IB Computer Science

HW Pset 01

Java Programming

Student________________

Date: __________________

Homework:

1. Read pages 69-76 in the AP Computer Science book, Control Structures and the Chapter Summary
2. Answer questions 1-5 pages 77-78. Compile and execute any program segments that you need to in order to understand the question and the answer. Answers begin on page 88.
3. Answer questions 9-10 pages 79. Compile and execute any program segments that you need to in order to understand the question and the answer. Answers are on page 89.

Question 1

```
// Name: Mr. Brennan
// File: HWPset01.java
// Purpose: Demonstrate the solutions to Problem Set 1, Question 1

import java.util.Scanner;

public class HWPset01 {

   public static void one_I() {
      double x = 14.7;
      int    y = x;    // trying to convert a double to an integer
      }

   public static void one_II() {
      double x = 14.7;
      int    y = (int) x;    // casting a double to an integer
      }

   public static void one_III() {
      int    x = 14;
      double y = x;    // trying to convert an integer to a double
      }

   public static void main(String[] args)     {
      } // main

} // class HWPset01
```

Test

```
javac HWPset01.java
HWPset01.java:11: error: possible loss of precision
        int    y = x;    // trying to convert a double to an integer
               ^
  required: int
  found:    double
1 error
```

Explanation

A compile error occurs in one_I() when trying to convert from a double to an integer. The value of x is 14.7, if it is intended that the value of x become 14, then a type cast must be used like in method one_II(). No loss of data will occur when converting from an integer to a double, so nothing is wrong with one_III().

Question 2

```
System.out.print("\\* This is not \n a comment *\\");
```

Explanation

To print a \ in a print statement, it must be escaped. Generally a character following a \ will have special meaning. The \n will force the continuation on a new line.
So: \\* This is not \n a comment *\\ will print:

```
\* Thus us not
 a comment *\
```

Question 3

```
if (n != 0 && x / n > 100)
    statement1; // do something, or nothing
else
    statement2;  // do something or nothing
```

Explanation

This question is really asking if the program will have a run time error if n = 0, and you try to divide x by 0. Notice that this is a logical ***and*** operation. Both parts of the condition must be true for the "if true" block to be executed. That means that both

`n != 0 must be true, and      x / n > 100 has to be true.`

If `n != 0` is false, then there is no way for the whole if statement to be true, so statement2 will be execute. When the java interpreter evaluates only enough of the statement to determine its "true" this is called **short circuit evaluation,** java stops evaluating the rest of the expression, so no run time error will occur.

Note: If the statement read as

```
if (n != 0 && x / n > 100)
```

and n was equal to 0, then the first expression would be evaluated to false, causing the second expression to be evaluated – which would generate a run time error because x/0 will cause a run time error.

Question 4

Consider the following program statements:

```
 double answer0 = 13/5;
System.out.println("13 / 5 = " + answer0);

double answer1 = 13/5.0;
System.out.println("13 / 5.0 = " + answer1);

 double answer2 = 13.0/5;
System.out.println("13.0 / 5 = " + answer2);
```

```
The output is
13 / 5 = 2.0
13 / 5.0 = 2.6
13.0 / 5 = 2.6
```

Explanation

13/5 is dividing an integer by an integer – which will result with an integer (2). When the value of 2 is placed into variable answer0 which is declared above as type double it will be assigned 2.0

Question 5

```
  int result = 13 - 3 * 6 / 4 % 3;
```

Explanation

This is an order of operations problem, it might help to put in parenthesis to show exactly what should happen:

*,/, and % all have the same precedence, so they will be performed first, from left to right:

```
    int result = 13 - (((3 * 6) / 4) % 3);

Calculate result = 13 - (((18) / 4) % 3);
          result = 13 - ((4) % 3);
          result = 13 - (1);
          result = 12;
```

Question 9

This is a program trace question, format it the way you want:

```
      int num = 22; // we were given it
      if (num > 0)  // which it is, because it is 22
         if (num % 5 == 0) // which it is not, 22 % 5 is 2
             System.out.println(num); // will not be executed.
         else
             System.out.println(num + " is negative");
             // this statement will be executed, even though
             // n is 22.
```

Output:

```
22 is negative
This segment could be fixed using { } to get the last else
statement to correspond to the if (num > 0) test.
```

Question 10

This is a program tracing question – create a trace table:

Line	Statement	x	y
1	int x = 30, y = 40;	30	40
2	if (x >= 0) { // which it is	30	40
3	if (x <= 100) { // which it is	30	40
4	y = x * 3;	30	90
5	if (y < 50) // which it is not	30	90
6	x /= 10; // not executed	30	90
7	}	30	90
8	else // not going to be executed because x<=100	30	90
9	y = x * 2; // not executed	30	90
10	} // end if (x >= 0) which it was	30	90

11	else // not going to be executed
	y = -x;
		30	90

Final Results		30	90