These questions should take about an hour. This will be a third of the value of the midterm exam. Please try to do these questions before Tuesday's lecture and we can cover any questions during the review.

1. Write a loop to increment the odd-positioned elements of an int array **a** of size **size** by one. For instance, if **a** was declared as `int a[] = {3, 5, 6, 8}` and **size** was 4, at the end of the loop **a** would be `{3, 6, 6, 9}`. Assume **a** and **size** exist and do not need to be declared.

   One solution (not the only one):

   ```
   for (int i = 0; i < size; ++i) {
         if (i % 2 == 1) {
               array[i] += 1;
         }
   }
   ```

2. Declare and define a function called `findMin` that has parameters of an `int` array named **array** and an `int` named **size**. findMin returns an `int` equal to the minimum value in the array (e.g., the return value is less than or equal to all the other values in the array). `<climits>` has been included which gives you access to `INT_MIN` and `INT_MAX`.

   I saw that this question was somewhat ambiguous for size == 0 while writing these answers. What should it return? I chose to return the maximum int value.
   This is one solution, but not the only one:

   ```
   int findMin(int array[], int size) {
         int min = INT_MAX, i = 0;
         while (i < size) {
               if (array[i] < min) {
                     min = array[i];
               }
         }
         return min;
   }
   ```

3. Declare and define a class named Person.
    a. It has private member variables `int age` and `double height`
    b. It has public member functions
        i. `getAge` (no parameters, returns an int)
        ii. `incrementAge` (no return value or parameters)
        iii. `setHeight` (no return value, a double parameter)
        iv. `getHeight` (double return value, no parameters)
    c. It has a two argument constructor that has `int` and `double` parameters to set `age` and `height`.
    d. Do the declaration and definition in the same code block {}, similar to the Lecture 8 example code.

```
class Person {
        int age;
        double height;
   public:
        Person(int age, double height) {
                this->age = age;
                this->height = height;
        int getAge() {
                return age;
        }
        void incrementAge() {
                ++age;
        }
        void setHeight(double height) {
                this->height = height;
        }
        double getHeight() {
                return height;
        }
};
```

4. When reading input from the console, why would we use cin.get(...) instead of cin >> ?
    I would suggest using it for reading strings generally, but it is only necessary when we want to capture whitespace or differentiate the separator (e.g., ;)

5. Label the elements of this class with member functions, member variables, constructor, and destructor. Mark the access modifier of each member.

```
<line of code>                <element type>          <access modifier>
class Singleton {
      int x;                  member variable         private
      double y;               member_variable         private
      Singleton(int, double); constructor             private
   public:
      ~Singleton();           destructor              public
      int getX();             member function         public
      double getY();          member function         public

}
```

6. Circle all of the operators in this expression:

```
double x = -a + b[3] / c.d % e;
```

7. What would print to the console in this example program?

```
#include <iostream>
void printStuff(int i) {
      std::cout << "int: " << i << std::endl;
}
void printStuff(double d) {
      std::cout << "double: " << d << std::endl;
}
int main() {
      double x = 3.5;
      printStuff(x);
      printStuff(4);
      printStuff(2.0);
      return 0;
}
```

```
double: 3.5
int: 4
double 2.0
```

8. What would print to the console in this example program?

```
#include <iostream>
void increment1(int i) {
    i = i + 1;
}
void increment2(int& i) {
    i = i + 1;
}
int main() {
    int x = 2;
    std::cout << x << std::endl;
    increment1(x);
    std::cout << x << std::endl;
    increment2(x);
    std::cout << x << std::endl;
    return 0;
}
```

```
2
2
3
```

9. What would print to the console in this code snippet?

```
    int i = 0;
    while (i < 10) {
        ++i;
        if (i % 2 == 0) {
            continue;
        }
        std::cout << i << ", ";
    }
```

```
1, 3, 5, 7, 9,
// Note the trailing comma. Could we make the print better?
```

10. What access modifier would allow any part of a program to access a class member?
public

11. What is the difference between a class and an object?
A class is a template and is the program that we write. And object is an instantiation of a class

12. What is the main difference between the members of a class and a struct?
Class members are private by default, struct members are public by default

13. Which of the following is valid C++ code?

a. `int ++x = 3;`

b. `int x = 3; ++x;`

c. `3 = int x; ++x;`

d. `++x; int x = 3;`

14. What type of program is g++? Briefly describe what g++ does (1 sentence).
g++ is a compiler. g++ converts high-level, human-readable C++ code to machine code

15. Assume this array has been declared: `int a[] = {55, 43, 19, 90}`
What would you see in the console after each of these lines of code have been run?

a. `cout << a[0] << endl;`

55

b. `cout << a[3] << endl;`

90

c. `cout << a[4] << endl;`

Unknown! It could be an error (like a segmentation fault), or it could print a number if that part of memory is part of the program.