CS 401 MIDTERM REVIEW: SAMPLE PROBLEM SET #1

Q#1 Two pieces of code are said to be "execution equivalent" if they each produce the same output. Determine if the following pairs of code fragments are execution equivalent.

Code Pair A: **NOT EQUIVALENT**

```
for( int i=0 ; i<10 ; i++)
{
      System.out.println(i);
}
```

```
int i=0;
while( i<10 )
{
      i++; // 0 NOT PRINTED
      System.out.println(i);
}
```

Code Pair B: **ARE EQUIVALENT**

```
for( int i=10 ; i>0 ; i = i-2 )
{
      System.out.println(i);
}
```

```
int i=10;
while( i>0 )
{
      System.out.println(i);
      i = i-2;
}
```

Code Pair C: **ARE EQUIVALENT**

```
for( int i=1 ; i<10 ; i++ )
{
      System.out.println(i);
}
```

```
int number = 1;
for( int i=9 ; i>0 ; i-- )
{
      System.out.println(number);
      number++;
}
```

CS 401 MIDTERM REVIEW: SAMPLE PROBLEM SET #1

Q#2 Fill in the blanks in the code fragment, so that when executed, it produces the output indicated below.

Desired Output:

```
0 10 0
1 9 9
2 8 16
3 7 21
4 6 24
```

Code Fragment:

```
int forwardCounter = 0;

int backwardCounter = 10;

while( forwardCounter <  backwardCounter )
{
      System.out.println( forwardCounter + " " +
      backwardCounter + " " +
      ( forwardCounter* backwardCounter );


      forwardCounter ++;

      backwardCounter --;
}
```

Q#3 Write the output that these lines of code generate. Use the following variables and methods.

`// MAIN`
`int a = 6;`
`int b = 8;`
`int c = 2;`
	`private static int foo(`
	`int b, int c)`
	`{`
	`  return b-c;`
	`}`
		`private static void bar(int a, int b)`
		`{`
		`  int temp  = a;`
		`  a = b;`
		`  b= temp;`
		`}`

Code as called from main

Output

1pt	A.	`System.out.println( a / c );`	**3**
1pt	B.	`System.out.println( a % b );`	**6**
1pt	C.	`System.out.println( a + b * c);`	**22**
1pt	D.	`System.out.println( a > b || b > c );`	**true**
1pt	E.	`System.out.println( a < b && b < c );`	**false**
1pt	F.	`System.out.println((double)(a+b+c)/3);`	**5.333333333333333**
1pt	G.	`System.out.println( foo( a, b) );`	**-2**

1pt	H.	`bar( a, b );`
		`System.out.println("a = " + a +`
		`", b = " + b);`
			a = 6, b = 8

Q#4 Write the output that these lines of code generate.

A.
```
String s = new String("Pittsburgh");
int index = 0;
while( s.charAt(index) != 's' )
{    System.out.print(s.charAt(index));
     index++;
}
```

Outputs: **Pitt**

B.

```
String city = new String("Colorado Springs");
System.out.println( city.substring(11, city.length()-1) );
```

Outputs: **ring**

C.

```
int i = 15;
do{
     System.out.print( i + "\t" );
     i = i * 2 / 3;
}while( i > 10 );
```

Outputs: **15**

D.

```
for( int j=0 ; j<10 ; j++ )
     if( j/3 == 0 )
          System.out.print( j + "\t" );
```

Outputs: **0 1 2**

Q#5 Trace this matrix code and determine what it outputs.

```
char [][] charMat = new char[7][7];
for( int i=0 ; i<charMat.length ; i++ )
     for( int j=0 ; j<charMat[i].length ; j++ )
     {    if( i==charMat.length/2 || j==charMat[i].length/2 ||
              i == j || i+j == charMat.length-1 )
               charMat[i][j] = '*';
          else
               charMat[i][j] = ' ';
     }
for( int i=0 ; i<charMat.length ; i++ )
{    for( int j=0 ; j<charMat[i].length ; j++ )
          System.out.print( charMat[i][j] );
     System.out.println();
} //
```

Outputs: (see next page)

```
* * *
 * * *
  ***
*******
  ***
 * * *
* * *
```

Q#6

a) What does it mean to overload a method?

Write more than one method with the same name in the same scope.

b) How does the compiler resolve the overloads (tell them apart)?

By the parameter lists. Each overloaded method must differ from all others somewhere in its parameter list – either the number of parameters or the types must differ somewhere if the number is the same.

Q#7

a) Assume int x=3;

expr => (x/10) evaluates to: **0**

expr => (x/2.0) evaluates to: **1.5**

expr=> (x + 5 * 4 + 3 / 2) evaluates to: **24**

b) Assume `boolean b1 = true, b2 = false; int i = 100, j = -4;`

`b1 || (j>0) || (b2 && i==100)` evaluates to: **true**

`b1 && (b2 || i>j) && (b1 != b2)` evaluates to: **true**

`(b2 || !b2)` evaluates to: **true**

c) Go back to (b) and circle the operators that short circuit

d) Write INVALID after any of the following statements if they are invalid

Assume: `String s1,s2; int i,j ; double d1,d2;`

```
s1 = "Hello World";

s2 = "3" + "14159";

s1 = 3.14159;
```
INVALID

```
d1 = s2;
```
INVALID

e) Assume these definitions:

`int i = 9; boolean boolean b;`

Then trace the following code fragment:

```
int i = 9;
boolean b;

if (i>0)
{   b = (1 % 2) != 0;
    i = i % 2;
}
else
{   i = i + (i%2);
    b = false;
}

if (i%2 ==0 )
```

```
{
     if (i>0)
     {
          i = 7;
          b = true;
     }
}
else
{
    if (b)
         i = 9 - (i%5);
    else
         b = (i%2) != 0 ;
}
```

What are the final values of b and i ? **b: true, i: 8**

Q#8 What is the output of this code segment?

```
boolean inputOk = true;

if (inputOk)
     System.out.println("Thank you");
     System.out.println("Your Input was OK");

if (!inputOk)
     System.out.println("Sorry - try again you");
     System.out.println("Your Input was bad");
```

Thank you
Your Input was OK
Your Input was bad

Q#9 What is the output of this code segment?

After each of the description, write "FOR","WHILE" or "DO" to indicate which loop form is best suited for the task.

Assume you must use a loop to do the given task

a) Prompt a user for a number in a desired range until they get it right **DO**

b) Calculate the sum of all the numbers from 1 to 100 **FOR**

c) Divide a given number by 2 until the number reaches 1. **WHILE**

Q#10 Write a method that that receives an int and returns it as a String

```
static String num2String( int n )
{
     return "" + n;  // LET + OPERATOR DO THE WORK ☺
}
```

Q#11 What's wrong with this code?

```
int[] x = null;
x = new String[5];
for (int i = 0; i < x.length; i++)
     System.out.println(x[i]);
```

X is an int[] ref cant assign String[] ref into it

Q#12 What's wrong with this code?

```
public static void main(String[] args)
{
     int x = 5;
     System.out.println(multiply(x,5));
}
public static int multiply(int num1, int y)
{
     return (x * y);
}
```

var x not visible in multiply method – use num1

Q#13 What's wrong with this code?

```
int[] x = null;
x = new int[10];
for (int i = 0; i < x.length; x = null)
{
     // do whatever
}
```

after first time assigning null into x, it crashes on x.length(dereferencing null ptr)

Q#14 What's wrong with this code?

```
int[] x = null;
x = new int[10];
for (int i = 0; i <= x.length; ++i)
{
     x[i] = i;
}
```

i goes beyond last physical index which is .length-1 array index out of bounds exception occurs

Q#15 What's the output of this program?

```
public static void main(String[] args)
{
     int[] arr = new int[10];
     fillArray(arr);
     System.out.print(arr[t] + " ");
}
static void fillArray(int[] arr)
{
     arr = new int[5];
     for (int i = 0; i < arr.length; i++)
          arr[i] = i * 2;
}
```

0 0 0 0 0 0 0 0 0 0