Lecture 11 (review session for intro to computing)
==================================================================

1) (Big-Oh notation continued)

Assume that each of the following operations takes 1 time unit: an arithmetic op, a comparison, a logical op, an assignment, accessing a particular row in an array, accessing a particular column in an array, a variable declaration (allocation of space in memory for the declared variable).

```java
public void fillArray(int[][] arr, int n)
{
    int row, col;
    for (row=0; row < n; row++)
        for (col=0; col < n; col++)
            arr[row][col] = 10*row + col;
}
```

a) Find the worst-case time, T(n) for the method fillArray above.

Solution:

a) We have two int declaration operations (allocation of space for variables).
   For each loop, there is one initialization (row = 0 or col =0), n+1 for loop comparisons (row < n or col < n) and n increments (row++ or col++). Each increment is equivalent to an assignment and an arithmetic op. Each loop repeats the statements/operations contained within it n times.

   Within the inner for loop there are two operations to access a particular row and column of the 2 dimensional array (arr[row][col]), one assignment of a value to that particular array location, one multiplication, and one addition.

   \[ T(n) = 2 + 1 + n + 1 + 2n + n * (1 + n+1 + 2n + 5) \]
   \[ = 2 + 2 + 3n + n * (7 + 3n) \]
   \[ = 2 + 2 + 3n + 7n + 3n^2 \]
   \[ = 3n^2 + 10n + 4 \]

   \[ 3n^2 + 10n + 4 <= 3n^2 + n^2 + n^2 \]
   for \( c = 5 \), \( n0 = 6 \), and \( T(n) <= c*f(n) \)
   \( (3n^2 + 10n + 4 <= 5n^2) \)

b) therefore \( T(n) \) is \( O(n^2) \)

==================================================================

2) (2-dimensional arrays)

What are the contents of the array arr above (from the method fillArray) when n = 5?

Solution:
3) (Big-Oh notation)

We have an algorithm which has time $T(n) = 4n + 4$.

a) To find the worst case time, if we pick the value 5 as the constant $c$, for the inequality $T(n) \leq c*f(n)$ ($4n + 4 \leq 5n$ in this case) what is the value of $n_0$?

b) If we pick the value 6 as the constant $c$, for the inequality $T(n) \leq c*f(n)$ ($4n + 4 \leq 6n$ in this case) what is the value of $n_0$?

Solution:

a) $n_0 = 4$

b) $n_0 = 2$

4) (pre-increment/post-increment pre-decrement/post-decrement operators)

What are the values of result1 through result4?

```java
public static void main(String[] args) {
    int i=25, j=30, k=4, l = 9;
    int result1, result2, result3, result4;
    result1 = 5 + ++i;
    result2 = 87 / --j;
    result3 = 50 % k++;
    result4 = 6 * l--;
}
```

Solution:

result1 is 31 (5 + 26)
result2 is 3 (87 / 29)
result3 is 2 (50 % 4)
result4 is 54 (6 * 9)

5) (Primitive and class types and call/return-by-value)

Remember:

In Java you make a class cloneable by having it implement the Cloneable interface; e.g., to make class Person cloneable you would write:

```java
public class Person implements Cloneable {
    // the usual class variables, constructors, and methods
    // go here
}
```

a) What are the 3 lines that the main method in class TestNumber prints
b) What are the 2 lines that the main method in class TestVector prints out?
c) What changes would you make in class TestVector to ensure that the Vector v in that class cannot be inappropriately modified?

```java
public class TestNumber {
    private int number;

    public void setNumber(int n) {
        number = n;
    }

    public int getNumber() {
        return number;
    }

    public static void main(String[] args) {
        TestNumber newNum = new TestNumber();
        newNum.setNumber(25);
        System.out.println("number is: " + newNum.getNumber());
        int n1 = newNum.getNumber();
        n1 = 90;
        System.out.println("number is: " + newNum.getNumber());
        System.out.println("n1 is: " + n1);
    }
}

public class TestVector {
    private Vector v;

    public void setVector(int[] arr) {
        v = new Vector(arr.length);
        for (int i = 0; i < arr.length; i++)
            v.addElement(new Integer(arr[i]));
    }

    public Vector getVector() {
        return v;
    }

    public void printVector() {
        Integer n;
        System.out.print("Vector v has contents: ");
        for (int i=0; i < v.size(); i++) {
            n = (Integer)v.elementAt(i);
            System.out.print(" "+n.intValue());
        }
        System.out.println(".");
    }

    public static void main(String[] args) {
        int[] arr1 = {1, 2, 3, 4, 5};
        TestVector newVec = new TestVector();
        newVec.setVector(arr1);
        newVec.printVector();
        Vector n1 = newVec.getVector();
        n1.addElement(new Integer(90));
        n1.addElement(new Integer(-25));
        newVec.printVector();
    }
}
```
lecture11solutions

Solution:

a)
number is 25
number is 25
n1 is 90

b) Vector v has contents: 1 2 3 4 5.
Vector v has contents: 1 2 3 4 5 90 -25.

c)
public Vector getVector()
{
    return v.clone();
}

==============================================================================

6)

(static and non-static variables)

a) What is the line printed out by the main method of class TestChar1?
b) What is the line printed out by the main method of class TestChar2?

class TestChar1 {
    private char c;

    public void setChar(char ch)
    {
        c = ch;
    }

    public char getChar()
    {
        return ch;
    }

    public static void main(String[] args)
    {
        TestChar1 newChar1 = new TestChar1();
        newChar1.setChar('a');
        TestChar1 newChar2 = new TestChar1();
        newChar2.setChar('z');
        char c1 = newChar1.getChar();
        char c2 = newChar2.getChar();
        System.out.println("c1 is " + c1 + " c2 is " + c2);
    }
}

class TestChar2 {
    public static char c;

    public void setChar(char ch)
    {
        c = ch;
    }

    public char getChar()
    {
        return ch;
    }

    public static void main(String[] args)
Solution:

a) c1 is 'a' c2 is 'z'
b) c1 is 'z' c2 is 'z'

For a class that has a static variable, only one version of that variable exists for all instantiations of objects of that class. This means that if there are 20 objects created of type TestChar2 as given directly above, all these instantiations will have a variable called number with the same value. Static variables can be used by different objects of the same class for synchronization.

7) (spot the errors -- 7 compile-time, 2 run-time)

```java
import java.util.*;
public class Test{
    String s1 = "hey";
    Vector strVec = new Vector(20);
    char ch = 'a';

    public void setVector(String[] strs)
    {
        int i = 0;
        do {
            strVec.addElement(strs[i]);
        } while (i < strs.length)
        strVec.trimToSize();
    }

    public static void main(String[] args)
    {
        String a = {"one", "two", "three", "four"};
        setVector(a);
        String b[] = {};
        setVector(b);
    }
}
```

Solution (errors):

```java
import java.util.*;
public class Test{
    String s1 = "hey";
    Vector strVec = new Vector(20);
    char ch = 'a'; // should be char ch = 'a'; (compile)

    public void setVector(String[] strs)
    {
        int i = 0;
```
8) (iteration and recursion)

Write a method int square(int n) that given a number n returns the value of n*n (n squared) without using the multiplication operator. You may use your knowledge of multiplication via addition from homework 1.

a) Your method int square(int n) should call a recursive method that actually computes the square of n.

b) Your method int square(int n) should call an iterative method
that actually computes the square of n.

Solution:

a)

```java
public int square(int n)
{
    return product(n,n);
}

public int product(int n, int m)
{
    // n and m are whole numbers greater than or equal to 0
    // base case
    if (m==1) {
        return n;
    }
    else {
        return(n + product(n, m-1));
    }
}
```

b)

```java
public int square(int n)
{
    return product(n,n);
}

public int product(int n, int m)
{
    int p = 0;
    for(int i=0; i < m; i++)
        p+=n;
    return p;
}
```

(Tracing through a recursive method)

9)

Write out a trace of the return values when
fib(5) is called.

```java
public int fib(int n)
{
    // calculates fib(n) where n >= 0
    if (n == 0)
        return 0;
    else if (n == 1)
        return 1;
    else
        return fib(n-1) + fib(n-2)
}
```

Solution:

```plaintext
fib(5) = fib(4) + fib(3)
       = [fib(3) + fib(2)] + [fib(2) + fib(1)]
       = [[fib(2) + fib(1)] + [fib(1) + fib(0)]] + [[fib(1) + fib(0)] + [1]]
       = [[[fib(1) + fib(0)] + [1]] + [1 + 0]] + [[1 + 0] + [1]]
       = [[[1 + 0] + [1]] + [1 + 0]] + [[1 + 0] + [1]]
       = [[[1] + [1]] + [1]] + [[1] + [1]]
       = [[[2]] + [1]] + [[2]]
       = [3] + [2]
       = 5
```
10) 
(inheritance)
Imagine that you have defined a class called BaseClass which has the following attributes:

```java
private int num;
private String arg;
```

and the following methods:

```java
public void doSomething(String app);
private void appendArg(String app);
public int getNum();
public void setNum(int n);
public String getArg();
public void setArg(String a);
```

You create a new class DerivedClass that is a subclass of BaseClass and define a variable newClass in DerivedClass's main method as follows:

```java
DerivedClass newClass = new DerivedClass();
```

Which of the following calls in main would be invalid:

```java
int a = newClass.getNum();
newClass.setNum(56);
newClass.setArg("happy happy");
newClass.appendArg("joy joy");
String newStr=newClass.getArg();
```

Solution:

```java
newClass.appendArg("joy joy");
// you cannot access private methods or variables of a base class
// within a derived class
```

11) 
(method overloading)

What's wrong here?

```java
import java.util.*;
public class Area{
    public double area(int s1, double s2)
    {
        return(s1*s2);
    }
    public double area(int s1, int s2)
    {
        return(s1*s2*1.0);
    }
    public double area(double r)
    {
        return(Math.PI*r*r);
    }
    public int area(int s1, int s2)
    {
        return(s1*s2);
    }
}
```

Page 8
Solution:

```java
int area(int s1, int s2) {
    // can't overload on return type only
    // this method and double area(int s1, int s2)
    // differ only in terms of their return type
    // so one of them has to go
    return(s1*s2);
}
```

12)
(exceptions)

What's the difference in output between a call to

double quot = divide1(5, 0);
and
double quot = divide2(5, 0);

```java
public double divide1(int numerator, int denominator)
{
    double quotient = SENTINEL;
    try {
        if (denominator == 0)
            throw new DivideByZeroException();
        quotient = numerator/(double)denominator;
        System.out.println(numerator + "/
                + denominator
                + " = " + quotient);
    } catch(DivideByZeroException e) {
            System.out.println(e.getMessage());
            System.exit(0);
    } return(quotient);
}

public double divide2(int numerator, int denominator)
{
    double quotient = SENTINEL;
    try {
        if (denominator == 0)
            throw new DivideByZeroException();
        quotient = numerator/(double)denominator;
        System.out.println(numerator + "/
                + denominator
                + " = " + quotient);
    } catch(DivideByZeroException e) {
            System.out.println(e.getMessage());
    } finally { return(quotient); }
}
```

Solution:

divide1 exits after printing out the error message,
divide2 prints the error message and then returns the value SENTINEL when the finally block is executed.

13)
(command line input to a program)

a) If I've defined a java class TestNew and compiled it, what is the value of args.length in main after I type in the following command?:

    java TestNew input.txt output.txt

b) What would the values stored in args be?

Solution:
a) 
args.length = 2

b) 
args[0] = "input.txt"
args[1] = "output.txt"

14)
We want to copy the contents of one file to another in the class defined below. What expression should go in the parentheses after the while statement. What statements should occur in the body of the while loop? What happens when a FileNotFoundException occurs in main given the class definition below (if for example, input.txt cannot be found)?:

```java
public class CopyFile{
    public static void main(String[] args) {
        try {
            FileReader fr = new FileReader("input.txt");
            FileOutputStream fw = new FileOutputStream("output.txt");
            BufferedReader in = new BufferedReader(fr);
            PrintWriter out = new PrintWriter(fw);
            String getLine = in.readLine();
            while ( ) {
                in.close();
                out.close();
            } catch (Exception e2) {
                System.err.println("An exception occurred");
                return;
            } catch (IOException e1) {
                System.err.println("An IO exception occurred");
                return;
            } catch (FileNotFoundException e) {
                System.err.println("File not found!");
                return;
            }
        }
    }
}
```

Solution:
In the body of the loop we want to copy the line we've
read from the input file to the output file, and read another line (we check whether we have reached the end of the input file by testing if getLine == null in the while loop condition).

The error message "An exception occurred" is printed. To point out that it is a FileNotFoundException, we would need to re-order the catch-blocks so that the more specific exceptions like FileNotFoundException and IOException come before the more general one, Exception.

```java
public class CopyFile{
    public static void main(String[] args)
    {
        try {
            FileReader fr = new FileReader("input.txt");
            FileOutputStream fw = new FileOutputStream("output.txt");
            BufferedReader in = new BufferedReader(fr);
            PrintWriter out = new PrintWriter(fw);
            String getLine = in.readLine();
            while ( getLine != null ) {
                out.println(getLine);
                getLine = in.readLine();
            }
            in.close();
            out.close();
        } catch (FileNotFoundException e) {
            System.err.println("File not found!");
            return;
        } catch (IOException e1) {
            System.err.println("An IO exception occurred");
            return;
        } catch (Exception e2) {
            System.err.println("An exception occurred");
            return;
        }
    }
}
```