CMPSC 32 W19
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 46 POINTS.

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

Question 1 (6 points)

Write whether each statement is True or False. If False, <u>briefly</u> state why (1 point each)

a. C++ will not provide a default copy constructor for a class if one is not explicitly defined.

b. A class' members are automatically set to private unless specified otherwise.

c. The best-case running time for the selection sort implementation done in lecture is O(n).

d. Assuming `class X` has been defined in your code, then the statement `X* arr[10];` will not compile if `class X` does not have a default constructor explicitly defined.

e. The size of an empty class or structure (i.e. they do not contain any members) is 0 bytes.

f. Object files (.o files) are generated by the Preprocessing step in the C++ build process.

Question 2 (5 points)

Given an incomplete definition of selectionSort as implemented in lecture, fill in the blanks with a constant, variable, or expression such that the function will work correctly.

```
void selectionSort(int a[], size_t size) {
      int temp;
      int largestIndex;
      int largest;

      for (int i = ___________________; i ___________________ 0; i--) {

            largestIndex = ___________________;

            largest = ___________________;

            for (int j = 1; j <= ___________________; j++) {

                  if (a[j] ___________________ largest) {

                        ___________________ = a[j];

                        ___________________ = j;
                  }
            }
            temp = a[i];

            a[i] = ___________________;

            a[largestIndex] = ___________________;

      }
}
```

Question 3 (4 points)

a. **Briefly** explain the two main errors when writing a recursive function that will result in a stack overflow.

b. **Briefly** define the purpose of the C++ compiler **`-o`** and **`-c`** flags.

Question 4 (4 points)
Given the following namespace definitions and main function, write the output in the blank space to the right of each statement. If the code does not compile, simply write ERROR. You may treat each line independently from each other.

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

namespace A {
    void f() { cout << "A.f()" << endl; }
    void g() { cout << "A.g()" << endl; }
}
namespace B {
    void f() { cout << "B.f()" << endl; }
}
void f() { cout << "f()" << endl; }
void g() { cout << "g()" << endl; }
using namespace std;
using namespace B;

int main() {

    f();          _________________________________

    g();          _________________________________

    A::f();       _________________________________

    A::g();       _________________________________

    B::f();       _________________________________

    B::g();       _________________________________

    ::f();        _________________________________

    ::g();        _________________________________

}
```

Question 5 (5 points)
Consider the following class definition:

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

Write the output for each `cout` statement in the blank spaces for the following code segment:

```
X a;
cout << &a << endl;                              ___________0xb34144__________

cout << reinterpret_cast<void*>(&a.a) << endl;   _____________________________

cout << &a.b << endl;                            _____________________________

cout << &a.c << endl;                            _____________________________

cout << reinterpret_cast<void*>(&a.d) << endl;   _____________________________

cout << &a.e << endl;                            _____________________________
```

Question 6 (6 points)

a. Write the function `sumVector` that takes in a vector of ints and returns the sum of all of the vector's values. Your solution **must explicitly use `std::iterators`** in order to access the elements in the vector. You may assume that `namespace std` is being used in your file. Your solution must be syntactically correct in order to receive full credit.

```
int sumVector(vector<int> &v) {








}
```

Question 7 (4 points)

Given an initial array of values, write the state of the array after each iteration using the **bubbleSort** algorithm discussed in lecture. You may assume that the array will be sorted in ascending (least-to-greatest) order.

Values **Iterations**	**8**	**5**	**2**	**4**	**1**
i = 4					
i = 3					
i = 2					
i = 1					

Question 8 (12 points)

Given a class interface file (.h file) for a Person as discussed in class.

```
class X {
private:
    vector<string>* v;
public:
    X() { v = new vector<string>(); }
    ~X();                                            // implement destructor
    X(const X& obj);                                 // implement copy constructor
    vector<string>* getVector() const { return v; }
};
```

a. Implement the destructor and the copy constructor in the space below such that a deep copy is done. Note that the assignment operator **is not** included in this implementation. You may assume all necessary libraries are included in `X.cpp`. You need to provide the definitions (including the method signatures) in your solution.

```
// X.cpp:
```

b. Assuming your implementation from part a is correct, write the output for the following code segment:

```
void printVector(vector<string> v) {
    for (int i = 0; i < v.size(); i++)
      cout << v[i] << endl;
}

int main() {
    X a;
    X b;
    a.getVector()->push_back("A");
    a.getVector()->push_back("B");
    b = a;
    X c = a;
    b.getVector()->push_back("C");
    a.getVector()->push_back("D");
    printVector(*a.getVector());
    printVector(*b.getVector());
    printVector(*c.getVector());
}
```

Scratch Page