Object Oriented Programming
V22.0470 Fall 2001

Final Exam

Write the answers to question 1 and 2 on this sheet. Write the other answers in the exam booklet.

1. True/False. Please circle the correct answer on this sheet.

(a) T F The C++ declaration \texttt{const int \*p;} declares a pointer whose value can change, but which cannot be used to change the value contained in the memory location it points to.
(b) T F Java does not have the equivalent of C++’s private and protected derivation.
(c) T F In C++, exceptions raised by the \texttt{throw} construct must be objects belonging to a subclass of \texttt{Exception}.
(d) T F A fundamental difference between the \texttt{vector} and \texttt{list} class templates in the C++ STL is that elements of a \texttt{vector} are contiguous in memory.
(e) T F If two Java classes implement the same interface, they are considered subtypes of each other.
(f) T F An object of class \texttt{Thread} only starts a new thread executing when the \texttt{start()} method is called.
(g) T F “C++” stands for “Complexity and more complexity”.
(h) T F The C++ copy constructor is invoked when an object is passed by reference to a function.
(i) T F A Java program can be written without ever using the word \texttt{class}.
(j) T F A C++ program can be written without ever using the word \texttt{class}.

2. Multiple Choice. Circle the one desired response.

(a) Which one of the following Java code will \texttt{not} compile?

i. \texttt{class A \{} \texttt{A()} \}\ class B \texttt{extends A \{} \texttt{B(int x) \}} \}

ii. \texttt{class A \{} \texttt{A(int x) \}\ class B \texttt{extends A \{} \texttt{B(int x) \{} \texttt{super(x);} \}\}}

iii. \texttt{class A \{} \texttt{A()} \}\ class B \texttt{extends A \{} \texttt{B() \}} \}

iv. \texttt{class A \{} \texttt{A(int x) \}\ class B \texttt{extends A \{} \texttt{B(int x) \}} \}

(b) Given the declaration

\begin{verbatim}
    template<class T> class A {}
\end{verbatim}

which one of the following \texttt{will not} compile?

i. \texttt{class B: public A<int> \{};

ii. \texttt{template <class T> class B: public A \{};

iii. \texttt{template <class T> class B: public A<T> \{};

iv. \texttt{template <class T> class B: public A<int> \{};

1
(c) Given the class definition

```java
class A {
    public:
        int operator()() { return 5; }
        A operator+(int) { return *this; }
    }
```

which one of the following will not compile?

i. A f; int x = f();
ii. A x,y; x = x+y;
iii. A x; x = x+3;
iv. A f,x; f = f+x();

(d) Java's support for GUI programming (provided by the java.awt and javax.swing packages) is considered “event driven” because

i. the programmer must write loops that continually check if events, such as mouse clicks, have occurred.
ii. events, such as mouse clicks, happen asynchronously and cause parts of the GUI code to be executed.
iii. when events, such as mouse clicks, occur, Java exceptions are raised.
iv. just getting GUI code to work correctly is a notable event.

(e) Which one of the following statements is not true about multiple inheritance in C++?

i. It allows a class to be considered a subclass of two other classes.
ii. It allows objects of the child class to have all the members of two parent classes.
iii. If a class is derived from two parent classes which both have a method with the same signature, the compiler will generate an error message.
iv. A class can be derived from more than two classes.

3. Consider the definition of a class that acts like an array with bounds-checking.

```java
class my_array {
    public:
        my_array(int s): size(s) { a = new int[size]; }
        int &operator[](int i);
    private:
        int size;
        int *a;
    }
```

(a) Give the code for my_array's operator[] method, as one would define it outside the class.

(b) Overload the insertion (<<) and extraction (>>) operators so that they behave as follows:

- The insertion operator outputs all the elements of the array, separated by spaces, to the output stream.
- The extraction operator reads the elements of the array from the input stream.

Do not modify the my_array class to accomplish this.
(c) What is the best way for class my_array be modified to allow for more efficient insertion and extraction operators?

(d) Why is it necessary for the return type of operator[] to be “int &”? 

4. In Java, write the code to construct two frames (of type java.awt.Frame or javax.swing.JFrame, it doesn’t matter). In the second frame, there should be a button that, when clicked on, causes a button to appear in the first frame. FYI, in class Button there is a method:

    void addActionListener(ActionListener l)

where ActionListener is defined by

    interface ActionListener {
        void actionPerformed(ActionEvent e);
    }

5. (a) In C++, write a template function sum that can be used to sum up the elements of an array, as long as addition is defined on those elements. For example, given your template function definition, the code:

    int a[] = {1, 2, 3, 4};
    float b[] = {1.2, 3.4, 6.7};
    cout << sum(a, 4) << endl; // the parameters are the array
    cout << sum(b, 3) << endl; // and the size of the array

would print

    10
    11.3

(b) Given the following class definition,

    class student {
        public:
            student(int a) : age(a) {}
        private:
            int age
    };

what method could be added to the student class in order to be able to pass an array of students to sum, where the desired result is a student whose age is the sum of the ages of the students in the array? Write the code for this method.

(c) How can an array of students be passed to sum without adding a new method to the student class? Write the code that accomplishes this. If necessary, you can modify the student class, but you must keep age private (and cannot add a new method).

6. (a) Define the terms dynamic overloading (also known as dynamic dispatch and overriding) and static overloading.

(b) Write some Java code that illustrates both dynamic and static overloading. Be sure to include both the definitions of statically and dynamically overloaded methods and the use of them.

(c) Do the same thing as in part (b), but in C++.