UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE AND ENGINEERING

APS105F — Computer Fundamentals
Final Examination
December 5, 2005

Examiners: Faye Baron, Tobi Kral, James MacLean, Baochun Li

Duration: 150 minutes

- Examination Type A: This is a "closed book" examination; no aids are permitted.
- Calculator Type 4: No electronic or mechanical computing devices are permitted.
- Write your answers in the spaces provided. Please answer using a pen or dark pencil.
- If more space is required, blank pages are provided at the back of the examination.
- Rough work, if necessary, can be done on the backs of the pages.
- This examination has **17** pages, including **3** pages for extra scratch space.
- You must use the Java programming language to answer programming questions.
- Please write clearly in all your solutions.
- You may assume that the following methods of the class `In` are available:
  `getInt`, `getLong`, `getFloat`, `getDouble`, `getString`, and `getChar`.

Circle your lecture section: **LEC 01** J. MacLean or **LEC 02** T. Kral or **LEC 03** B. Li or **LEC 04** F. Baron

Name _____

Student Number _____ ECF Login _____

Marks

Question	1	2	3	4	5	6	Total
Value	17	17	17	17	16	16	100
Mark							

1. [17 Marks]

Part 1: True/False Questions (1 mark each, no penalty for incorrect answers)

Circle the correct answer for each of the following:

1. **True** or **False**: An even number cannot be represented with respect to an odd number base.

2. **True** or **False**: The `trim()` method in the `String` class removes all extra spaces from a `String` object.

3. **True** or **False**: If `s1 == s2` returns `true` for two non-null `String` reference variables `s1` and `s2`, it means that the `String` objects they refer to contain the same characters in the same order.

4. **True** or **False**: Selection sort is $\mathcal{O}(n \log n)$.

5. **True** or **False**: Bubble sort is always faster than insertion sort.

6. **True** or **False**: After `String[] names = new String[5];` is executed, `names[1]` has the value `null`.

7. **True** or **False**: `s1.equals(s2)` returns `true` if and only if `s1` and `s2` refer to the same `String` object.

8. **True** or **False**: There is no limit to the number of times you may recursively call a method.

9. **True** or **False**: Given `String s = "sunshine"`, then `s.substring(3, s.length() - 1)` will evaluate to `"shine"`.

10. **True** or **False**: Class fields are automatically initialized to zero whenever an object for that class is created.

11. **True** or **False**: In each iteration of selection sort, one and only one pair of items in the array will be swapped.

12. **True** or **False**: The statement `int[][][] list = new int[15][][];` leads to a compile-time error.

13. **True** or **False**: The UNIX command `cp` can be used to make a copy of an entire UNIX directory (including all the files and subdirectories inside) to a different directory location.

Part 2: Multiple Choice Questions (1 mark each, no penalty for incorrect answers)

Circle one and only one choice that best answers the question.

1. Consider the following Java code:

```
class Confusion
{
  private static int x = 1;

  private static int getX()
  {
    int x = 2;
    return x;
  }

  public static void main(String[] args)
  {
    System.out.println(getX());
  }
}
```

When we run this Java program, the value printed will be:

(A) 1

(B) 2

(C) Nothing, we will encounter a run-time error

(D) Nothing, this code will not compile

2. An algorithm whose running time is known to be $O(2^n)$ requires t seconds to process a problem of size n. Assuming the same computing environment, about how long would the algorithm need to process a similar problem of size $3n$?

(A) t^3 seconds

(B) $2^n t$ seconds

(C) $2^{2n} t$ seconds

(D) $8t$ seconds

3. Consider the following Java code segment:

```
class DoThings
{
  public static int doSomething(int i)
  {
     int j = i + 1;
     int k = doSomethingElse(i);
     return k;
  }

  public static int doSomethingElse(int x)
  {
     x += j;
     return x;
  }
}
```

Which of the following best describes the output when `DoThings.doSomething(5)` is executed:

(A) the method `doSomething` returns 11

(B) the method `doSomething` returns 10

(C) Nothing, this code will not compile

(D) Nothing, we will encounter a run-time error

4. Which one of the following algorithms, all meant to solve the same problem, is the most efficient?

(A) An $\mathcal{O}(\log n)$ algorithm

(B) An $\mathcal{O}(n \log n)$ algorithm

(C) An $\mathcal{O}(\sqrt{n})$ algorithm

(D) An $\mathcal{O}(n)$ algorithm

(E) An $\mathcal{O}(n^2)$ algorithm

2. [17 Marks]

Please provide a concise answer to the following questions.

1. (3 marks) What will be printed by the following Java fragment?

```
int m = 10;
int n = 0;

while (m > n)
{
    System.out.println(m + "+" + n);
    m --;
    n += 2;
}
```

2. (5 marks) The *Fibonacci* sequence consists of the numbers

$$1, 1, 2, 3, 5, 8, \ldots$$

and is recursively defined as:

$$f(n) = \begin{cases} 1 & \text{if } n = 1 \\ 1 & \text{if } n = 2 \\ f(n-1) + f(n-2) & \text{if } n > 2 \end{cases}$$

A recursive method that calculates $f(n)$ is:

```
public static int fibonacci(int n)
{
  if (n == 1 || n == 2) // assume n > 0
    return 1;

  return fibonacci(n - 1) + fibonacci(n - 2);
}
```

In this context, answer the questions on the next page.

(a) (1 mark) How many times is the method `fibonacci` called to calculate $f(5)$?

(b) (2 marks) Is this method efficient? Why or why not?

(c) (2 marks) In the space below, rewrite the `fibonacci` method so that it calculates $f(n)$ without using recursion.

```
public static int fibonacci(int n)
{

}
```

3. (3 marks) Suppose that an array initially contains the values {5, 3, 6, 2, 4}, and we would like to sort the array in ascending order. Please show what the contents of the array would look like after the first pass of

   (a) insertion sort

   (b) selection sort

   (c) bubble sort

4. (3 marks) The quantity x^n, where x is a real number and n is a positive integer, may be (naively) computed recursively according to:

$$x^n = \begin{cases} x \cdot x^{n-1} & , n > 0 \\ 1 & , n = 0 \end{cases}$$

   Suggest a more efficient recursive computation, where "more efficient" means less recursive calls for a given value of n. Please precisely describe the main idea, without writing any Java code.

5. (3 marks) Assume an integer array, `int[] a`, has been created and initialized. Write a `while` loop that prints all the elements of `a`, one per line, repeatedly and forever. Do **not** use any `if` statements in your solution. For example, if `int[] a = {1, 2, 3}`, the following will be printed by the `while` loop.

```
1
2
3
1
2
3
1
2
3
1
2
⋮
```

3. [17 Marks]

(a) (9 marks) Write a class named `Point` to store and manipulate objects representing points in a plane. Your class must include the following methods:

- (1 mark) A constructor that takes two `double` parameters representing the `x` and `y` coordinates of the point.
- (1 mark) A default constructor that creates a point at `(0, 0)`.
- (1 mark) A `toString()` method that returns a `String` representing a `Point` in the format `(x, y)` (`(1.5, -3.72)` for example).
- (2 marks) An instance method `equals` that takes one `Point` object as its parameter, and returns a `boolean` value `true` if and only if the implicit object and the explicit object parameter represent the same point in the plane.
- (2 marks) An instance method `distance` that takes one `Point` object as its parameter, and returns the distance between the implicit and explicit `Point` objects.
- (2 marks) A class method `countPoints` that returns an `int` type integer, representing the total number of `Point` objects created so far.

(b) (8 marks) Write a class named `Circle` to store and manipulate circle objects. Your class must have the following methods:

- ◇ (2 marks) A constructor that takes a `Point` parameter representing the center of the circle and a `double` parameter representing its radius.
- ◇ (3 marks) An instance method `area` that returns the area of the implicit `Circle` object.
- ◇ (3 marks) An instance method `overlap` that takes one `Circle` object `other` as its parameter, and returns a `boolean` value. The `overlap` method returns `true` if and only if there exists a point that is contained in both the implicit `Circle` object and the explicit `Circle` object.

4. [17 Marks]

Write a method `public static int binarySearch(int [] list, int item)` that performs the *Binary Search* algorithm in a sorted array of `int` type integers, using a *non-recursive* implementation. The array `list` is sorted in ascending order, but there may be duplicates in the array. The method `binarySearch` should return the index of `item` in the array, if the search is successful. If `item` is not found in the array, the method should return `-1`. When there are more than one occurrence of the item to be searched, return the index of the *first* occurrence. For example, if the array is `10, 15, 15, 15, 20`, and the integer to be searched is `15`, then the index `1` should be returned.

```
public static int binarySearch(int[] list, int item)
{




}
```

5. [16 Marks]

In digital signal processing software, signals are often represented as arrays of `double` floating-point numbers. A common operation involving two signals is called *convolution*, and is mathematically described as

$$y[n] = \begin{cases} \sum_{m=0}^{M-1} h[m]x[n-m] & \text{, for } n = M-1 \ldots N-1 \\ 0 & \text{, for } 0 \leq n < M-1 \end{cases}$$

where h is an array of length M, and x and y are arrays of length N. (For example, this operation may be used to apply a digital filter, represented by h, to a signal, represented by x.)

Write a method named `convolve` that takes as parameters two `double` arrays (representing h and x), performs the operation described above and returns an array containing the result of the convolution, *i.e.* y. Pay attention to your array indices.

6. [16 Marks]

The problem of generating all subsets of a set of N elements has a simple recursive formulation. For example, if you start with the set $\{A, B, C\}$, the set of all subsets can be divided into two groups: (1) those that contain A and (2) those that do not. In either case, the subsets in each group simply contain all possible subsets of the remaining elements, $\{B, C\}$. Thus, the complete list of subsets of $\{A, B, C\}$ is

Subsets containing A: $\{A, B, C\}$, $\{A, B\}$, $\{A, C\}$, $\{A\}$

Subsets not containing A: $\{B, C\}$, $\{B\}$, $\{C\}$, $\{\}$

Write a Java method `public static void subset(String s)` that prints all subsets of a set given as a parameter. We use a `String` to represent a set. For example, the set $\{A, B, C\}$ is represented by a `String` `"ABC"`. You may assume that any set passed to your method has no repeated elements, and elements of the set are characters. As an example, when `subset("ABC")` is called, we observe the following output:

```
{ABC}
{AB}
{AC}
{A}
{BC}
{B}
{C}
{}
```

You must use recursion in your implementation.

Use this page to continue your solutions to any of the questions in this examination, if more space is needed. Please clearly mark the question number(s).

Use this page as scratch space. Please do not write your solutions to exam questions on this page.

Use this page as scratch space. Please do not write your solutions to exam questions on this page.

Use this page as scratch space. Please do not write your solutions to exam questions on this page.