Comp 170 Exam 2 Overview.

Exam Ground Rules
The exam will be closed book, no calculators. You may bring the extensive printed javaNotes.pdf plus your notes on two sides of 8.5x11 inch paper (either both sides of one sheet, or two sheets written on single sides).

Main topics that may be on exam 2: Python: user created classes; Java through Loops and Array exercises in my notes, and the Runestone sections referenced.

1. Python user defined classes: __init__, __str__, self, instance variables, writing and using methods. The body of the code in these situations will not be complicated, just emphasizing user defined class syntax. NOT any other special named double underscore methods like __add__. Write code and be able to follow code to answer “What would this print?”.
2. Concepts of variables, functions, decisions, for and while loops, used in Java.
3. Java types int, double, boolean, char, String, ArrayList; casts between numerical types; wrapper types for use in ArrayLists.
4. Keyboard/screen input and output in Java. System.out.print and System.out.println. You may assume the existence of my UI class when writing, or use a Scanner directly. NO file reading/writing.
5. Read all forms of Java loops: while-loop and both kinds of for-loop, but you can choose whatever form you like to write.
6. Write code that involves creating arrays, simple sequential reading or writing to arrays, using them as parameters and return values.
7. Follow combinations of if statements, function calls, while and for loops, to answer “What does this print?”.
8. Simple and compound if statements (reading deeply nested ones, writing simpler ones)
9. Compound conditionals with &&, ||, ! (like Python and, or, not)
10. Write new combinations of if statements and loops (your choice of loop structure), including nested loops. Interactive/listener loops.
11. Operators ++ (increment), -- (decrement), +=, -=, …;
12. You can assume all classes needed have import statements, so you do not have to remember or add the long names with packages and dots.

How the topics get used:
1. “What does this print” will still be important, with more complicated code than you would be asked to write.
2. You will likely be asked to write more significant bits of code than on Exam 1. Loops, arrays, and function calls will likely be prominent, mixed in with if statements....

The recommended text exercises provide further practice.

Read the following before looking at either the problems or the solutions (same as first exam)!
1. Study first, compiling notes, and then look at the sample problems. Look at the list at the top of the page and start by filling in any holes. The sample problems cannot give complete coverage, and if you look at them first, you are likely to study just these points first, and will not get an idea how well you are prepared in general.
2. Do not look at the answers until you have fully studied and tried the problems and gotten help getting over rough spots in the problems if you need it! Looking at the answers before this time makes the problems be just a few more displayed examples, rather than an opportunity to actively learn by doing and check out where you are. The doing is likely to help you be able to do again on a test.

Review questions start on the next page:
1. What is printed by this code fragment?

```java
int x = 1;  //1
while(x < 4) {  //2
    System.out.println(x*x);  //3
    x++;  //4
}
```

2. What is printed by this code fragment?

```java
int x = 2;  //1
while (x < 8) {  //2
    if (x % 2 == 0) {  //3
        x--;  //4
    } else {  //5
        x += 3;
    }
    System.out.print(" "+x);  //6
}
```

3. What is printed by the following fragment?

```java
int[] v = {5, 3, -2, 1};  //1
int n = 0;  //2
for (int k = 1; k < v.length; k++) {  //3
    n += k*v[k];  //4
    System.out.print(" "+n);  //5
}
```

4. What is printed by this program?

```java
class SX2A {
    public static void main(String[] args) {
        String[] list = {"up", "down", "on", "off"};
        System.out.println(foo(list));
    }
    public static String foo(String[] a) {
        String s = ""
        for(int i = a.length-1; i >= 0; i--){
            s += a[i] + a[i];
        }
        return s;
    }
}
```

5. Complete the function definitions:

```java
/** Squares the elements of x:*  
 *  For example, if x initially contains 2, 3, 5, 7,  
 *  then at the end x contains 4, 9, 25, 49. */  
public static void aSqr(int[] x) {

/** returns the product of the elements of x (or 1 if there are no elements):  
 *  For example if x initially contains 2, 3, 5,  
 *  then aProd returns 30 (2*3*5). */  
public static int aProd(int[] x) {
```
6. What is printed by this code fragment?
```java
int a = (int)((5/3)*4.0)
int b = (int)((8.88/2)*20)
System.out.println(a + " " + b);
```

7. Complete the static method below, which shows the results of repeatedly dividing an integer n by k, with
   integer division, stopping at 0, and printing each number considered, with a blank before it. Examples:
   Output for keepDividing(6, 2) is 6 3 1 0   Output for keepDividing(66, 3) is 66 22 7 2 0
```
```java
public static void keepDividing(int n, int k)
```

8. Complete the definition of the method.
   ```java
   /**
   * prints out all the even numbers between 0 and n, including 0 and n, starting with 0, each with one space before it.
   * Note n may be negative. For example evenPrint(6) displays 0 2 4 6       evenPrint(0) displays 0
   * evenPrint(3) displays 0 2 evenPrint(-4) and evenPrint(-5) both display 0 -2 -4 */
   public static void evenPrint(int n)
   ```

9. Complete the definition of the method.
   ```java
   /**
   * Return a new array with the numbers in nums negated.
   * If nums contains 5, 2, -4 then the array returned contains -5,-2, 4. */
   public static int[] negArray(int[] nums)
   ```

10. Write a main program that finds the winner in a vote. The program starts by printing “Enter votes”.
    Suppose people take turns entering "y" or "n", and when everyone has voted, someone enters "q". The
    program ends by saying either "Yes won", "No won" or "A tie". You may assume the only answers given
    are "y", "n" or "q". (No error testing is needed.) (You may use UI.promptLine with empty prompt.)
    Remember test Strings for equality with equals, or test characters for equality with ==.
    Three sample runs:
    Enter votes: Enter votes: Enter votes:
    y      n      y
    n      n      n
    y      y      n
    y      q      y
    y      No won q
    q      A tie
    Yes won
    A tie
    ```java
    public static void main(String[] args)
    ```

11. What is printed by this code fragment?
    ```java
    for (int a = 1; a < 4; a++) {
        for (int b = 5, b < 7; b++) {
            System.out.print("/" + a + " " + b);
        }
    }
    ```

12. Complete the body of the method:
    ```java
    public static ArrayList<String> shortWords(ArrayList<String> words)
    // Return a new list of all the entries of words that are shorter than 4 characters. For example if words contained
    // "Yes", "I", "like", "my", "Python", then the new returned list would contain "Yes", "I", "my".
    ```

13. What is printed?
    ```java
    int[] a = {9, 5, 8, 6};
    int i = 2;
    System.out.println(2*a[i] + " " + a[i+1] + " " + (a[i-1] - a[i-2]));
    ```
14. Complete this code (a bit of a challenge to do concisely without the hint):

```java
// ** print nested boxes with the characters in s; a single copy of the last character is innermost.
* nest("abc") prints:  nest("12") prints:  nest("x") prints:
  aaaaa                111                x
  abbbba               121
  abcba                111  // Hint: How many rows and columns? The index in s
  abbbba               // of the char to print is given by the minimum distance
  aaaaa               // to any edge; you can use Math.min repeatedly.

static void nest(String s)
```

15. What does the Python code print?

```python
class SamX:
    def __init__(self, x):  #1
        self.x = x  #2
        self.y = 7  #3
    
def diddle(z):  #4
        self.y += self.x + z  #5
    
def __str__(self):  #6
        return 'x: {} y: {}'.format(self.x, self.y)  #7

sx = SamX(3)  #8
sx.diddle(5)  #9
print(sx)  #10
```

16. Write an instance method, areSame, for the Python class SamX, above, that returns True if the two instance variables are equal, and False otherwise. Can you do it without an if-statement?

```python
# SOLUTIONS ON NEXT PAGE
```
Exam 2 Review problem solutions
1:  

```
line n comment  
1  1  
2  2<4 true  
3  print 1*1 = 1  
4  2 1+1=2  
2  2<4 true  
3  print 2*2 = 4  
4  3 2+1=3  
2  3<4 true  
3  print 3*3 = 9  
4  4 3+1=4  
2  4<4 false  
```

2:  

Details of each time through the loop: init x = 2
2 < 8 so do body of loop: even x, so decrement x: x = 1, print
1 < 8 so do body of loop: not even x, so add 3: x = 4, print
4 < 8 so do body of loop: even x, so decrement x: x = 3, print
3 < 8 so do body of loop: not even x, so add 3: x = 6, print
6 < 8 so do body of loop: even x, so decrement x: x = 5, print
5 < 8 so do body of loop: not even x, so add 3: x = 8, print
not 8 < 8: done with while statement

3:  

```
line n k comment  
index  0 1 2 3  
1  v value 5 3 -2 1  
2  0  
3  1 1<4 true  
4  3 0+1*v[1] = v[1] = 3  
5  print 3  
3  2 1+1=2; 2<4 true  
4  -1 3+2*v[2] = 3+2*(-2) = -1  
5  print -1  
3  3 2+1=3; 3<4 true  
4  2 -1+3*v[3] = -1+3*(1) = 2  
5  print 2  
3  4 3+1=4; 4<4 false  
```

4:  

```
(loop appends two copies of each array element, traversing the array in reverse order)  
```
5:
static void aSqr(int[] x)
{
    for (int i = 0; i < x.length; i++) {
        x[i] = x[i] * x[i];
    }
}

static int aProd(int[] x)
{
    int prod = 1;
    for (int i = 0; i < x.length; i++) {
        prod = prod * x[i];
    }
    return prod;
}

6: 4 88
(int)((5/3)*4.0)
= (int)((1)*4.0)  integer quotient
= (int)(4.0)
= 4

(int)((8.88/2)*20)
= (int)(88.8)
= 88 casting to int truncates fractional part

7: {
    System.out.print(" "+n);  // do not forget the first value
    while (n != 0) {
        n /= k;
        System.out.print(" "+n);
    }
}

8:
{
    if (n > 0)
        for(int i = 0; i <= n; i+=2) {
            System.out.print(" "+i);
        }
    else
        for(int i = 0; i >= n; i-=2) {
            System.out.print(" "+i);
        }
}

9:
{
    int[] negated = new int[nums.length];
    for(int i = 0; i <= nums.length; i++) {
        negated[i] = -nums[i];
    }
    return negated;
}
10:{ // can count both yes and no, or just extra yes's:
    int extraYes = 0;
    System.out.println("Enter votes:"); // use these comments keeping char c
    String ch = UI.promptLine(""); // char c = UI.promptLine("").charAt(0);
    while (!ch.equals("q")) { // condition (c != 'q')
        if (ch.equals("y")) { // condition (c == 'y')
            extraYes++;
        } else {
            extraYes--;
        }
        ch = UI.promptLine(""); // c = UI.promptLine("").charAt(0);
    } // Could read lines from a Scanner, but need to set it up:
    Scanner in = new Scanner(System.in);
    // Then read next line with:
    ch = in.nextLine();
}
11:/1 5/1 6/2 5/2 6/3 5/3 6
Finish inner loop before returning to outer loop heading:

12:
    { // or by index in words, use next two lines
        ArrayList<String> shorts = new ArrayList<String>();
        for(String word: words) { // or: for(int i = 0; i < words.size(); i++) {
            if (word.length() < 4)
                shorts.add(word);
        }
        return shorts;
    }
13: 16 6 -4
coming from
2*a[2], a[3], (a[1] - a[0]),
then 2*8 6 (5-9)

14: If we let the row and column indices r and c start from 0, then the highest value is n below. As suggested, we go through all row and column positions calculating the minimum distance to an edge.
An example: In the first example n is 4. The character at row 3, column 2 (counting from 0) is 'b', and min distance to edge vertically is min(3, 4-3) = 1; min distance horizontally is min(2, 4-2) = 2; overall min(1, 2) is 1, so the character chosen is s.charAt(1): 'b'. Same idea in general. Nested loops for rows and columns; remember the place for newline. This is longer than you are likely to see on an exam, but does demonstrate printing in nested loops.

```java
{  
    int n = 2*s.length() - 2;
    for (int r = 0; r <= n; r++) {
        int rmin = Math.min(r, n-r);  // or accumulate string for row
                                      // String row = "";
        for (int c = 0; c <= n; c++) {
            int cmin = Math.min(c, n-c);
            int d = Math.min(rmin, cmin);
            System.out.print(s.charAt(d));    // row += s.charAt(d);
        }
        System.out.println();  //need newline
    }
}
```

15: x: 3 y: 15

line self.x self.y x z comment
    t.x t.y (will be in line 8 assignment)
1-7  remember class definition
  8  call constructor, pass 3
  1
  2
  3
  4
  5
  6
    t same as self in constructor (no fancy arrows like in CodeLens)
  7  call diddle for t, pass 5
  8
  9
 10
10 print value returned, x: 3 y: 15

16: # would go inside class SamX definition, indented with other methods

```python
    def areSame(self): # constructor showed instance variables x and y
        return self.x == self.y
```