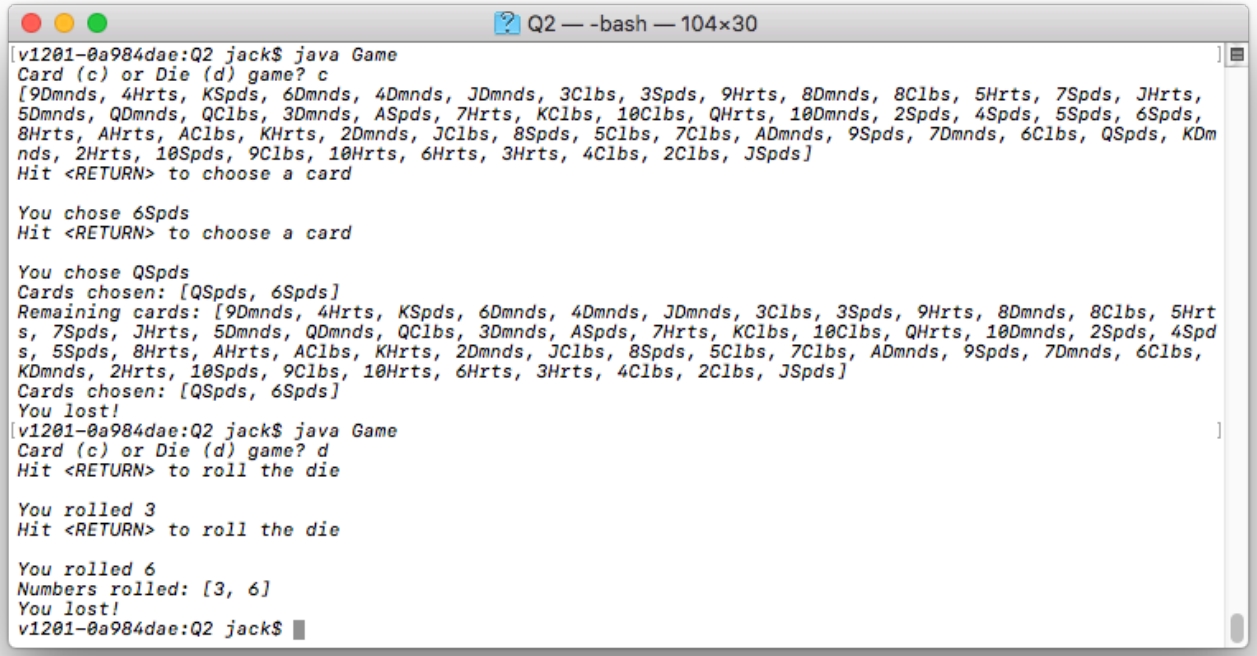
RandomInterface.java

```
public interface RandomInterface {
    // Simply defines a method for retrieving the next random number
    public double next();
}
```

LinearCongruentialGenerator.java

```
public class LinearCongruentialGenerator implements RandomInterface {
    // Generates pseudo-random numbers using:
    // X(n+1) = (aX(n) + c) (mod m)
    // for suitable a, c and m. The numbers are "normalised" to the range
    // [0, 1) by computing X(n+1) / m.

    private long a, c, m, seed;
    // Need to be long in order to hold typical values ...

    public LinearCongruentialGenerator(long a_value, long c_value, long m_value,
long s_value) {
        a = a_value;
        c = c_value;
        m = m_value;
        seed = s_value;
```

```
    }

    public LinearCongruentialGenerator(long seed) {
        // Set a, c and m to values suggested in Press, Teukolsky, et al.,
"Numberical Recipies"
        this(1664525, 1013904223, 4294967296l, seed);
        // NB "l" on the end is the way that a long integer can be specified. The
        // smaller ones are type-cast silently to longs, but the large number is
too
        // big to fit into an ordinary int, so needs to be defined explicitly
    }

    public LinearCongruentialGenerator() {
        // (Re-)set seed to an arbitrary value, having first constructed the
object using
        // zero as the seed. The point is that we don't know what m is until
after it has
        // been initialised.

        this(0);
        seed = System.currentTimeMillis() % m;

    }

    public static void main(String args[]) {
        // Just a little bit of test code, to illustrate use of this class.
        RandomInterface r = new LinearCongruentialGenerator();
        for (int i = 0; i < 10; i++) {
            System.out.println(r.next());
        }
        // Since RandomInterface doesn't know about the instance variables
defined in this
        // particular implementation, LinearCongruentialGenerator, we need to
type-cast
        // in order to print out the parameters (primarily for "debugging"
purposes).

        LinearCongruentialGenerator temp = (LinearCongruentialGenerator) r;
        System.out.println("a: " + temp.a + "  c: " + temp.c + "  m: " + temp.m +
"  seed: " + temp.seed);

    }

    public double next() {
        seed = (a * seed + c) % m;
        return (double) seed / m;
    }
}
```


Q2 — -bash — 104×30
v1201-0a984dae:Q2 jack$ java Game
Card (c) or Die (d) game? c
[9Dmnds, 4Hrts, KSpds, 6Dmnds, 4Dmnds, JDmnds, 3Clbs, 3Spds, 9Hrts, 8Dmnds, 8Clbs, 5Hrts, 7Spds, JHrts, 5Dmnds, QDmnds, QClbs, 3Dmnds, ASpds, 7Hrts, KClbs, 10Clbs, QHrts, 10Dmnds, 2Spds, 4Spds, 5Spds, 6Spds, 8Hrts, AHrts, AClbs, KHrts, 2Dmnds, JClbs, 8Spds, 5Clbs, 7Clbs, ADmnds, 9Spds, 7Dmnds, 6Clbs, QSpds, KDmnds, 2Hrts, 10Spds, 9Clbs, 10Hrts, 6Hrts, 3Hrts, 4Clbs, 2Clbs, JSpds]
Hit <RETURN> to choose a card
You chose 6Spds
Hit <RETURN> to choose a card
You chose QSpds
Cards chosen: [QSpds, 6Spds]
Remaining cards: [9Dmnds, 4Hrts, KSpds, 6Dmnds, 4Dmnds, JDmnds, 3Clbs, 3Spds, 9Hrts, 8Dmnds, 8Clbs, 5Hrts, 7Spds, JHrts, 5Dmnds, QDmnds, QClbs, 3Dmnds, ASpds, 7Hrts, KClbs, 10Clbs, QHrts, 10Dmnds, 2Spds, 4Spds, 5Spds, 8Hrts, AHrts, AClbs, KHrts, 2Dmnds, JClbs, 8Spds, 5Clbs, 7Clbs, ADmnds, 9Spds, 7Dmnds, 6Clbs, KDmnds, 2Hrts, 10Spds, 9Clbs, 10Hrts, 6Hrts, 3Hrts, 4Clbs, 2Clbs, JSpds]
Cards chosen: [QSpds, 6Spds]
You lost!
v1201-0a984dae:Q2 jack$ java Game
Card (c) or Die (d) game? d
Hit <RETURN> to roll the die
You rolled 3
Hit <RETURN> to roll the die
You rolled 6
Numbers rolled: [3, 6]
You lost!
v1201-0a984dae:Q2 jack$