Cornell University
Department of Computer Science

CS 99: Fundamental Programming Concepts
Summer 2000

Preliminary Examination #1
July 12, 2000

Instructions:

- This is a difficult exam. Do not expect to solve every problem or finish the exam in the allotted time. The entire course will be curved at the end of the term. So, for example, it is possible that a 50% on this exam could become a passing grade. It all depends on how well the class does as a whole.
- There are 8 problems on this exam. The number of points each problem is worth is listed in brackets before the problem. Partial credit will be awarded if you provide your reasoning.
- You will have one hour to complete the exam.
- Write all your answers on the exam itself. If you run out of space on a problem, write OVER on the front side of the page, and use the back side to continue your answer.
- This is a closed book exam. You may not use any materials, including calculators, during the exam other than the exam handout and writing instruments.
- If you feel a question is ambiguous, stay in your seat, raise your hand, and someone will come to help you. If you still feel the question is ambiguous, state some reasonable assumptions and solve the problem accordingly.
- If you finish before the last 10 minutes of the exam, you may leave quietly. If you finish in the last ten minutes of the exam, please remain in your seat until the end of the exam. Do not take out any additional materials.
- Do not begin reading the rest of the exam until instructed to begin. When you start, make sure that you have all 9 pages of the exam.
- Write your 4-digit secret number (from the student questionnaire you filled out on the first day of the class – probably the last 4 digits of you student ID number) below. You may also write your name, if you wish, or we will look it up based on your number *after* we grade your exam.

Name or secret number: ____

Problem	**Points**	**Score**
1	20	
2	10	
3	20	
4	15	
5	10	
6	20	
7	10	
8	15	
Total	120	

1. **True/False.** [20 points, each correct answer is worth 2 points, each answer left blank is worth 1 point, each incorrect answer is worth 0 points]

T a. Syntax refers to the rules for writing legal statements in Java.

F b. Arguments are the variables listed in the signature of a method.

F c. Robustness refers to the use of good style in a program.

F d. A variable is in scope from its declaration to the end of the method in which it is declared.

T e. String equality cannot be tested using the `==` operator.

F f. The concatenation operator in Java is `%`.

F g. A call to a method of type `void` can be used as an argument to the `Math.sqrt` method.

F h. A literal is a variable whose value cannot be changed.

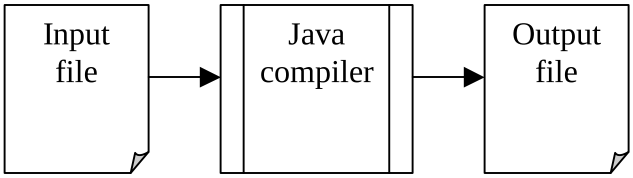
T i. Pseudocode is a way to express algorithms.

F j. It is never valid to assign the empty string to a variable.

2. **Compiling and executing Java programs** [10 points]

   a. [6 points] The Java compilation and execution process

      i. The first phase of the process uses the Java compiler to produce an output file from an input file, as in the following diagram:



Complete the following:

	Input file	Output file
Type of file:	Source code	bytecode
File extension:	.java	.class

      ii. There are two ways to execute the output file from above:

         1. compile it with a JIT

         2. interpret it with a JVM

      iii. Which of the above results in faster execution?

Circle one: [✓ 1] 2 Note: we also took (2) if you got part ii backwards.

   b. [4 points] List the 4 components of computer architecture as discussed in class:

      i. Processor

      ii. Memory

      iii. Input

      iv. Output

3. **Variables and Types** [20 points]

a. [4 points] Circle the variable names in the list below that would be legal in Java:

i. salesTax%

ii. [✓ bwv_645]

iii. 2ndNumber

iv. static

b. [8 points] Choose an appropriate primitive type for each of these values and list the Java keyword for that type next to the value. Recall that `String` is not a primitive type.

i.	2.718281828459	`double`
ii.	$	`char`
iii.	42	`int (byte, short, long)`
iv.	false	`boolean`

c. [8 points] Provide the type of each of the following expressions, given these declarations:

```
int a;
double b;
boolean c;
```

i.	`a > b`	`boolean`
ii.	`“a = ” + a`	`String`
iii.	`(c) ? a : a+5`	`int`
iv.	`1 - b / a`	`double`

Note for (b) and (c): you had to list the Java keyword for the type because the type of the expression is based on the types of the variables and literals in it, all of which have specific names in Java. `bool` is not a type in Java. “integer” is not acceptable for c.iii because all of the four types listed in b.iii above are integers, but only `int` is the correct type for the expression in c.iii.

4. **Program output.** [15 points] For each of the following program fragments, provide the exact output produced by the fragment. Use the box next to the program to provide the output.

   a. [5 points]

```
int i = 2, j=0, k = 3, l=-1;

System.out.println(7 / i);
System.out.println(k+l*j-1);
System.out.println(25 % k);
j *= 3;
System.out.println(j);
l++;
System.out.println(l);
```

```
3
2
1
0
0
```

   b. [10 points] Assume that `a()` has just been called:

```
static void a() {
     d(-1, -1);
     c(3, 0);
     b(2);
     System.out.println();
     System.out.println("!");
}

static int b(int m) {
     return d(2, m);
}

static void c(int j, int k) {
     j = b(1);
     d(1, 3);
}

                 i      j
static int d(int ~~x~~, int ~~y~~) {
     System.out.print(i * j);
     return i * j;
}
```

```
1234
!
```

5. [10 points] Identify all compile-time errors in the following program fragment by circling them and providing a brief phrase describing the error.

```
static foo() {
  int i = 5, j;
  j = bar(i);
}

static double bar(int a) {
  return i * 2
}

static double baz() {
  char c=’Z’;
  qux(char c, boolean true);
}

static int qux(char a, boolean b) {
  return b || a == “5”;
}
```

- `foo()`
  - Missing return type for `foo()`
  - Assignment of double to `j`
- `bar()`
  - `i` not in scope
  - missing semicolon
- `baz()`
  - extra types in method call
  - missing `return`
- `qux()`
  - double quotes around char literal
  - return type mismatch

Note: you received 2 points as a base score, +1 point for each of the above that you identified correctly, and -1 point for each item you claimed was an error but wasn’t.

6. **Writing code fragments** [20 points]

   a. [3 points] Provide the exact signature for `main`:

```
public static void main(String[] args)
```

   b. [7 points] Rewrite this algebraic equation as an assignment statement in Java. You may assume all variables have been appropriately declared.

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

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

c. [4 points] Write a condition for:

Exactly 1 of `x` and `y` is greater than 6

```
(x>6 || y>6) && !(x>6 && y>6)
```

Note there are at least 4 solutions, this is just one of them.

d. [6 points] Finish the following `switch` statement that uses a contestant's place (given in the integer `place`) to set the string `ribbonColor`, according to the following table:

Place	**Color**
1st	Blue
2nd	Red
3rd	White
any other	None

```
switch (place) {
case 1:
     ribbonColor = “Blue”;
     break;
case 2:
     ribbonColor = “Red”;
     break;
case 3:
     ribbonColor = “White”;
     break;
default:
     ribbonColor = “None”;
}
```

7. [7 points] Write a method that accepts a `double` as a parameter and returns the cube of it. You do not need to provide a specification-comment.

```
static double cube(double num) {
     return num * num * num;
}
```

[3 points] Using a call to the method you just wrote, assign the cube of the variable `x` to the variable `y`.

```
y = cube(x);
```

8. [15 points] Write a method that accepts the lengths of the sides of a triangle (all integers) and returns a `boolean` indicating if it is a right triangle. Again, you do not need to provide a specification-comment. Recall the Pythagorean theorem: a right triangle has two sides, *a* and *b*, and a hypotenuse, *c*, such that $a^2 + b^2 = c^2$. Your method should work correctly no matter what the order in which the lengths are passed (e.g., 3, 4, 5; 4, 3, 5; etc.).

```
static boolean isPythagorean(int a, int b, int c) {
     return isPythTriple(a, b, c)
            || isPythTriple(a, c, b)
            || isPythTriple(b, c, a);
}

static boolean isPythTriple(int a, int b, int c) {
     return a*a + b*b == c*c;
}
```

Note you could do this in one method by inlining `isPythTriple()`.