CMPSC 32 S18
Object Oriented Design and Implementation

Midterm 1 Examination

Please state your answers as clearly as possible. This exam not only tests your understanding of the material, but also how well you can convey your understanding to us. Remember that you are solely responsible for the answers to the questions, therefore, please refrain from consulting with your class peers.

Please write all your answers **LEGIBLY** and **CLEARLY**. If we cannot decipher your answers, you will not receive credit.

No electronic devices are allowed during the exam (calculators, cell phones, laptops, etc.).

READ all questions carefully before attempting to answer. If there are any ambiguities in the statement of questions, please ask us. **You may assume that each problem is correct and solvable unless the question specifically asks about errors.**

THE GRADE IN THIS EXAM IS A TOTAL OF 62 POINTS.

Name (as it would appear on the official course roster)	**Umail Address**
	@umail.ucsb.edu

Question 1 (11 points)

a. Given the following C++ statements, write one line of C++ code that removes the "1" value from the vector (you may assume all necessary STL components are included).

```
vector<string> v; v.push_back("0"); v.push_back("1"); v.push_back("2");
```

b. What are the three main steps of the C++ build process? For each step, **briefly** state what the main purpose of the step is.

c. **Briefly** state what is the main difference between a `struct` and a `class` in C++?

Question 2 (10 points)
Given the following two class definitions:

```
class X {                                         class Y {
       short a;                                          char a;
       char b;                                           char b;
       char c;                                           int c;
       int d;                                            short d;
       int e;                                            int e;
};                                                };
```

a. Draw out the memory structure g++ would allocate for an instance of each class. You may assume one read operation fetches 4 bytes of data. **Clearly illustrate** how the variables and their sizes are allocated in memory, and any memory padding g++ does.

b. What is the size (in bytes) of an instance for class X?

c. What is the size (in bytes) of an instance for class Y?

Question 3 (6 points)
Assume we have a program containing several .cpp and .h files. The files that our program requires are:

`main.cpp, x.cpp, x.h, y.cpp, y.h, z.cpp, z.h`

Also assume that `main.cpp` contains the main function and includes `x.h`, `x.cpp` includes `y.h`, and `y.cpp` includes `z.h`. For parts a and b, write Makefile rules for the following targets. Write the rules as it would appear in the Makefile.

```
a. # main target generates an executable that depends on intermediate object files.
   # Include any dependencies and a single command line in your solution below.

main:
```

```
b. # clean target removes object files (.o) and executable (main) in current directory.
   # Include any dependencies and a single command line in your solution below.

clean:
```

Question 4 (10 points)

Complete the definition of the **insertion sort** function. `int a[]` is an integer array to be sorted **in-place** from least to greatest, and `size_t size` is the size of the array `a[]`.

```
void insertionSort(int a[], size_t size) {













}
```

Question 5 (11 points)

a. What is the O-notation for the **best-case scenario** for bubble sort (assuming the implementation contains the optimization discussed in lecture)? **Briefly explain why.**

b. What is the O-notation for the **best-case scenario** for insertion sort? **Briefly explain why.**

c. Similar to what you did in h07, show the state of values for each iteration of selection sort in the table below. The sorted elements should be from least to greatest.

Values / Iterations	10	5	2	3	8
i = 4					
i = 3					
i = 2					
i = 1					

Question 6 (14 points)

For the following class interface representing a House object, provide the definitions for `house.cpp` for the constructors and class methods.

```
// house.h
class House {
public:
	// constructor that initializes the size and price of a house to 0.
	House();
	// Copy constructor
	House(House& house);
	// returns the size of the House
	int getSize() const;
	// returns the price of the House
	double getPrice() const;
	// sets the size of the House
	void setSize(int size);
	// sets the price of the House
	void setPrice(double price);
	// Increases the price of the house by the parameter value representing
	// the percentage increase. For example, 0.01 represents a 1% increase.
	// This function sets the house price to the new value and returns the updated
	// value.
	double increasePrice(double percentage);
private:
	int size; // represents square footage of the house
	double price; // represents the price of the house
};
-------
// house.cpp
```