Final Examination
CSE 11, UCSD
Practice

RULES:

1. Don't start the exam until you are told to.

2.. This is a closed-book, closed-notes, no-calculator exam. Don't refer to any materials other than the exam itself.

3. Write your name, and your cs11 login name, on each page of the exam when you get to it. Also check to make sure you have all the pages.

4. Do not talk to anyone but an exam proctor during the exam. Do not look at anyone else's exam. If you're wearing a billed cap, turn the bill around back or take it off. Turn off cell phones and pagers.

5. If you have a question, raise your hand and an exam proctor will come to you.

6. You have 2 hours 30 minutes to finish the exam. When you are done, give your exam to a proctor. The proctor will check your picture ID and sign the ID check below.

7. Exam and course grades will be emailed to your cs11 account by Monday next week.

8..Exams will be returned 3rd week of next quarter in 3218 APM. Look for notices posted in APM for details.

Regrading policy: If, when your exam is returned, you discover an error in grading, immediately attach a note to the exam explaining your reason for requesting a regrade and return the exam to the proctor. Do not write on the exam itself; exams bearing extraneous marks will not be considered for re-grading. NO exams will be considered for re-grading after being removed from the room. Regrades should be requested only if there is clearly a mistake in scoring your paper, such as tallying your score. Also note that the whole exam will be regraded, which could result in a lower score.

#	Topic	max pts	actual pts
1.	General T/F	28	
2.	Identifiers	4	
3.	Literals	10	
4.	Expressions	12	
5.	Array use	15	
6.	Loop execution	12	
7.	General FB	21	
8.	Member access	16	
9.	Class features/args	12	
10.	Function definitions	15	
11.	The future	5	
	TOTAL	150	

GRADER:_____ ID CHECK:_____

1. [28 pts.] (1 pt each correct; -1 pt each incorrect; 0 pt blank) True or False:
   a. __T____ Java is an object-oriented programming language.
   b. __F____ **`for`** and **`if`** statements are examples of iterative control constructs in Java.
   c. __T____ In Java, constructors are never inherited.
   d. __F____ Runtime errors in programs are detected by the compiler.
   e. __T____ You can change the order of application of operators in a Java expression by using parentheses.
   f. __T____ "Design your algorithm before implementing it" is a good rule of Java programming.
   g. __T____ "Use descriptive function names" is a good rule of Java programming.
   h. __F____ "Make instance variables public" is a good rule of Java programming.
   i. __T____ Overloading a method means defining a method to have the same name but a different number or type of arguments.
   j. __T____ In Java, the controlling expression of a **`while`** statement must be of type **`boolean`**.
   k. __T____ A private instance method of an object can access protected instance variables of that object.
   l. __T____ If a static method of a class takes an object of that class as argument, it can access private data members of that object.
   m. __F____ To use a static method of a class, there must first be an instance of the class.
   n. __T____ All instances of a class have the same types of instance variables and the same instance methods.
   o. __T____ Storing numerical data in a binary file usually takes less disk space than using a text file.
   p. __T____ Without sorting or hashing, the best way to search an array is using sequential search.
   q. __T____ In Java, a constructor for the base class is called whenever an object of a derived class is created.
   r. __T____ In Java, a **`float`** variable can contain a larger number than a **`long`** variable can.
   s. __F____ Every Java method definition must contain a return statement somewhere in it.
   t. __F____ A Java applet definition contains a method named **`main`**, which is called when the applet starts.
   u. __F____ In Java, a declaration of a variable of array type creates an array.
   v. __T____ In Java, once an array is created, its length cannot be changed.
   w. __F____ The expression **`new FileOutputStream("f");`** will throw an exception if the file **`f`** does not exist.
   x. __F____ To understand how to use a class, it is important to read the definitions of the private methods of the class.
   y. __F____ If **`arr`** is an array of **`int`**s, **`arr[arr.length]`** refers to the last element of the array.
   z. __F____ The statement **`String[][] a = new String[5][10]`** creates 50 String objects.
   aa. __F____ **`SavitchIn`** is a class in the **`java.io`** package.
   ab. __T____"Top-down" software design involves decomposing a problem into simpler subproblems.
2. [4 pts.] (1 pt each) For each of the following, write YES if it is a syntactically legal Java identifier, NO if not.
   a. _YES_____ **`_my_favorite_identifier_`**
   b. _NO_____ **`getNumerator()`**
   c. _NO_____ **`3e10`**
   d. _NO_____ **`java.awt`**

---

3. [10 pts.] (2 pts each) For each expression, write I if it is an integer literal constant; F if it is a floating literal constant ; C if it is a character literal constant; S if it is a string literal constant; N if it is none of these.

   a. __N____ **`Math.PI`**

   b. __S____ **`"int"`**

   c. __I____ **`31459`**

   d. __N____ **`final char c;`**

   e. __N____ **`'Good bye.\n'`**

4. [12 pts] (2 pts each) Consider these declaration statements:

```
int num = 5;
double bar[] = {0.5, 0.5, -1.5, 2.5}, x1=1.0, x2=7.777, x3=3.0;
```

   Now consider each of the following expressions, in the scope of those declarations. For each, write the *value of the expression* as a literal constant of the appropriate type; or ERROR if the expression would cause a compile-time or run-time error. (Consider each expression separately: when determining the value of an expression, ignore side effects of other expressions in the list, if any.)

   a. ___-1.5_____ **`bar[2]`**

   b. ___false_____ **`bar[0] > bar[1]`**

   c. ___ERROR_____ **`x3 < x2 < x1`**

   d. ___1.0_____ **`x3 = x2 = x1`**

   e. ___3_____ **`num / 2 + num % 2`**

   f. ___4_____ **`num-- - 1`**

5. [15 pts.] (3pts each) For each of the following problems, the user will enter a list of 1000 numbers from the keyboard. Suppose that you want to solve the problem with a Java program containing fewer than 100 statements (but without using recursion or files, or any objects other than an array). For each problem, say whether it is NECESSARY or NOT NECESSARY to use an array to solve the problem, with these constraints. (Think carefully about each one!)

   a. _NECESSARY____Print out only the numbers in the list that are larger than the 200th number entered.

   b. _NOT NEC._____Print out “true” just in case more than 200 of the numbers are divisible by 200.

   c. _NOT NEC._____Print out “true” just in case the largest of the numbers in the list appears in the list more than 200 times.

   d. _NECESSARY ___Print out the 200 largest numbers that were entered.

   e. _NECESSARY___Print out the average of the 200 largest numbers that were entered.

6. [12 pts.] (3 pts each blank) Consider each of the following Java program fragments (a.)-(b.) separately. For each one, state *how many* lines of output it prints out, and what the *largest number* printed out would be. (In these examples there are no infinite loops.)

---

a.
```
int i, j;
for(i=1;i<=50;++i) {
for(j=1;j<=10;++j)
   System.out.println(j);
if(j>10) break;
}
```

How many lines printed?___10_____ Largest number printed?___10_____

---

b.
```
int a[] = {0,3,6,9,12,15}, k = 1;
try {
   do {
      System.out.println(a[k-1]);
      k = k+1;
   } while (k < 5);
} catch (ArrayIndexOutOfBoundsException e) {}
```

How many lines printed?___4_____ Largest number printed?___9_____

---

7. [21 pts.] (3pts each) Fill in the blank with the best answer.

   a. In Java, you refer to an array element by using the array name together with a bracketed int expression called a(n) __index _____.

   b. An array that has only some of its elements storing meaningful values is called a(n) __partially filled_____ array.

   c. If a class is labelled _____final_____ it its definition, it cannot be used as a base class.

   d. The style of programming in which user interactions with GUI objects determine the flow of execution in your program is called _____event driven_____ programming.

   e. A constructor with no arguments is called a(n) _____default_____ constructor.

   f. _____javadoc_____ is a JDK program that can create hyperlinked interface documentation from suitably commented Java source code.

   g. _____inheritance_____ is the acquisition by one class of the variables and methods of another class.

8. [16 pts.] (2 pts each) Consider the following class definitions:

```
public class BB {
    public BB() {x = 11.0;}
    public BB(double x) { this.x = x; }
    public double getIT() {return x + 1.0;}
    public double getX() {return x;}
    private double x;
}

public class DD extends BB {
    public DD() {this(0);}
    public DD(double y) {this.y = y;}
    public DD(DD o) {y = o.y;}
    public double getIT() {return getX() + y ; }
    public boolean testIT() {return y>0;}
    private static double y;
}
```

Now consider the marked statements in the program below. For each one, write as a literal constant of the appropriate type what would be printed when the statement executes; or write ERR if the statement is illegal.

```
public class F {
          public static void main(String args[])
          {
                    BB x = new BB(22.0);
                    BB z = new DD();
                    DD y = new DD(88.0);

a. ___22.0_________   System.out.println( x.getX() );
b. ___11.0_________   System.out.println( y.getX() );
c. ___11.0_________   System.out.println( z.getX() );
e. ___false________  System.out.println( x.equals(z) );
f. ___99.0_________   System.out.println( y.getIT() );
g. ___99.0_________   System.out.println( z.getIT() );
h. ___true_________   System.out.println( y.testIT() );
i. ___ERR_________    System.out.println( z.testIT() );

          }
```

9. [12 pts.] (2pts each blank) Consider the following program:

```
public class Final {
     public static int fun(int j)
     {
          System.out.println(j + 2);
          j = j + 2;
          return j;
     }
     public static void main(String args[])
     {
          int j=2;
          int i=1;
          j = fun(i) + 4;
          System.out.println(i + " " + j);
     }
}
```

When this program runs, three numbers are printed out. What are they?

First __3_____, then ___1_____, and finally ___7_____.

Now consider this somewhat similar program:

```
public class Final {
     public static int fun(int[] i)
     {
          System.out.println(i[0] + 2);
          i[0] = i[0] + 2;
          return i[0];
     }
     public static void main(String args[])
     {
          int j=2;
          int i=1;
          int jj[] = {j};
          int ii[] = {i};
          ii[0] = fun(jj) + 4;
          System.out.println(i + " " + j);
     }
}
```

When this program runs, three numbers are printed out. What are they?

First __4_____, then ___1_____, and finally ___2_____.

10. [15 pts. 3pts ea correct; -3pts ea incorrect; 0pts blank] The problem is to write a Java method with header **`public static boolean find(int[] a, int key)`** that returns true if and only if **`key`** is an element of the array **`a`**. For each of the following, write YES if the statements shown would work correctly as the body of **`find()`**, NO if not.

a.__no_____
```
for(int i=0; i<a.length; i++) {
   if(key==a[i]) break;
   return false;
}
return true;
```

b.__yes_____
```
long count=0;
for(int j=0; j<a.length; j++) {
   if(key==a[j]) count++;
}
return count != 0;
```

c.__yes_____
```
boolean found=false;
int i=0;
while(true) {
   if(key==a[i++]) return true;
}
return found;
```

d.__no_____
```
boolean found = false;
int i=0;
try { while(true) found = (key == a[i++]); }
catch (ArrayIndexOutOfBoundsException x) {}
return found;
```

e.__yes_____
```
int i;
for(i=0; i<a.length && a[i] != key; i++) ;
return i < a.length;
```

11. [5 pts.] As a programmer, what would you like to do with computers that has never been done before? (A brief answer -- 2 or 3 sentences -- in clear English is worth 5 points. Say what you think; we're interested in hearing about it. Continue to the back of this page or the next page if needed.)

(blank paper)