UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE AND ENGINEERING

APS105F — Computer Fundamentals
Final Examination — December, 2007

Examiners: Wael Aboelsaadat, Tom Fairgrieve, W. James MacLean

Duration: 2.5 hours

- 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. **DO NOT** detach any pages from the exam paper.
- Rough work, if necessary, can be done on the backs of the pages.
- This examination has 14 pages (including the cover page).
- Read all instructions and write your name, student number, and ECF login on the first page **before** you start.
- You must use the C++ programming language to answer programming questions.
- While you are not required to comment any code you write in this exam, useful comments *may improve your mark* if it helps the marker better understand what you are attempting to do.
- Except for the true/false questions, we will award partial marks where possible. Attempt answers to all questions.
- Enjoy the holiday season!

Last Name _____ First Name _____

Student Number _____ ECF Login _____

MARKS

Question	1	2	3	4	5	6	7	8	**Total**
Value	20	15	5	10	10	10	10	10	**90**
Mark									

1. (20 Marks, 1 mark each)
   Circle the correct answer for each of the following:

   (a) **True** or **False**: A sorted linked list can be searched as quickly as a sorted array.

   (b) **True** or **False**: The C++ compiler ignores white-space in string constants.

   (c) **True** or **False**: After a call to the `open()` function in the `ofstream` class, it is safe to immediately start writing data to the file.

   (d) **True** or **False**: Variables must be initialized in a different statement from the statement in which they were declared.

   (e) **True** or **False**: The following statement will not generate a compiler error:
   `while(false);`

   (f) **True** or **False**: The following statements will cause `y` to contain the value 6:
   ```
   int x = 5 ;
   int y = x++ ;
   ```

   (g) **True** or **False**: If `a` and `b` are both `struct` variables of the same type, then the statement `a = b;` copies all the values stored in `b` into `a`.

   (h) **True** or **False**: The expression `42 - 4*(42/4)` gives the same result as `42 % 4`.

   (i) **True** or **False**: To use the C-string functions (*e.g.* `strlen()`) in your program, you need to include the line "`#include <string>`" at the start of your program.

   (j) **True** or **False**: A C++ function may return a `struct` value.

   (k) **True** or **False**: "Call-by-reference" means that any local variable in a C++ function can be accessed by any other function.

   (l) **True** or **False**: Binary search will operate properly on any array of values.

   (m) **True** or **False**: The function headers `void printArray(int *a)` and `void printArray(int a[])` are equivalent.

   (n) **True** or **False**: It is an acceptable practice to allocate memory using `new` without ever using `delete` to free the memory.

   (o) **True** or **False**: File permissions can be changed with the command `chmod`.

   (p) **True** or **False**: Recursive functions may return any type of value.

   (q) **True** or **False**: Given a large list of items, the selection sort algorithm almost always executes faster than the merge sort algorithm.

   (r) **True** or **False**: Recursive functions always execute faster than iterative functions.

   (s) **True** or **False**: Only functions that return a value may be recursive.

   (t) **True** or **False**: After executing the following code fragment, variable `ptr2` points to the same memory location as variable `ptr1`.

   ```
   int *ptr1, *ptr2;
   ptr1 = new int;
   ptr2 = &ptr1;
   ```

2. (15 Marks)

(a) i. (1 Mark)
What command would you use on the ECF lab computers to compile a file named `Lab9.c++` and have the executable program placed in a file named `Lab9`?

ii. (1 Mark)
What command would you use on the ECF lab computers to execute the program in a file named `Lab9` and have the output placed in a file named `Lab9.output`?

iii. (1 Mark)
What command would you use on the ECF lab computers to copy the file `Lab5Starter.c++` from the directory `/share/copy/aps105` into your current working directory?

(b) (2 Marks)
Write a single C++ *expression* that evaluates to a random three-digit integer. The leading digit must not be zero.

(c) (2 Marks)
Write a single boolean *expression* that tests a `char` variable `myChar` and determines whether or not it is a letter of the alphabet.

(d) (2 Marks)
Write C++ statements to declare a character array capable of storing a C-string of up to 100 characters, and read a value into it from the keyboard. The value input by the user may contain spaces, so read the entire line of input into your C-string.

(e) (2 Marks)
Given the declaration `int arr[25];`, show how to store the address of the second-last element of the array into a variable of type `int *` *without* using the `&` operator.

(f) (2 Marks)
Write a C++ statement that prints the ASCII code associated with the `char` variable `c` using `cout`. Do not declare any additional variables to do this.

(g) (2 Marks)
Rewrite the function:

```
void f( int &a )
{
   cout << a << endl;
}
```

so that it uses a call-by-value pointer variable instead of a call-by-reference `int` variable.

3. (5 marks)
Show what output is displayed when the following C++ program is executed:

```
#include <iostream>

using namespace std;

int f1(int n, int m)
{
    cout << "n= " << n << "  m= " << m << endl;
    if(n < m)
        return 0;
    else if(n==m)
        return m + f1(n-1,m);
    else
        return n + f1(n-2,m-1);
}

int main()
{
    cout << f1(5,4);
    return 0;
}
```

Output:

4. (10 Marks)
Write a complete C++ program that prompts the user to enter an integer number n and then prints all of the even squares between 1 and n. For example, if the user enters 150, the program should print the following:

```
4
16
36
64
100
144
```

5. (10 Marks)

(a) (5 Marks)
Write a C++ function that allocates memory for a 2-D array of `char` and returns a pointer to the array. The arguments to the function are integer values indicating the number of rows and columns in the array.

(b) (5 Marks)
Write a function named `isInOrder` that traverses a linked list, whose nodes are defined by the structure given below, and returns a boolean that indicates if the list is in alphabetical order. Do not copy any strings in your solution. Assume that all characters in `data` members are lower-case.

```
struct ListNode
{
  char data[20];
  ListNode *next ;
};
```

6. (10 Marks)
A known limitation of simple binary search is that, if it returns a valid index, the caller does not know if the returned index is for the first, last, or some other occurrence of the value sought in the given array. You are to write a new function called `bSearchAll` that returns the indices of the first and last occurrences of the value sought in a `struct` of type `IndexRange`, defined as

```
struct IndexRange
{
  int firstIndex ;
  int lastIndex  ;
};
```

The struct returned will store the index of the first occurrence in `firstIndex` and the index of the last occurrence in `lastIndex`. If only one instance is found, both index values will give the index of that instance. If none are found, both index values will contain `-1`. Assume you already have a function that performs binary search, with declaration

```
int bSearch(const int a[], const int size, const int key);
```

Call the `bSearch()` function in `bSearchAll()` to find an occurrence of the sought-after value (`key`) in an array `a` that has `size` elements. The `bSearch()` function returns an index of the sought-after value in the given array, or `-1` if the value is not found.

Continue your solution to question 6 on this page.

7. (10 Marks)
Recall from class that a stack can be thought of as a linked list where all additions (*push*) and all deletions (*pop*) are done at the same end of the list, and this end is called the *top* of the stack. In this question you are to implement a simple stack using linked lists (the type of lists you used in Lab 8). Your stack will store `double` values for use in a calculator program. You are to write the following functions:

`bool isEmpty(StackNode *head)` — this function returns `true` if there are no nodes in the stack, and `false` otherwise.

`double peek(StackNode *head)` — this function returns the `double` value stored in the node at the top of the stack, but does not change the stack. Although this function should not be called on an empty stack, use the `assert()` function to cause the program to stop should this condition be violated.

`void push(StackNode *&head, double x)` — this function adds a node storing the value in `x` to the stack.

`double pop(StackNode *&head)` — this function returns the `double` value stored in the node at the top of the stack, and removes the corresponding node from the stack. Be sure to return any allocated memory when you are done using it. Although this function should not be called on an empty stack, use the `assert()` function to cause the program to stop should this condition be violated.

As part of your answer, show the declaration for `StackNode` that will allow a suitable implementation of the stack functions above.

Hint: while you are free to choose which end of the list to use for additions and deletions, making the right choice will make this question simpler.

Continue your solution to question 7 on this page.

8. (10 Marks)
A prime number is a positive integer that has exactly two distinct positive integer divisors, namely 1 and itself. The first 5 prime numbers are: 2,3,5,7,11.

The Greek mathematician Eratosthenes devised a scheme for determining all prime numbers less than or equal to a given integer N. An algorithmic description of his idea follows:

- Create an array that stores the integers (in order) from 0 to N. If $N = 15$, the array would be:

```
        [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]
        ---------------------------------------------------------------------
arr | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
        ---------------------------------------------------------------------
```

- Eliminate 0 and 1 from consideration as prime numbers, perhaps by setting their array entries to -1.
- Starting with the first prime (2), eliminate further multiples of this prime from consideration (perhaps by setting the array entries for the multiples to -1) because they are not prime numbers.
- The next prime will be the first integer following the current prime that has not been eliminated.
- Multiples of this prime are then eliminated, and again the remaining integers are searched for the next prime.
- The algorithm continues in this fashion until all numbers less than or equal to N that are not prime numbers have been eliminated.

Write a complete C++ program that reads a number N from the user, applies Eratosthenes' algorithm, and then prints the prime numbers less than or equal to N.

Continue answers to any questions here if necessary.
Clearly indicate which question the answer belongs to.

Continue answers to any questions here if necessary.
Clearly indicate which question the answer belongs to.