German University in Cairo
Media Engineering and Technology
Prof. Dr. Slim Abdennadher
Dr. Wael Abouelsaadat
Dr. Mohammed Abdel Megeed

Introduction to Computer Programming, Spring Term 2017
Practice Assignment 1

Discussion: 18.2.17 - 23.2.17

Exercise 1-1 Java Expressions
To be discussed in the Tutorial

a) $-(-1)$

Solution:

Syntactically correct, value 1.

b) $--1$

Solution:

Syntactically incorrect. "$--$" is an operator in Java that only works on variables. Correct uses of this operator (and the related "++" operator) are: "++i", "$--$i", "i++", and "i$--$".

c) $2*4(-6)$

Solution:

Syntactically incorrect.
There is an operator missing between `4` and `(-6)`.

d) $(12)*(4)-6$

Solution:

Syntactically correct, value `42`.

e) $29\%5\%3*4-2$

Solution:

Syntactically correct, value `2`.

f) $29\%5\%3-4*2$

Solution:

Syntactically correct, value `-7`.

g) 7%3 = 1

Solution:

Syntactically Incorrect.
The assignment operator = requires a variable on the left side. Danger of confusion with the equality operator ==.

h) 1 / 0

Solution:

Syntactically correct, runtime error.

i) 1 / 0.0

Solution:

Syntactically correct, value infinity.

j) ((10 − 2) ∗ 4%)8)

Solution:

Syntactically incorrect ,''')'' expected.

k)
```
int x = 2;
int y = 3;
int z = 4;
int v = x | y & z;
```

Solution:

Syntactically correct, v = 2. A declaration is also an expression in Java (highest precedence).

l)
```
int x = 9;
int y = 0;
boolean z = (x<10 | x/y == 0);
System.out.print ("Success! The value is " + z );
```

Solution:

Syntactically correct, but due to the eager evaluation behavior of the bitwise 'l' operator, both expressions will be evaluated, resulting in an ArithmeticException.

m)
```
int x = 9;
int y = 0;
boolean z = (x<10 || x/y == 0);
System.out.print ("Success! The value is " + z );
```

Solution:

Syntactically correct. Due to the short-circuiting behavior of the logical 'll' operator, only the first expression will be evaluated, the condition as a whole will be satisfied, and Success! will be printed.

n)
```
float x = 2.5;
float y = 3.4;
float z = x + y;
System.out.print ("The value is " + z );
```

Solution:

Syntactically incorrect. By default in Java, a floating point number is represented as **double** with 64-bits. Thus, we can not put a **double** in **float** without typecasting.

o)
```
int x = 34567456;
long y = 1234567;
long z = x + y;
System.out.println("The value is " + z );
y = 46547864784282;
z = x + y;
System.out.print ("The value is " + z );
```

Solution:

Syntactically incorrect. By default in Java, an integer number is represented as **int** with 32-bits. If we want to store a **long** integer that exceeds the range of the **int**, we need to typecast it.

p)
```
byte x = 12;
byte y = 8;
byte z = x + y;
System.out.print ("The value is " + z );
```

Solution:

Syntactically incorrect. The addition of `x` and `y` may exceed the range of the **byte**. Thus, we need to typecast it into **byte** as well.

q)
```
String x = "CSEN";
int y = 202;
String z = x + y + "!";
System.out.println( z );
String u = y + x + "!";
System.out.println( u );
String v = y;
System.out.println( v );
String m = "204";
int o = m;
System.out.println( o );
```

Solution:

Syntactically incorrect. **int** can not be converted into `String`, we need to concatenate it to a `String`. Also, `String` can not be converted into **int** directly, we need to parse it.

Exercise 1-2 Errors and Conventions
To be discussed in the lab

In the following lab exercises, follow the instructions carefully and observe the outcome. In each case you should be able to explain why the output is as it is. You are encouraged to check out the useful links on MET Website `http://met.guc.edu.eg/Courses/Links.aspx?crsEdId=480`

For each of the .java classes in the " Lab 1 Exercises.zip " zipped folder, compile the files on JCreator. You are asked to define in each file the lines which have the following faults:

a) Syntax Errors

b) Runtime Errors

c) Logic Errors

Furthermore, specifically define the errors. For example, the following line of code:

```
String Word = "Hello World!";
system.out.print(Word)
```

has two Syntax Errors and 1 bad identifier naming. The correct solution is:

```
String word = "Hello World!";
System.out.println(word);
```

Exercise 1-3 Number Precision, Exactness
To be discussed in the Lab

Given the snippet of code below, what will be the values of d3 and d4 that will be printed?

```
double d1 = 1.03;
double d2 = 0.42;
double d3 = d1 - d2;
System.out.println(d3);

double d4 = 0.1;
float f   = (float) d4;
d4        = 1 - f;
System.out.println(d4);
```

Solution:

d3 will equal 0.6100000000000001, and d4 will equal 0.8999999761581421.

Exercise 1-4 Picking a random card
To be discussed in the lab

Given a deck of 1000 cards, you would like to pick the n^th card where n is a random number.

You should try out the following two methods:

- `Math.random()` as discussed in the lecture.

  Solution:

```
import java.util.*;
public class PA1
{
    public static void main(String[]args)
    {
        int x = (int) ((Math.random()*1000) +1);
        System.out.println(x);
    }
}
```

- `nextInt(int x)` where x is the upper bound of the range (exclusive). **Hint**: it is used with a `Random` type variable.

  Solution:

```
import java.util.*;
public class PA1
{
    public static void main(String[]args)
    {
        Random randGen = new Random();
        int randomAng = randGen.nextInt(1000);
        System.out.println(randomAng);
    }
}
```

Exercise 1-5 Supermarket Change

Write a Java program CountChange to count change. Given the number of quarter, dimes, nickles, and pennies the program should output the total as a single value in dollars and pennies.

Hint:

- One dollar corresponds to 100 pennies.
- One quarter corresponds to 25 pennies.
- One dime corresponds to 10 pennies.
- One nickle corresponds to 5 pennies.

For example if we have: 3 quarters, 2 dimes, 1 nickle and 6 pennies, then the total is 1.06 dollars.

Solution:

```
public class CountChange
{
    public static void main(String [] args) {
        int quarters, dimes, nickles, pennies;
        quarters = 3;
        dimes    = 2;
        nickles  = 1;
        pennies  = 6;
        double total = quarters * 25 + dimes * 10 + nickles * 5 + pennies;
        System.out.println("Total is: $" + total/100);
    }
}
```

Exercise 1-6 Cook in a Hurry

You want to cook some pasta, but you are short of time. To be sure you can finish cooking before your next appointment, you need to know how long it takes for the water to boil.

At its highest setting, your stove needs two minutes per liter ($1l = \frac{1}{1{,}000}m^3$) to reach the boiling point. You use a cylindric pot.

Write a program that, given the diameter of the pot and the hight of the water in it, calculates the the time needed for the water to boil.

Hint. The Volume of a cilinder is $\pi \times r^2 \times h$.

Solution:

```
public class CookSomePasta {
    public static void main (String[] args) {
        final double PI = 3.1415;
        double diameter = 7.0;
        double height   = 6.0;
        double radius   = diameter / 2;

        // calculate the volume
        double volume   = PI * radius * radius * height;

        // Assume that diameter & height are given in centimeters
        double liters   = volume / 1000;

        // The stove needs two minutes per liter
        double time     = liters * 2;

        System.out.print("Stove needs " + time + " minutes.");
    }
}
```

```
import java.util.*;
public class CookSomePasta {
    public static void main (String[] args) {
        final double PI = 3.1415;
        Scanner sc      = new Scanner(System.in);

        System.out.print("Enter diameter:");
        double diameter = sc.nextDouble();

        System.out.print("Enter height:");
        double height   = sc.nextDouble();

        double radius   = diameter / 2;

        // calculate the volume
        double volume   = PI * radius * radius * height;

        // Assume that diameter & height are given in centimeters
        double liters   = volume / 1000;

        // The stove needs two minutes per liter
        double time     = liters * 2;

        System.out.print("Stove needs " + time + " minutes.");
    }
}
```

Exercise 1-7 Electrical Resistance

The equivalent resistance of resistors connected in series is calculated by adding the resistances of the individual resistors.
The formula for resistors connected in parallel is a little more complex. Given two resistors with resistances R1 and R2 connected in parallel the equivalent resistance is given by the inverse of the sum of the inverses; e.g. $\frac{1}{Req} = \frac{1}{R1} + \frac{1}{R2}$.
Write a program that given 3 resistances outputs the equivalent resistance when they are connected in series and when they are connected in parallel.

Solution:

```
public class Resistance
{
    public static void main (String[] args) throws IOException
    {
        float r1 = 8, r2 = 8, r3 = 4;
        float resS    = r1 + r2 + r3;
        float resPInv = 1/r1 + 1/r2 + 1/r3;
        float resP    = 1 / resPInv;

        System.out.println("Series resistance:" + resS);
        System.out.println("Parallel resistance:"  + resP);
    }
}
```

Exercise 1-8 Temperature Conversion

Assume that you have a Celsius scale temperature of 100 degrees and you wish to convert it into degrees on the

Fahrenheit scale and the Kelvin scale. This conversion is done using the following formulas.

$$T_f = \frac{9}{5} \times T_c + 32, \textit{ and}$$
$$T_k = T_c + 273.$$

Where T_f is temperature in degrees Fahrenheit, T_c is temperature in degrees Celsius, and T_k is temperature in degrees Kelvin.

Write a Java program that takes as an input a degree in Celsius scale and converts it into both Fahrenheit and Kelvin scales.

Solution:

```
import java.util.*;
public class TemperatureCalculator {
    public static void main (String [] args) {
        Scanner sc = new Scanner(System.in);
        double celsius, fahrenheit, kelvin;

        System.out.print("Please␣insert␣the␣temperature␣in␣degrees␣Celsius␣");
        celsius    = sc.nextDouble();
        fahrenheit = (9.0/5 * celsius) + 32;
        kelvin     = celsius + 237 ;

        System.out.println("␣Temperature␣in␣degrees␣Fahrenheit␣is␣" + fahrenheit);
        System.out.println("␣Temperature␣in␣degrees␣Kelvin␣is␣" + kelvin);
    }
}
```

Exercise 1-9 Octal Conversion

Assume that you have an octal number (base 8) and you would like to convert it into its equivalent decimal number (base 10) and binary number (base 2). For example: $(27)_8$ is equal to $(23)_{10}$ in decimal and $(10111)_2$ in binary.

Write a sequential Java program to convert a two digit octal number into its equivalent binary and decimal numbers.

Solution:

```
import java.util.*;
public class Octal
{
    public static void main (String[] args) {
        int oct0, oct1;
        int q0, q1, q2, r0, r1, r2;
        Scanner sc = new Scanner(System.in);
        System.out.print("Please␣enter␣the␣octal␣number␣:");
        int octal = sc.nextInt();

        //extracting the digits
        oct0 = octal%10;
        oct1 = octal/10;

        // Calculate the decimal Value
        double dec = oct0 * 1 + oct1 * 8;
        int result = (int) dec;
        System.out.println(octal + "␣in␣octal␣=␣" + result + "␣in␣decimal.");
```

```
        //converting the first digit base 8 into 3 bits
        q0 = oct1/2;
        r0 = oct1%2;
        q1 = q0/2;
        r1 = q0%2;
        q2 = q1/2;
        r2 = q1%2;
        System.out.print(octal + " in octal = "+ r2 +" "+ r1 +" "+ r0 +" ");

        //converting the second digit base 8 into 3 bits
        q0 = oct0/2;
        r0 = oct0%2;
        q1 = q0/2;
        r1 = q0%2;
        q2 = q1/2;
        r2 = q1%2;
        System.out.println(r2 +" "+ r1 +" "+ r0 +" in binary.");
    }
}
```