University of Toronto
Faculty of Applied Science and Engineering

ECE 244F

PROGRAMMING FUNDAMENTALS

Fall 2016

Midterm Test

Examiners: T.S. Abdelrahman and D. Yuan

Duration: 110 minutes

This test is OPEN Textbook and CLOSED notes. The use of computing and/or communicating devices is NOT permitted.

Do not remove any sheets from this test book. Answer all questions in the space provided. No additional sheets are permitted.

Work independently. The value of each question is indicated. The total value of all questions is 100.

Write your name and student number in the space below. Do the same on the top of each sheet of this exam book.

Name: _____
(Underline last name)

Student Number: _____

Q1. _____	**Q7.** _____
Q2. _____	**Q8.** _____
Q3. _____	**Q9.** _____
Q4. _____	**Q10.** _____
Q5. _____	**Q11.** _____
Q6. _____	**Q12.** _____

Total

Question 1. (10 marks). *General.*

(a) What Unix command would you type to create a new directory called `ece244` in the current working directory?

Write the command here:

(b) What is the name of the Integrated Development Environment (IDE) software tool you were asked to use in lab assignment 1 to compile, run and debug your code?

Write the name here:

(c) Every C++ program must contain exactly one global function named `main`.

Circle one answer: Yes No

(d) The statement: `delete p`; de-allocates the pointer `p`.

Circle one answer: Yes No

(e) Every class has at least one constructor function, even when none is declared.

Circle one answer: Yes No

(f) A `stringstream` variable can be passed by value in function calls.

Circle one answer: Yes No

(g) It is safe for a function to return a pointer to a local variable declared and used inside the function.

Circle one answer: Yes No

(h) Which of the following statements is correct? Circle the correct statement.

1. A constructor is called when an object is declared.
2. A constructor is called when an object is used.
3. All of the above.
4. None of the above.

(i) How many destructors may a class have? Circle the correct answer.

1. Zero.
2. One.
3. Any number as defined by the programmer.

(j) A default version of the assignment operator, `operator=,` is created automatically for a C++ class if the class does not define one.

Circle one answer: Yes No

Question 2. (9 marks). *Parameter Passing and Scopes.*

Consider the following program.

```
int z = 0;

void f1(int x, int y) {
  x++;
  y++;
  z++;
}

void f2(int &x, int &y) {
  x++;
  y++;
  z++;
}

void f3(int *x, int *y) {
  (*x)++;
  (*y)++;
  z++;
}

int main() {
  int x = 0, y = 0;
  f1(x, y);
  cout << "x = " << x << ", y = " << y << ", z = " << z << endl;

  f2(x, y);
  cout << "x = " << x << ", y = " << y << ", z = " << z << endl;

  f3(&x, &y);
  cout << "x = " << x << ", y = " << y << ", z = " << z << endl;
  return 0;
}
```

In the space provided below, write the output that an execution of the above program would produce in the order in which it is produced. Use one entry in the table for each line of output produced. You may or may not need all entries in the table.

Question 3. (5 marks). *Scopes.*

The following class definition describes a simple C++ class called `EmptyClass`.

```
#include <iostream>
using namespace std;

class EmptyClass {
private:
        int myValue;
public:
        EmptyClass();
        EmptyClass(int v);
       ~EmptyClass();
};

EmptyClass::EmptyClass() {
       myValue = 0;
       cout << "Constructing " << myValue << endl;
}

EmptyClass::EmptyClass(int v) {
       myValue = v;
       cout << "Constructing " << myValue << endl;
}

EmptyClass::~EmptyClass() {
       cout << "Destructing " << myValue << endl;
}
```

Now consider the following code, which uses `EmptyClass`:

```
void doNothing() {
        EmptyClass a[2];
        return;
}


EmptyClass* doSomething() {
        EmptyClass* a = new EmptyClass(2);
        doNothing();
        return a;
}


int main() {
        EmptyClass a(3);
        if ((2 + 2) == 4) {
           EmptyClass b;
           EmptyClass* a = doSomething();
           delete a;
        }
        return (0);
}
```

In the space provided below, write the output that an execution of the above program would produce in the order in which it is produced. Use one entry in the table for each line of output produced. You may not need all entries in the table.

Question 4. (10 marks). *Functions and Objects.*

Assume there exists four classes: `Charmeleon`, `Weedle`, `Spearow` and `Oddish`. These classes are implemented correctly and are available for use.

(a) Write the declaration (prototype) of a function called `trainer`. The function is member of the class `Charmeleon`, takes one argument, called `s`, that is an object of type `Spearow`, passed by value, and returns no value.

(b) Write the declaration (prototype) of a function called `sonic`. The function is member of the class `Spearow`, takes one argument, called `w`, that is an object of type `Weedle`. The function must modify this object. The function returns by value an object of type `Oddish`.

(c) Write the declaration (prototype) of a function called `slayer`. The function is member of the class `Oddish`, takes one argument, called `p`, that is a pointer to an object of type `Spearow`, allocates an object of type `Spearow`, and modifies it argument to store the address of this object. The function returns a boolean value.

(d) Write the declaration (prototype) of a function called `sonic`. The function is not a member of any class (i.e., a global function), takes no arguments and returns a pointer to an object of type `Weedle`.

(e) Write the declaration (prototype) of a function called `slayer`. The function is member of the class `Weedle`, takes two arguments: a pointer to an object of type `Oddish`, called `p`, and an object of type `Spearow`, called `s`, passed by reference. The function returns by reference an object of type `Oddish`.

Question 5. (6 marks). *Classes.*

The following code contains several compile-time errors. Identify these lines of code with errors and list them in the table shown on the next page, along with a short explanation of what the error is. To make it easy to refer to lines of code, lines numbers are shown (they are not part of the code). Where a line is not numbered, you may assume it is error-free.

You may not need all the rows of the table. Should there be multiple errors on a line, use multiple entries in the table. Attempt to list the errors in increasing line number.

```
   class aBadClass {
     private:
01     int x;
02     int negate();

     public:
03     float y;
04     aBadClass(int i, float f);
05    ~aBadClass(int k);
06     void evenWorse(const float f);
       void print() const;
   };

07 aBadClass::aBadClass(int i, float f) {
08   x = i;
09 }

10 void aBadClass::~aBadClass (int k) {
11    x = k;
12 }

13 void aBadClass::evenWorse (const float f) {
14    y = (float) -1*x;
15    f = f/y;
   }

16 void aBadClass::print() const {
17    cout << x << end;
18    x = -x;
   }

19 int main () {
20   aBadClass a;
21   aBadClass b(1,2.5);
22   aBadClass c(1,3.5);
23   c.x = 2;
24   c.y = 5.8;
25   c.negate();
26   c.evenWorse(6.8);
     return (0);
   }
```

Line Number	Explanation of Error

Question 6. (6 marks). *Object Initialization.*

Consider the following declaration of a class, stored in a file called `LabMark.h`, which represents a lab mark of a student. You may assume that this class is implemented correctly and is **error-free**.

```
using namespace std;
#include <iostream>

class LabMark {
   private:
      int studentNumber;
      int mark;

   public:
      LabMark ();
      LabMark (int d, int m);

      int getStudentNumber ();
      int getMark();

      void setStudentNumber(int d);
      void setMark(int m);

      void print();
};
```

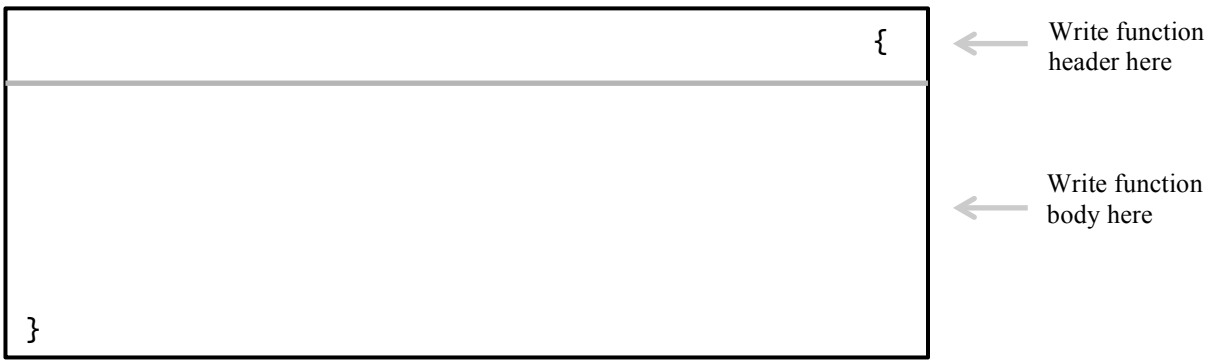
We wish to be able to write the following code in the function `main`.

```
#include <iostream>
#include "LabMark.h" // This contains the class definition above
#include <string>
#include <sstream>

using namespace std;

int main () {
    string str;
    str = "0123456789 8"; // Student number followed by mark
                          // Assume no errors in str
    LabMark aStudent(str);
    aStudent.print();
    return (0);
}
```

The above code requires that one member function be added to the class. Write this function in the space below. Your answer should not exceed a few lines of code.



Question 7. (8 marks). *Constructor/Destructor.*

Consider the following program.

```
01 class Box
02 {
03 private:
04   int length;
05   int breadth;
06   int height;
07 public:
08   Box() {                              // Call this Constructor 1
09     length = breadth = height = 0;
10   }
11   Box (int _length) {                  // Call this Constructor 2
12     length = _length;
13     breadth = height = 0;
14   }
15   Box (int _l, int _b, int _h) {       // Call this Constructor 3
16     length = _l;
17     breadth = _b;
18     height = _h;
19   }
20   int getLength () { return length; }
21   int getBreadth () { return breadth; }
22   int getHeight () { return height; }
23   ~Box () {
24     cout << "Box is deleted." << endl;
25   }
26 };
27
28 void package (Box& b1, Box& b2) {
29    Box t(b1.getLength() + b2.getLength(),
30          b1.getBreadth() + b2.getBreadth(),
31          b1.getHeight() + b2.getHeight());
32
33 }
34
35 int main () {
36   Box a(2);
37   Box b(3);
38   Box c;
39   Box boxes[5];
40   Box *p = NULL;
41   p = new Box(1, 2, 3);
42   Box *q[3];
43   package (a, b);
44   delete p;
45   return 0;
46 }
```

This question asks you when the constructors and destructor are being invoked. In the table below, identify the line number of each statement that will cause constructors or destructor being invoked. This is an example of how to write your answers:

Line Number	Invocation of constructors/destructor
50	Constructor 2 x 3
60	Destructor x 5

The first entry indicates that the execution of the statement at line 50 will cause "Constructor 2" to be invoked 3 times; the second entry indicates that the execution of the statement at line 60 will cause the destructor to be invoked 5 times. You do not need to use all entries in the table.

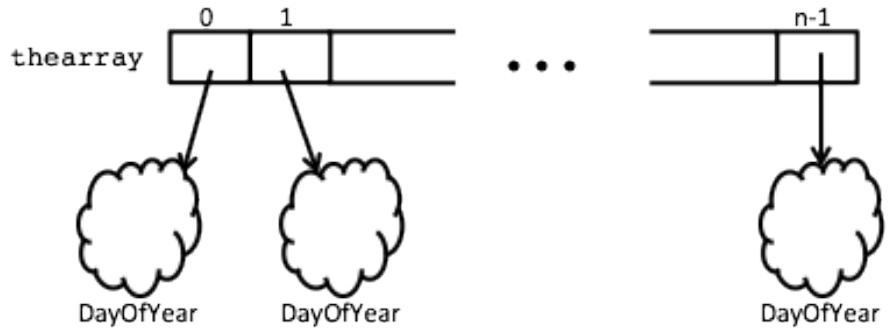
Line Number	Invocation of constructors/destructor

Question 8. (10 marks). *Arrays and Objects.*

Consider the following definition of the class `DayOfYear`. Assume it has been correctly implemented.

```
class DayOfYear {
   private:
       int day;
       int month;
   public:
            DayOfYear();             // The default constructor
            DayOfYear(int d, int m); // The second constructor
           ~DayOfYear();
       int getDay();
       int getMonth();
       void setDay(int d);
       void setMonth(int m);
       void print();
};
```

We wish to allocate the data structure shown below, *dynamically* at run time. Given the input `n` from the user, where `n` is a positive integer greater than 1, we wish to dynamically allocate an `n`-element array, called `thearray`. Each element points to a dynamically allocated object of type `DayOfYear`.



Based on this, please answer the following questions.

(a) (1 mark). Give the declaration of the variable `thearray`.

(b) (2 mark). Write lines of code to dynamically allocate the array `thearray`.

(c) **(1 mark)**. Which constructor is used to initialize each of the array elements in the declaration above? Write the name of the constructor (see class definition above), or NONE if no constructor is invoked.

(d) **(3 marks)**. Write code to allocate `n` objects of type `DayOfYear` and have each element of `thearray` point to one object. Each object should have its `day` and `month` initialized to the same value: the index of the array element that points to the object. As an example, the object that is pointed to by the array element at index 5 should have its `day` and `month` members initialized to 5.

(e) **(1 mark)**. Given a value `t` that is between 0 and n-1, write code to change the value of the `day` member of the object pointed to by the element `t` of the array to 1.

(f) **(2 marks)**. Write C++ code to de-allocate the objects and array allocated in the parts above so that no memory leaks exist.

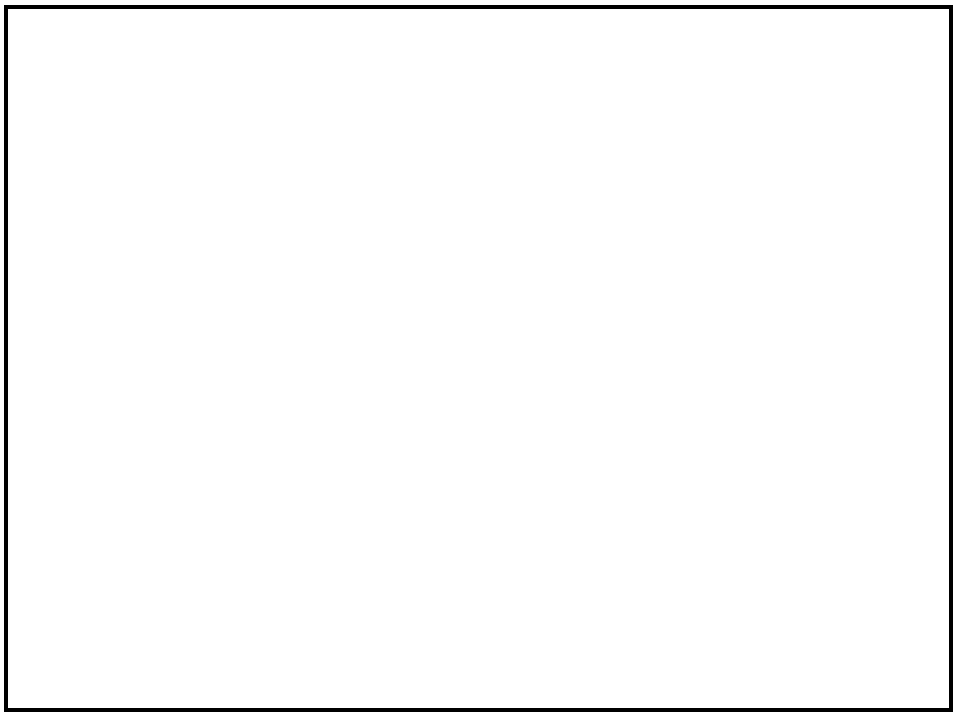
Question 9. (10 marks). *C++ I/O.*

The following program reads 10 integers from the standard input until either an invalid integer is given or the end-of-file is reached. The 10 integers appear on a single input line. Re-write the program so it used a `stringstream` for input rather than `cin`.

Hint: use `getline` as you did in the lab assignments. You should keep the same code as much as possible, making modifications to use a `stringstream`.

```
#include <iostream>
using namespace std;
int main() {
  int i;
  int value;
  for (i=0; i < 10; ++i) {
     cin >> value;
     if (cin.fail()) {
        if (cin.eof()) {
           cout << "More integers expected." << endl;
           return (0);
        }
        else {
           cout << "Invalid integer." << endl;
           return (0);
        }
     }
     cout << "Integer value is: " << value << endl;
  }
   return (0);
}
```

Write your code in the box below.



Question 10. (8 marks). *Operator Overloading.*

The following class is used to create objects that represent complex numbers, similar to the one described in class.

```
using namespace std;
#include <iostream>

class complex {
  private:
        float real;
        float imag;
   public:
        complex();
        complex(float r,float i);

        float getReal() const;
        float getImag () const;
        void  setReal(float r);
        void  setImag(float i);
        complex operator+ (const complex & rhs) const;
        complex operator- (const complex & rhs) const;
        complex operator* (const complex & rhs) const;
        complex operator/ (const complex & rhs) const;
        bool operator== (const complex & rhs) const;
        void print() const;
};
```

We wish to overload the "+=" operator for the `Complex` class to be able to write code like this in a non-member function (say `main`):

```
Complex X(0.1,8.3);
Complex Y(4.7,6.1);
   :
X += Y;
```

The overloaded operator adds the value of `X` to that of `Y` and then assigns the result to `X`, similar to the += operator for data types such as `int`s or `float`s.

Write the implementation of the overloaded `operator+=` function, as a member of `Complex`. Clearly show the function header and its body. Write your answer in the box in the next page.

Write the overloaded `operator+=` function as a member of `Complex` below

Question 11. (10 marks). *Pointers and Structures*.

You are to complete the implementation of code that represents a line in a two-dimensional plane. The skeleton of the code is given to you below. The code uses pointers and `struct`s. Namely, the declarations of the two structures, `struct point` and `struct line`, are provided to you. You will need to complete the implementation of the function `create_line`. Here are the requirements:

- `create_line` takes four parameters, representing the `x` and `y` coordinates of the end point 1 and the X and Y coordinates of the end point 2. This function should dynamically allocate memory space to represent the line and points. You have to use `struct line` and `struct point` to represent the line and point.

- In `create_line`, you are not allowed to declare any additional variables other than `p` and `q`. In addition, you cannot use `p` in the code you write, i.e., you can only use `q` in your code.

- At the end of `create_line`, all of the dynamically allocated memory space has to be deleted. That is, at the end of `create_line`, there should be no memory leak.

- `create_line` should output one line:

  "`The line is from (srcX,srcY) to (dstX,dstY).`"

  Where `srcX`, `srcY`, `dstX`, `dstY` are coordinate values of the two points.

- You do not need to worry about any error handling. For example, you do not need to check that the input values are valid coordinates.

An example invocation of `create_line` is provided in the main function. This line will cause `create_line` to output: "`The line is from (2,3) to (7,8).`"

```
struct point {
  int *x;
  int *y;
};


struct line {
  struct point *x;
  struct point *y;
};
```

```
void create_line (int srcX, int srcY, int dstX, int dstY) {
  struct line *p;
  struct line **q = &p;
  /* Write your code here. Remember, you cannot use p. */









}


int main () {
  create_line(2,3,7,8);
  return 0;
}
```

Question 12. (8 marks). *Program Organization and Compilation*.

(a) (4 marks). Suppose you design two classes: `MajorClass` and `MinorClass`. For each of these classes, you have a definition file and an implementation file. Thus, you have four files: `MajorClass.h`, `MajorClass.cpp`, `MinorClass.h` and `MinorClass.cpp`. Also you write a program `main.cpp` that uses the two classes. The files have been successfully compiled into a single executable `main.exe`.

Now suppose after the files have been compiled, you edit `MajorClass.cpp` to change its functionality. What are the compile commands needed so that you can reflect the changes made to `MajorClass.cpp` in the executable `main.exe`. Your commands must compile as few files as possible. Assume you are using the `g++` compiler.

Write the commands here:

(b) (4 marks). Consider the program below, organized into two `.h` files and three `.cpp` files.

Mystery.h

```
#ifndef MYSTERY_H
#define MYSTERY_H


class Mystery {
private:
   double m;
public:
   Mystery();
};

#endif
```

Mystery2.h

```
#ifndef MYSTERY2_H
#define MYSTERY_H

class Mystery2 {
private:
   double m2;
public:
   Mystery2();
};

#endif
```

main.cpp

```
#include "Mystery.h"


int main () {
   Mystery myMystery;
   Mystery2 myM2;
   return (0);
}
```

Mystery.cpp

```
#include "Mystery.h"
#include "Mystery.h"

Mystery::Mystery ()
{
}
```

Mystery2.cpp

```
#include "Mystery2.h"
#include "Mystery2.h"

Mystery2::Mystery2 ()
{
}
```

The program is compiled with the command:

```
g++ Mystery.cpp Mystery2.cpp main.cpp –o prog.exe
```

1. Two compiler errors are generated. Describe in one sentence the nature of each of these errors. Be very specific.

Error 1:

Error 2:

2. Modify some of the files shown above eliminate the errors. You can add, delete or modify lines in any of the 5 files. You should make only the minimal modifications necessary in order to make the program compile properly. Show your answer directly on the code in the boxes above.