CMPT 135: Sample Final

Last name *exactly as it appears on your student card*									**First name** *exactly as it appears on your student card*
Student Number									
SFU Email					**Section** if you know it!				

This is a **3 hour** test. It is **closed book**: no calculators, computers, notes, books, etc. are allowed. During the exam, do not look anyone else's exam, or allow anyone else to see your exam, or speak to any other students.

Important: Do **not** use any C++ library functions unless a question specifically permits it. Also, use only features of C++ discussed in the lectures and lecture notes.

Question	Out Of	Your Mark
Arrays and Pointers	10	
Classes and Objects	10	
Short Answers	10	
Inheritance	10	
Exceptions	10	
Recursion	8	
Algorithms	10	

Total	**68**	

Instructor: Toby Donaldson

Arrays and Pointers

(10 marks) Each correct answer is worth 1 mark; incorrect answers, or unanswered questions, are worth 0 marks.

Question	Your Answer
Write a `cout` statement that prints the address of variable `a`: `int a = 3;`	
Suppose `p` is a pointer to an `int`. Write a `cout` statement that prints the `int` `p` points to.	
True or *false*: the following code fragment does **not** compile: `int x = new int(5);` `cout << x;`	
Suppose `m` is a pointer to a `double` on the free store. Write a statement that deletes the memory `m` points to.	
Suppose `arr` is a pointer to a `double` array on the free store. Write a statement that deletes the memory `arr` points to.	
True or *false*: the following code fragment does **not** compile: `string s = "cat";` `string t = "dog";` `string* a = &s;` `string* b = &t;` `a = b;` `b = a;`	
True or *false*: the following code fragment causes a **memory leak** if executed: `string s = "cold";` `string* p = &s;` `p = nullptr;`	
True or *false*: the following function compiles, and has **no** memory leak or other run-time error: `void f() {` `int a = 5;` `int* p = &a;` `cout << a;` `delete p;` `}`	
True or *false*: it is an error to delete a pointer whose value is `nullptr`	
Suppose `arr` is an array of 10 `int` values all initialized to 0. What does `cout << arr[10]` print?	

Classes and Objects

(10 marks) Each correct answer is worth 1 mark; incorrect answers, or unanswered questions, are worth 0 marks.

Question	Your Answer
True or *false*: every object has at least one (possibly empty) constructor.	
True or *false*: every object has at least one (possibly empty) destructor.	
True or *false*: **initialization lists** can be used with any method in an object.	
True or *false*: a **default constructor** takes no inputs.	
True or *false*: by default, methods and variables in a `class` are `private`.	
True or *false*: an object's destructor is called automatically when the object goes out of scope, or is deleted.	
True or *false*: a class can define more than one constructor.	
True or *false*: a class can define more than one destructor.	
True or *false*: if you create a class called `Fraction` to represent fractions, then you can define a custom `operator+` for adding `Fraction` objects.	
True or *false*: all objects are classes, but not all classes are not objects.	

Short Answers

Question	Answer
a) (1 mark) What is the general name (not g++!) of the program that converts a C++ source code file (e.g. a .cpp file) into object code?	
b) (1 mark) What is the general name (not g++!) of the program that converts a C++ object code file into an executable file?	
c) (1 mark) What is the usual file name extension for C++ header files?	
d) (2 marks) Write a complete C++ program that prints "Hello, world!" on `cout` and does **not** have a `using` statement.	
e) (1 mark) *True* or *false*: in the worst case, **linear search** has to do 1000 comparisons when searching through a vector of n=1000 numbers.	
f) (1 mark) *True* or *false*: in the worst case, **binary search** only works on sorted data.	

g) (1 mark) *True* or *false*: it's usually faster to do a linear search on a vector of n numbers than it is to first sort that data and then do a binary search on it.	
h) (1 mark) When sorting n numbers using **insertion sort** (the sorting algorithm discussed in the class), about how many comparisons does it do in the worst case?	
i) (1 mark) Suppose you are using **linear search** to look for x in a vector of n numbers. What is the smallest number of comparisons linear search might need to do to find x?	

Inheritance

Consider the following class:

```
class PQueue {
public:
    virtual ~PQueue() { }

    virtual void insert(int x) = 0;
    virtual void remove_min() = 0;
    virtual int peek() const = 0;

    virtual int pop() {
        int result = peek();
        remove_min();
        return result;
    }
}; // class PQueue
```

Question	Your Answer

(1 mark) *True* or *false*: The following line of code causes a compiler error:
 	
`PQueue pq;`
`// ... pq used ...`	

(1 marks) What is the name we use for a class, such as `PQueue`, where all the methods are `public` and `virtual`, and at least one method is `=0`?	
(2 marks) Explain what `=0` at the end of some method headers means here.	

(2 marks) Explain what the `virtual` keyword means here.	
(2 marks) Why does `PQueue` include a **virtual destructor**?	
(1 mark) Define a new class named `Heap` that derives (i.e. inherits) from `PQueue`. You don't need to implement any methods or variables: just show how to do the inheritance in the class header line.	

(1 mark) Suppose you've (correctly!) written the `Heap` class from the previous question, and it has a default constructor. *True* or *false*: this code compiles:
 	
`PQueue* p = new Heap();`
`// ... p used ...`	

Exceptions

(10 marks) Suppose you have a function with the following header:

```
AST parse_json(const string& s);
```

`parse_json` takes a string as input returns a new object of type `AST`. It can be used like this:

```
// input is a string that has been defined earlier
AST tree = parse_json(input);
```

Suppose you know that `parse_json` could, potentially throw an exception. Re-write the above line of code so that if `parse_json` throws:

- `std::invalid_argument`, then "invalid argument" is printed to `cout` (and nothing else is printed)
- `std::out_of_range`, then "out of range" is printed to `cout` (and nothing else) is printed
- any other kind of exception, then "unknown error" is printed to `cout` (and nothing else is printed)
- no exception, then "ok" is printed to `cout` (and nothing else is printed)

Recursion

a) (5 marks) Write a function that uses recursion (and no loops or library functions, other then the standard C++ `string` class) to make a function called `repeat(s, n)` that returns a string consisting of `n` copies of the string `s`. For example:

`repeat("ha", 3)` returns `"hahaha"`
`repeat("pow!", 4)` returns `"pow!pow!pow!pow!"`
`repeat("pow!", 0)` returns `""`
`repeat("", 10)` returns `""`

If n <= 0, then `repeat(s, n)` returns the empty string `""`.

b) Consider this function:

```
void a() {
    cout << "Hello!\n";
    a();
}
```

This compiles in C++, and it may, or may not, run forever when executed.

i) (2 marks) Explain why and in what circumstances it might crash.

ii) (1 mark) In what circumstances might it run forever?

Algorithms

(10 marks) Write a function called `shortest(v)` that returns the **shortest** string in `v`, which is a `vector<string>`. If two or more strings are tied for the shortest, then any one of them can be returned. If `v` is empty, then use `cmpt::error` to cause an error. Make your function efficient --- don't do any unnecessary copying of strings or vectors.