CMPT 135 Midterm

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 **50 minute** test. It is **closed book**: no calculators, computers, notes, books, etc. are allowed.

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	8	
Pointers and Dynamic Arrays	10	
Classes and Objects	20	

Total	**38**	

Arrays

a) (3 marks) Write a fragment of C++ code that creates a variable named `temps` that is of type array of `double` (*not* a pointer!). Make it of length 500, and you use a loop to initialize all its value to 0.

b) (5 marks) Write a function that calculates and returns the sum of all the elements in *any* array of `double`s. The passed-in array should be of type array of `double` (*not* a pointer!). Write both the function header and its body.

Pointers and Dynamic Arrays

a) (1 mark) Write a fragment of C++ code that defines a variable `x` to be a `double` with the value 5, and then defines a pointer `p` that points to `x`.

b) (2 marks) Write a fragment of C++ code that creates a new array of 150 `double`s on the free store, and then immediately afterwards de-allocates that array.

c) (2 marks) Suppose `arr` points to an array of 150 `double`s on the free store. Write a fragment of C++ code that prints each element `arr` to the screen. **Important**: your code fragment must access the elements of `arr` **without** using []-notation anywhere.

d) (5 marks) Write a function called `make_fill(n, val)` that returns a pointer to a newly created array of `double`s of length `n`. Each element of the returned array should have the value `val`. If `n` is less than 0, then cause an error using `cmpt::error`.

Classes and Objects

(20 marks) Write a class called `Student` that stores the name (as a `string`) and `age` (as an `int`) of a university student. You class must have these features:

- All class variables are **private**, and all methods are **public**.
- A **default constructor** that uses an **initialization list** to set the student's name to "none" and age to -1.
- A **constructor** that uses an **initialization list** to set `name` and `age` to values passed into the constructor.
- A **copy constructor** that uses an initialization list to set the student's name and age to be the same as the name and age of another passed-in `Student` object.
- A **destructor** that prints the message "object deleted" when the `Student` object it's part of goes out of scope, or is deleted.
- A **getter method** that returns the name of the student, and another **getter method** that returns their age. Make sure these can be used with constant `Student` objects.
- Define an `<<` operator that lets you print the name and age of a `Student` object. Importantly, define this `<<` *outside* of the `Student` class. No special format is required: print the name and age in any convenient way.