

Cs 125 Fall 2009 Exam 2 Practice - Key

Before you start, please note: it is possible to get strong clues for answering some of the questions in the review by referring to other questions. For example, converting from code to a flow chart and vice versa. In general, you should not depend on both versions of a question showing up in the exam.

Also note: this study guide is not comprehensive. I strongly suggest reviewing: the practice problems worked on during class, homework assignments, material from the last exam & the self check questions in the text.

Defining constant variables

Write out the statement to create a constant in a method for the standard atmosphere unit & set it to 101,325. Observe proper naming conventions & select an appropriate data type.

```
final int STANDARD_ATMOSPHERE_UNIT = 101325;
```

Write out the statement to create a constant in a class for the Faraday constant and set it equal to 96,485.3383. Observe proper naming conventions & select an appropriate data type.

```
public static final double FARADAY_CONSTANT = 96485.3383;
```

Determine the output for the following operations – if the answer is text, enclose it in quotes:

7 % 4	=	3	125 + 101	=	226
4 % 7	=	4	"125" + 101	=	"125101"
7 % -5	=	2	125 + "101"	=	"125101"
0 % 3	=	0	"125" + "101"	=	"125101"
3 % 0	=	error – requires division by zero!	9 % 3	=	0

Write out the following formulas as Java code:

$$q = 2 \cos \frac{1}{2}(a + b) \cos \frac{1}{2}(a - b)$$

```
q = 2 * Math.cos(0.5*(a+b)) * Math.cos(0.5*(a-b));
```

$$\text{posX} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

```
posX = (-b + Math.sqrt(b*b - 4*a*c))/(2*a);
```

Write out the equation for the following Java code:

```
f = 2*Math.pow(x, 3) + 4*y*y + 2 / 3 * Math.sqrt(z)
```

$$f = 2x^3 + 4y^2 + \frac{2\sqrt{z}}{3}$$

```
g = Math.sin(Math.pow(x, y));
```

$$g = \sin(x^y)$$

Given the following code, write the code that:

- uses the substring method to set g equal to the text "turn"
- sets e equal to the empty string
- prints the length of string g

```
String p = "returning";  
String g;  
String e;
```

```
// set g equal to the substring "turn"
```

```
g = p.substring(2,6);
```

```
// set e equal to the empty string
```

```
e = "";
```

```
// print the length of g
```

```
System.out.println( g.length() );
```

Fill in some code to print out whether a number is above 10, below -3 or in between -3 & 10.

For example:

if the user enters 6, display: "6 between -3 & 10"

if the user enters -5, display: "-5 is below -3"

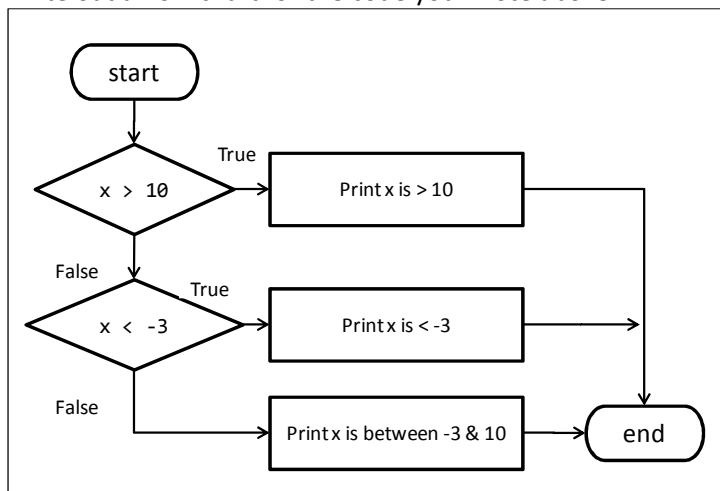
if the user enters 15, display: "15 is above 10"

```
// get input from user & save it in x
int x = in.nextInt();
```

```
// your code goes below here
```

```
if(x > 10)
    System.out.println(x + " is above 10");
else if(x < -3)
    System.out.println(x + " is below -3");
else
    System.out.println(x + " between -3 & 10");
```

Write out a flow chart for the code you wrote above:



Add in a single line of code to test if $-10 < x < -3$ or if $17 > x > 13$

```
// get input from user & save it in x
int x = in.nextInt();
```

```
// your code goes below here
```

```
if((-10 < x && x < -3) || (13 < x && x < 17))
```

```
// your code goes above here
```

```
{
    System.out.println("value is in the range -10 to -3 or 13 to 17");
}
```

Write the code to set the value of the variable **price** based on the variables **size** & **hasHoodie** (hasHoodie is true if the shirt has a hood, otherwise it is false). If the size is something invalid, don't bother setting the price, but print "error". Use the chart below to determine the prices.

	sizes		
hoodie	"small"	"med"	"large"
no	10.00	15.00	20.00
yes	10.00	16.00	25.00

```
String size;  
boolean hasHoodie;  
double price;  
  
// code to set the values of size & hasHoodie  
...  
// your code goes below here
```

```
if(size.equals("small"))  
{  
    price = 10.0;  
}  
else if(size.equals("med"))  
{  
    if(hasHoodie)  
        price = 16.0;  
    else  
        price = 15.0;  
}  
else if(size.equals("large"))  
{  
    if(hasHoodie)  
        price = 25.0;  
    else  
        price = 20.0;  
}  
else  
    System.out.println("error");
```

Examine the following code:

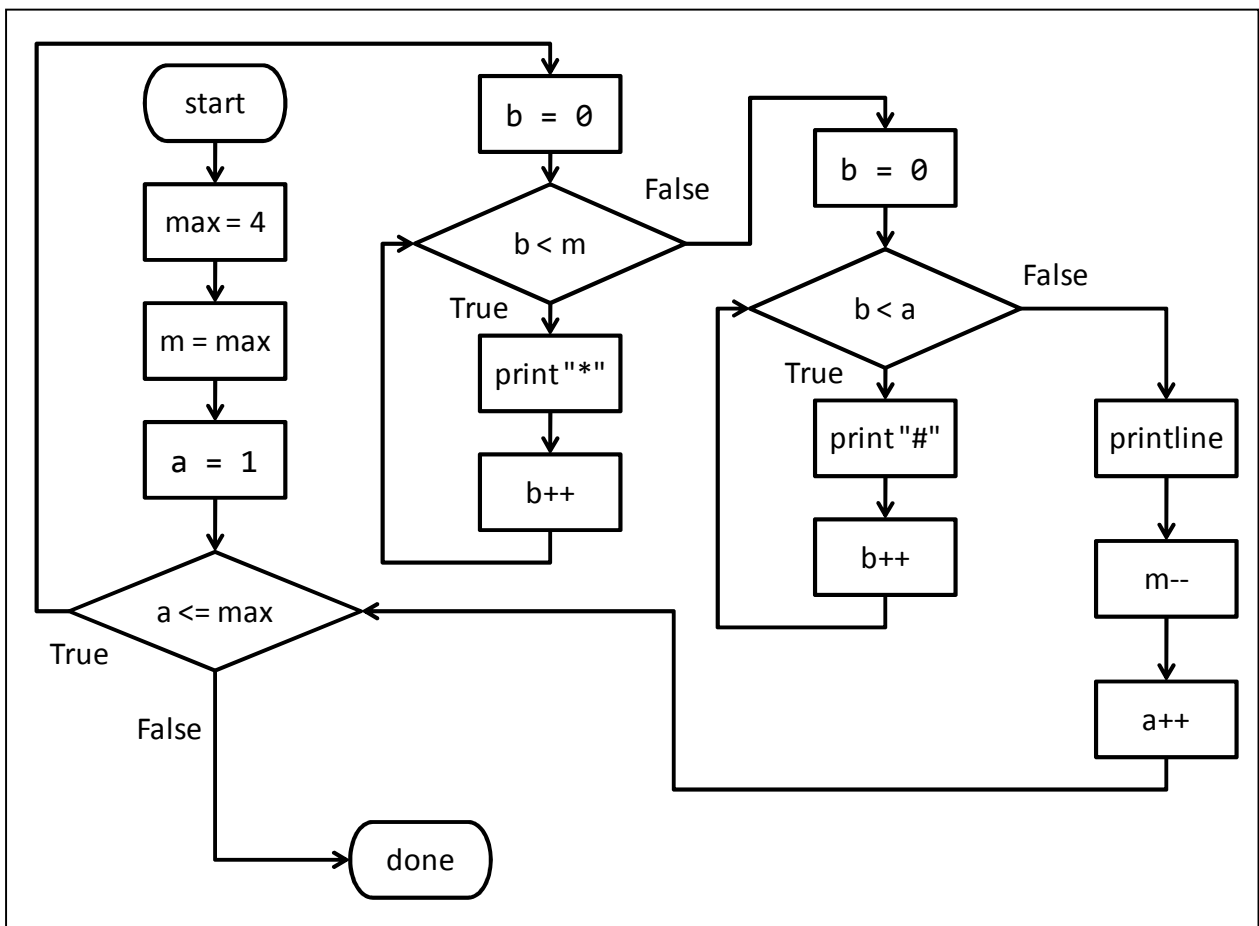
```
int max = 4;
int m = max;
for(int a=1; a<=max; a++)
{
    for(int b=0; b<m; b++)
        System.out.print("*");
    for(int b=0; b<a; b++)
        System.out.print("#");
    System.out.println("");
    m--;
}
```

final output is:

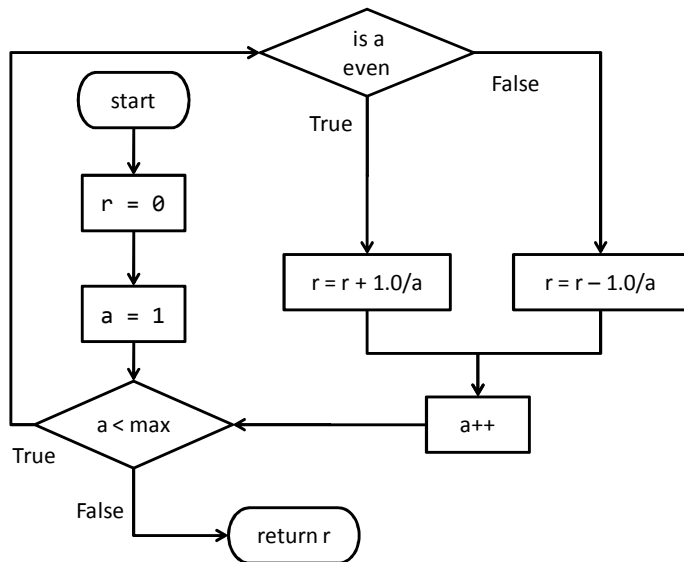
```
****#
***##
**###
*####
```

What is the final output displayed by the code:

Diagram the code using a flow chart:



Examine the following flow chart & convert into a static method called `alternatingSeries` which takes an `int` parameter `max` and returns a `double` as an answer. Be sure to include documentation comments for the method.



```

/**
 * calculate a section of the alternating series math function
 * @param max the number of elements in the series to calculate
 * @return the sum of the 0 to max elements of the alternating series
 public static double alternatingSeries(int max)
 {
     double r = 0;
     for(int a=1; a<=max; a++)
     {
         if((a%2) == 0)
             r = r + 1.0/a;
         else
             r = r - 1.0/a;
     }
     return r;
 }
  
```