UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE AND ENGINEERING

APS105F — Computer Fundamentals
Final Examination — December, 2002

Examiners: John Carter, Baochun Li, James MacLean

Duration: 2.5 h

- 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.
- 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 13 pages.
- You must use the Java programming language to answer programming questions.
- You may use any methods from the `Math` and `String` classes.
- You may assume that the following methods of the class `In` are available:
  `getInt`, `getLong`, `getFloat`, `getDouble`, `getString`, and `getChar`.

Circle your lecture section: L0101 MacLean or L0102 Carter or L0103 Li or L0104 Carter

Name ______________________________

Student Number ____________________ ECF Login ____________________

MARKS

Question	1	2	3	4	5	6	7	8	9	10	Total
Value	20	10	15	5	10	15	15	15	15	15	135
Mark											

1. [20 Marks]

Each part of this question is worth two marks. Answers should be brief: point form is permitted.

(a) Given that `x` is of type `double`, write a statement that will round `x` to one decimal place.

(b) Simplify the expression `!(x != y && x != z)` as much as possible.

(c) Rewrite the following fragment using a `for` statement.

```
i = 10;
do
{
  i--;
  System.out.println(i*i);
}
while (i > 0);
```

(d) Suppose that an array that initially contains the values `{5, 4, 7, 2, 3}` is to be sorted into ascending order using a selection sort. Show the contents of the array as it would appear after *each* of the first two passes of the selection sort.

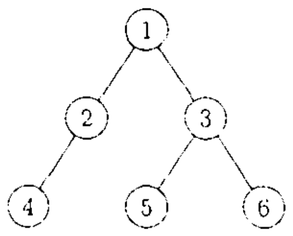
(e) Shown below is a Java method for adding matrices. Give a big-Oh estimate for the task of executing this method, assuming the matrices are square with n rows and n columns.

```
public static double[][] sum (double[][] a, double[][] b)
{
  double[][] c = new double[a.length][a[0].length];

  for (int i = 0; i < a.length; i++)
    for (int j = 0; j < a[0].length; j++)
      c[i][j] = a[i][j] + b[i][j];

  return c;
}
```

(f) Suppose that a method that is known to have time complexity $O(n^2)$ requires 20 time units to process a list of size 500. Estimate the time that would be required by the method to process a similar list of size 2000.

(g) What is a stack?

(h) The diagram shows a binary tree. State the order in which the nodes of the tree would be visited by a postorder traversal.



(i) The diagram below shows a binary search tree containing the elements 4, 7, and 9. Draw diagrams to show all other possible binary search trees containing these values.

```
   7
  / \
 4   9
```

(j) Find the base 10 representation of $2B9_{12}$.

2. [10 Marks]

Suppose that we want to create a class TempWorker in which each object has the following fields:

```
private String name;
private boolean experienced;
private int wpm;
```

The field experienced is true if and only if the worker is experienced with computers while the field wpm is the worker's typing speed in words per minute.

(a) Write an appropriate constructor with three parameters.

(b) Write an equals method for the class. Two TempWorker objects should be considered to be equal if (1) they are both experienced with computers or (2) neither one is experienced with computers and their typing speeds differ by less than five words per minute.

(c) Write a toString method for the class. As an example, for a worker named George who has experience with computers and types at a rate of 55 wpm, the method should return:
"George - experienced (yes) - 55 wpm"
If the worker is not experienced with computers, the word no should appear inside the parentheses.

3. [15 Marks]

Consider the following method:

```
public static int mystery (int[] list, int start, int finish)
{
  int result;
  System.out.println("Range: " + start + " " + finish);
  if (start == finish)
    result = list[start];
  else
  {
    int middle = (start + finish)/2;
    result = Math.max(mystery(list,start,middle),mystery(list,middle+1,finish));
  }
  System.out.println("Result: " + result);
  return result;
}
```

What would be printed by execution of the following statements?

```
int[] a = {5,7,9,2};
System.out.println(mystery(a,0,3));
```

4. [5 Marks]

Assume that we have an array, a, whose elements are of type `int`. Complete the definition of the method `duplicateSearch` to search for a specific integer `key` in the array. Rather than simply searching for the location of the key, we wish to know if there exist duplicates of the key in the array. If *exactly* two copies of the same integer `key` exist, the method returns `true`; otherwise it returns `false`.
For example, if we perform `duplicateSearch` for the key 5 in an array {`5, 4, 3, 1, 5, 1, 1`}, the method returns `true`. If we perform `duplicateSearch` for the key 4, 6 or 1 with the same array, the method returns `false`.

```
public static boolean duplicateSearch(int[] a, int key)
```

5. [10 Marks]

Using recursion, complete the definition of the `quickSort` method, sorting an array of integers (type `int`) in increasing order using quicksort. When choosing the *pivot* in each pass, randomly choose an item among items to be processed in that pass. In your solution, you may assume the existence of a method `partition` with header

```
public static int partition (int[] list, int low, int high)
```

The `partition` method rearranges the elements of the sub-array of `list` from index `low` to index `high` so that the element, x, that was at `list[low]` is now in a position so that all elements preceding it are less than or equal to x and all elements following are greater than x. The value returned by `partition` is the index of x after the partition has been performed.

```
public static void quickSort(int[] list)
```

6. [15 Marks]

Suppose that linked lists are represented in the usual way seen in class with `List` objects containing a field:

```
private Node head;
```

and `Node` objects containing fields:

```
int info;
Node link;
```

Assuming that lists are maintained so that their `int` fields are in increasing order with no repetitions, complete the definition of the method `intersectSum` so that it returns the sum of the `info` fields of the elements that appear in both the implicit `List` object and the list referred to by `other`. As an example, if the implicit `List` object contains 3 4 6 7 and the list referred to by `other` contains 1 3 6 8, then the method should return the value 9 (because the lists contain the common elements 3 and 6).

```
public int intersectSum (List other)
```

7. [15 Marks]

Two strings are said to be *anagrams* of one another if they contain the same characters in a (possibly) different order. For example, "the morse code" is an anagram of "here come dots", and "elvis" is an anagram of "lives". Complete the definition of the method isAnagram that returns true if and only if its two String parameters are anagrams of one another.

```
public static boolean isAnagram (String s1, String s2)
```

8. [15 Marks]

Sometimes it is necessary to perform arithmetic on integers larger than those that can be represented by Java's `long` data type. One way to do this for positive integers is to store them in arrays with one digit stored in each element (with the most significant digit stored in the first array element and the least significant digit stored in the last array element). For example, the number 152 could be stored in the array `x` by writing

```
int [] x = {1,5,2};
```

Complete the definition of a method named `addBig` that takes two `int` arrays as parameters and "adds" them, returning an array that contains the sum of the numbers stored in the input arrays.
Note: Your solution may return an array with a single leading '0'. You may assume that the elements of the input arrays are all single digits and that the input arrays have no leading zeros.

```
public static int [] addBig(int [] m, int [] n)
```

9. [15 Marks]

Assuming that binary trees are implemented as they were in class, complete the definition of the method `countVals` for the `Tree` class that returns the number of occurrences of `key` in a binary tree that stores integers in each node. If your answer requires more than one method, indicate clearly the class to which each method belongs.

```
public int countVals(int key)
```

10. [15 Marks]

We can call a binary tree "perfectly balanced" if, for every node in the tree, the node's left and right subtree have the same height. Write a method that returns `true` if a binary tree is perfectly balanced, and `false` otherwise. In your solution, you should not worry about the definition of the height of a subtree. Instead, you should simply assume the existence of a method `height` with header

```
    static int height (Node n)
```

that returns the height of a subtree with root `n`. If your answer requires more than one method, indicate clearly the class to which each method belongs.

Extra space *Please indicate clearly which question(s) you are answering on this page.*