Exam #1 Solutions

1 Short Answer (5 Points Each)

1. What is the difference between a compiler and an interpreter? Also, discuss Java's method.

   Solution: A compiler will take a program written in a high-level language, translate it into machine language and then save the machine language program to a file that can be run on the computer. An interpreter does essentially the same thing except that it translates the high-level language to machine language one command at a time and does not save the machine language program to a file. Java uses a combination of the two. There is a compile stage that translates the Java code into byte-code that the interpreter (known as the JVM or Java Virtual Machine) runs.

2. Java is a "platform-independent language." What is a *platform*, what does *platform-independent* mean, and how does Java attain its platform independence?

   Solution: A platform is an operating system, so platform-independent means that the same program can be run on any operating system. Java is compiled into byte-code, this byte code is then interpreted by the Java Virtual Machine (JVM). There is a JVM built for every common operating system, so Java byte-code can be run on any operating system.

3. What are the three types of programming errors? Briefly describe each of them.

   Solution:

   Syntax Error: An error in the program code due to misuse of the programming language.

   Run-time Error: An error that occurs during a run of the program which usually causes the program to terminate prematurely.

   Logic Error: This error occurs when the program is syntactically correct and there are no runtime errors but the program does not do what it was intended to do.

4. What are reserved words? Give four examples of Java reserved words.

   Solution: A reserved word is a word that is used for a particular use in the programming language and cannot be redefined. Hence the programmer cannot use a reserved word as a variable name. There are many reserved words in Java, some we have seen thus far are public, void, class, if, else, while, int, double, long, float, new, import, and static.

5. Write a single line of code that declares an integer variable `num1` and assigns to it a random integer between 15 and 27, inclusively.

   Solution:

   ```
   int num1 = (int) (Math.random() * 13) + 15;
   ```

6. Write a few lines of code that will take a string stored in `str1`, extract the first word and then convert that word to all uppercase characters and finally print the word out to the screen.

   Solution:

   ```
   str1 = str1.trim();
   int space = str1.indexOf(" ");
   String word = str1.substring(0, space);
   word = word.toUpperCase();
   System.out.println(word);
   ```

7. Answer the following questions about numeric data types in Java.

   (a) What happens when you overload an int?
   Solution: The value cycles around to the minimum value of an int.

   (b) What happens when you overload a double?
   Solution: The value turns into Infinity.

(c) What happens when you underload an int?
Solution: The value cycles around to the maximum value of an int.

(d) What happens when you underload a double?
Solution: The value turns into 0.

(e) What happens when you input an integer when the Scanner is doing a nextDouble?
Solution: The value is stored as a double data type and has the same value as the input integer.

2 Program Traces (15 Points Each)

1. For each of the given inputs, write the output of the program.

```java
import java.util.Scanner;

public class Exam01_Trace1 {

    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        System.out.print("Input n: ");
        int n = keyboard.nextInt();

        while (n > 1) {
            if (n % 2 == 0) {
                n = n / 2;
            } else {
                n = 3 * n + 1;
            }
            System.out.print(n + " ");
        }
    }
}
```

(a) `Input n: 5`
Solution:

```
16 8 4 2 1
```

(b) `Input n: 7`
Solution:

```
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
```

(c) `Input n: 20`
Solution:

```
10 5 16 8 4 2 1
```

2. For each of the given inputs, write the output of the program.

```java
import java.util.Scanner;

public class Exam01_Trace2 {

    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        System.out.print("Input str1: ");
        String str1 = keyboard.nextLine();
        System.out.print("Input str2: ");
        String str2 = keyboard.nextLine();

        int pos = str1.indexOf(str2);
        System.out.println(pos);

        if (pos <= 5) {
            str1 = str1.replaceAll(str2, "HERE");
            str2 = "Replaced All";
        } else if (pos > 15) {
            str2 = str1.replaceAll(" ", "");
            str1 = str2.substring(str2.length() / 4, str2.length() / 2);
        } else {
            String tempstr = str1;
            str1 = str2;
            str2 = tempstr;
        }

        System.out.println(str1);
        System.out.println(str2);
    }
}
```

(a)
```
Input str1: This is a string for testing Exam 1 Trace 2.
Input str2: a
```
Solution:

```
8
a
This is a string for testing Exam 1 Trace 2.
```

(b)
```
Input str1: This is a string for testing Exam 1 Trace 2.
Input str2: for
```
Solution:

```
17
tringfort
ThisisastringfortestingExam1Trace2.
```

(c)
```
Input str1: This is a string for testing Exam 1 Trace 2.
Input str2: is
```
Solution:

```
2
ThHERE HERE a string for testing Exam 1 Trace 2.
Replaced All
```

3 Coding (15 Points Each)

1. Write a program that helps the cashier at a store count back change. The program should ask for the amount of money to give back in change for a purchase (for example, $5.47) and then output the number of dollars, quarters, dimes, nickels, and pennies. You do not need to go higher than a dollar bill in change but the program should handle amounts like $125.24. The output should always use the highest denomination possible, an output of 12524 pennies does not solve the problem. For example, for an input of $5.47, the program should output 5 ones, 1 quarter, 2 dimes and 2 pennies. For the input of $125.24, the program should output 125 ones, 2 dimes and 4 pennies.

```java
import java.util.Scanner;

public class Exam01_1 {

    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        System.out.print("Amount to change: ");
        double money = keyboard.nextDouble();
        long cents = Math.round(money * 100);

        long ones = cents / 100;
        cents = cents % 100;
        long quarters = cents / 25;
        cents = cents % 25;
        long dimes = cents / 10;
        cents = cents % 10;
        long nickels = cents / 5;
        cents = cents % 5;
        long pennies = cents;

        System.out.println("Change");
        System.out.println("Ones: " + ones);
        System.out.println("Quarters: " + quarters);
        System.out.println("Dimes: " + dimes);
        System.out.println("Nickels: " + nickels);
        System.out.println("Pennies: " + pennies);
    }
}
```

2. In a standard poker deck of cards, each card has a face value and a suit. The face values are A (Ace), 2, 3, 4, 5, 6, 7, 8, 9, 10, J (Jack), Q (Queen), K (King) and the suits are ♢ (Diamonds), ♡ (Hearts), ♣ (Clubs) and ♠ (Spades).

   Write a program that will "deal" one card at random and output the value of the card, using the name of the suit and the name of the face value as either Ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, or King. Five runs of the program are listed below.

```
Card: 6 of Clubs
Card: Ace of Spades
Card: King of Clubs
Card: Queen of Clubs
Card: 10 of Diamonds
```

   Note that we started the program with the string variable `cardstr` set to the empty string and the final line does all of the output. Use conditional statements and string concatenation to transform `cardstr` into the final output string.

```java
public class Exam01_2 {

    public static void main(String[] args) {
        String cardstr = "";

        int face = (int) (Math.random() * 13) + 1;
        int suit = (int) (Math.random() * 4) + 1;

        if (face == 1) {
            cardstr += "Ace";
        } else if (face == 11) {
            cardstr += "Jack";
        } else if (face == 12) {
            cardstr += "Queen";
        } else if (face == 13) {
            cardstr += "King";
        } else {
            cardstr += face;
        }

        cardstr += " of ";

        if (suit == 1) {
            cardstr += "Diamonds";
        } else if (suit == 2) {
            cardstr += "Hearts";
        } else if (suit == 3) {
            cardstr += "Clubs";
        } else if (suit == 4) {
            cardstr += "Spades";
        }

        System.out.println("Card: " + cardstr);
    }
}
```

3. The factorial of a positive integer n is defined to be $n! = n \cdot (n-1) \cdot (n-2) \cdots 2 \cdot 1$, also we define $0! = 1$. Write a program that will allow the user to input an integer n and then the program should output the value $n!$. If the user inputs a negative number the program should print out an error message saying that the input was invalid. Note that factorials are integers, not decimal numbers, your program should use the appropriate data type.

```java
import java.util.Scanner;

public class Exam01_3 {

    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        System.out.print("n = ");
        int n = keyboard.nextInt();
        long fact = 1;

        if (n < 0) {
            System.out.print("Invalid input!");
        } else if (n == 0) {
            System.out.print("n! = 1");
        } else {
            while (n > 1) {
                fact *= n;
                n--;
            }
            System.out.print("n! = " + fact);
        }
    }
}
```