Question 1. (20 points) A little C++/STL programming. One of our friends is taking a Music Appreciation course to fulfill some of their distribution requirements. To keep track of famous composers and the kinds of works they produced, they've decided to write a small C++ program. They've created a map data structure with composer names as the keys and the kinds of works written by each composer stored in a vector of strings as the values. A sample looks like this:

```
map<string, vector<string>> composers =
  { {"bach",      {"choral", "chamber", "keyboard"}},
    {"verdi",     {"choral", "opera", "chamber"}},
    {"beethoven", {"choral", "symphony", "opera", "chamber",
                   "keyboard"}},
    {"mahler",    {"symphony", "choral", "chamber"}}
  };
```

For this problem, implement the function `find_composers`. There are two arguments to this function: the map of composer information shown above, and a string that represents a kind of work (symphony, opera, etc.). The result of the function should be a vector of strings containing names of all of the composers who wrote that kind of work. The order of the composers in the output list does not matter. For example, if `composers` is the above data, then the function should produce these results:

```
find_composers(composers, "opera") => {"beethoven", "verdi"}
find_composers(composers, "symphony") => {"mahler", "beethoven"}
```

It could also produce lists that give the composer names in a different order.

If the specified kind of work does not appear anywhere in the composers data, then the function should return an empty vector.

You should assume that all necessary headers have already been `#included` (i.e., you do not need to worry about header files), and you can also assume that a `using namespace std;` directive has been written at the top of the file.

Reminder: there is some possibly useful reference information at the end of the exam.

Hint: you might, or might not, find it useful to write an auxiliary (helper) function that reports whether a particular string does or does not appear in a vector. Entirely up to you if you think it might be useful.

Write your answer on the next page…

Question 1. (cont.) Write your C++ code below. The function header is provided for you…

```
// return a vector containing names of composers whose
// compositions include works that match kind
vector<string> find_composers(
            map<string, vector<string>> composers, string kind) {
  vector<string> result;
  for (const auto &pair: composers) {
    if (contains(kind, pair.second)) {
      result.push_back(pair.first);
    }
  }
  return result;

}

// Helper function
// return "string s appears in vector v"
bool contains(string s, const vector<string> &v) {
  for (auto & item: v) {
    if (item == s) {
      return true;
    }
  }
  return false;
}
```

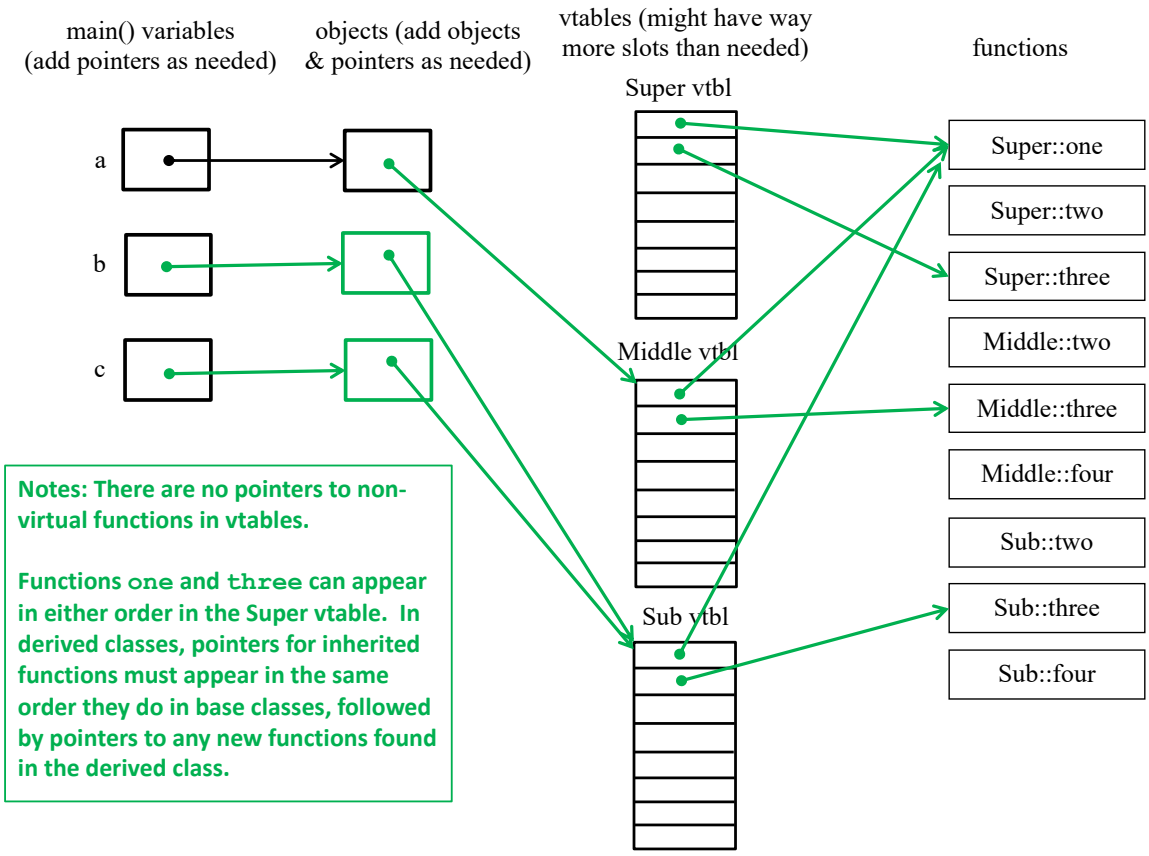
Note: STL contains a `find` algorithm that returns an iterator, which would actually be a good way to solve the problem instead of writing a separate helper function. It's fine to use that in a solution to this problem as long as it's used correctly.

Question 2. (20 points) Yet another bonkers inheritance question. As usual this program compiles and executes with no errors. Headers and `using namespace std;` omitted to save space.

```
class Super {
public:
  virtual void one()   {        cout << "Super::one" << endl; }
          void two()   {        cout << "Super::two" << endl; }
  virtual void three() { two(); cout << "Super::three" << endl; }
};

class Middle: public Super {
public:
  virtual void three() {          cout << "Middle::three" << endl; }
          void four()  { two();   cout << "Middle::four" << endl; }
          void two()   { three(); cout << "Middle::two" << endl; }
};

class Sub: public Middle {
public:
          void two()   {        cout << "Sub::two" << endl; }
          void four()  { one(); cout << "Sub::four" << endl; }
          void three() {        cout << "Sub::three" << endl; }
};

int main() {
  Super *a = new Middle();
  a->one();
  a->two();
  a->three();
  cout << "---(1)---" << endl;
  Super *b = new Sub();
  b->one();
  b->two();
  b->three();
  cout << "---(2)---" << endl;
  Middle *c = new Sub();
  c->one();
  c->two();
  c->three();
  c->four();
  return 0;
}
```

Continue with the problem on the next pages. **Do not remove this page from the exam.**

Question 2. (cont.) (a) (6 points) Complete the diagram below to show all of the variables, objects, virtual method tables (vtables) and functions in this program. Parts of the diagram are supplied for you.



(b) (14 points) What does this program print when it executes? (write your answer in multiple columns if needed)

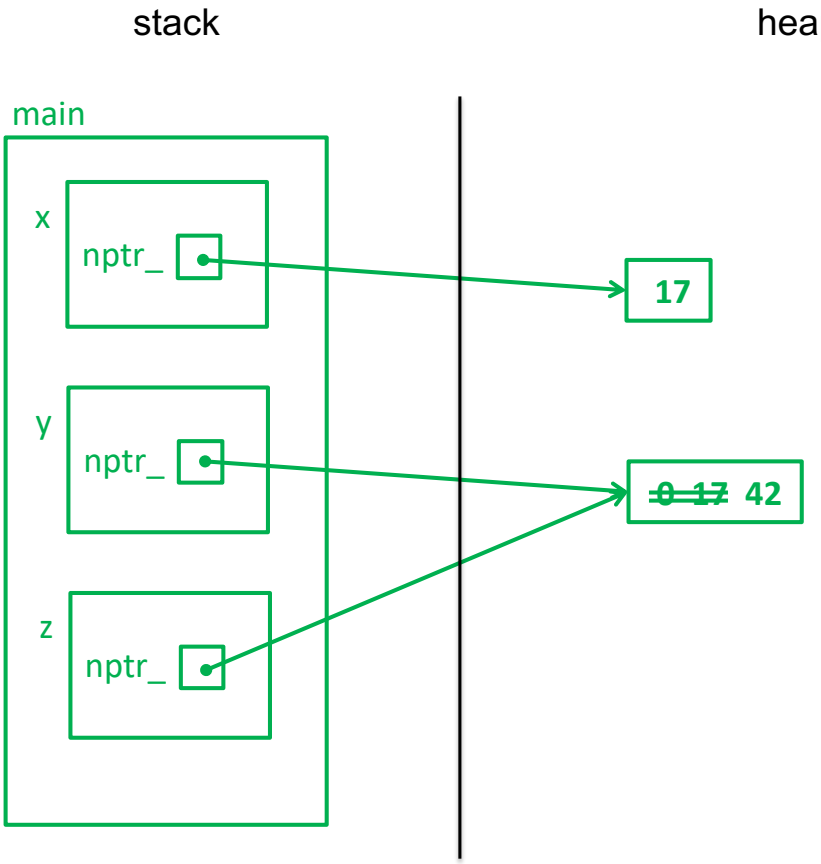
```
Super::one
Super::two
Middle::three
---(1)---
Super::one
Super::two
Sub::three
---(2)---
Super::one
Sub::three
Middle::two
Sub::three
Sub::three
Middle::two
Middle::four
```

Question 3. (20 points) A trip down memory lane. A colleague who is new to C++ is learning how classes work by building a small class that has an integer instance variable (somewhat like the `Integer` wrapper class in Java). Here is the code:

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

// wrapped integer on the heap
class Int {
public:
  Int() {
    nptr_ = new int(0);
  }

  Int(int val) {
    nptr_ = new int(val);
  }

  virtual ~Int() {
    delete nptr_;
  }

  Int& operator=(const Int& rhs) {
    if (this != &rhs) {
      *nptr_ = *rhs.nptr_;
    }
    return *this;
  }

  int get() { return *nptr_; }

  void set(int newval) { *nptr_ = newval; }

private:
  int *nptr_;
};

int main() {
  Int x(17);
  Int y;
  Int z(y);
  y = x;
  z.set(42);
  //  Draw memory diagram when execution reaches here //
  cout << x.get() << " " << y.get() << " " << z.get() << endl;
  return 0;
}
```

(continued on next page)

Question 3. (cont.) (a) (10 points) This program does compile and executes successfully, at least for a while. Draw a diagram showing the contents of memory when execution reaches the line in the `main` function with the comment that says "draw memory when execution reaches **here**". Your diagram should clearly separate data on the stack from data allocated on the heap. Be sure to indicate clearly which data on the stack is part of the stack frame for function `main` by drawing a box labeled "main" with appropriate variables and data inside.



(b) (2 points) What output is produced when the next line is executed to print the values of integers x, y, and z?

```
17 42 42
```

(continued on next page)

Question 3. (cont.) (c) (3 points) There is some sort of error in this program and it does not finish execution successfully. What, precisely, is the error and what goes wrong?

When we create the variable `z` in the statement `Int z(y)`, a synthesized default copy constructor is used to initialize `z`. This default copy constructor copies the pointer from `y`, which means that both `y` and `z` reference the same integer object on the heap.

When the program terminates, the destructors for both `y` and `z` try to delete this same heap object, resulting in a double-delete error.

(d) (3 points) How should we fix the error that you described above in part (c) that causes the program to crash? You should describe what to change in the code and give any specific replacement C++ code needed to fix the problem.

Add a copy constructor to class Int that creates a new heap object containing the same data as the original object when a copy is initialized so that each `Int` object has its own separate heap object. The following code will work:

```
Int (const Int & other) {
  nptr_ = new int(*other.nptr_);
}
```

(Of course, we can use an initializer list instead of putting the code in the body of the constructor. Either will solve the problem.)

Note: Several answers used `other.get()` instead of accessing the value directly inside the copy constructor. Technically this won't work if the parameter is a `const` parameter since `get` wasn't labeled as a `const` function. We overlooked that during grading. (`get` should have been labeled const in the original code, but we missed that due to an oversight.)

(e) (2 points) After fixing the error, what output does the program print when it is executed?

```
17 17 42
```

Question 4. (12 points) Templates. The code from the previous question is a wrapper for a single `int`(eger) value. There might be other cases where a simple wrapper like this would be useful for objects of other types, not just `int`.

Write your changes in the code below to modify the class **and** the main program that uses it so the class has a generic type parameter `T` for the type of the wrapped variable, and then modify the main program to use that template to create instances of the class with the actual type parameter `int`. Also, you should rename the class to be `Wrapper` instead of `Int`. The resulting program should work exactly the same as the original program, and will contain exactly the same bugs – don't fix those – but it should have a generic type for the wrapped value.

```
template <typename T> class ~~Int~~ Wrapper {
public:
  ~~Int~~ Wrapper() {
    nptr_ = new ~~int~~ T(0);  // OK to assume T(0) works for type T
  }

  ~~Int~~ Wrapper(~~int~~ T val) {
    nptr_ = new ~~int~~ T(val);
  }

  virtual ~ ~~Int~~ Wrapper() {
    delete nptr_;
  }

  ~~Int~~ Wrapper & operator=(const ~~Int~~ Wrapper& rhs) {
    if (this != &rhs) {
      *nptr_ = *rhs.nptr_;
    }
    return *this;
  }

  ~~int~~ T get() { return *nptr_; }

  void set(~~int~~ T newval) { *nptr_ = newval; }

private:
  ~~int~~ T *nptr_;
};

int main() {
  ~~Int~~ Wrapper<int>x(17);   // <int> is optional in these
  ~~Int~~ Wrapper<int>y;       // declarations.  The compiler can
  ~~Int~~ Wrapper<int>z(y);    // figure out the type from the 17
  y = x;                       // initial value for x.
  z.set(42);
  cout << x.get() << " " << y.get() << " " << z.get() << endl;
  return 0;
}
```

Question 5. (16 points) Yet another broken web server. Here is another attempt at creating a small web server, but, alas, it is buggy, like many of the ones we've been debugging over the last couple of weeks. But, like the ones we've been working on, with some fixes, this one can be repaired.

Write in corrections below to show how the code should be rearranged, changed, added to, or otherwise fixed to create a properly working server that accepts a connection from a client and then reads data from the client and writes it back until the socket is closed. Feel free to draw arrows showing how to move code around if needed, but be sure it is clear to the reader what you mean.

You should assume that all functions always succeed – ignore error handling for this question. Further, assume that the `hints` data structure is set up correctly for the call to `getaddrinfo` and the first address returned by `getaddrinfo` works and we don't need to search that linked list to find one that does work. Also, ignore the details of parameter lists – assume that all the "…" or missing parameters are valid and appropriate.

Also assume that `WrappedRead` and `WrappedWrite` are like the `read()` and `write()` POSIX functions, except that they automatically handle error conditions, including looping to retry the read or write if `EINTR` or `EAGAIN` errors occur.

Reminder: there is some potentially useful reference information at the end of the exam.

(Corrections shown in bold green)

```
int main(int argc, char** argv) {
 struct addrinfo hints, *rp;
 memset(&hints, 0, sizeof(hints));
 hints.ai_... = ...;  // specify values for desired options
 getaddrinfo(NULL, argv[1], &hints, &rp);  // assume *rp is ok
 int fd = socket(rp->ai_family, rp->ai_socktype,
                                          rp->ai_protocol);
 setsockopt(fd, SOL_SOCKET, SO_REUSEADDR, ...);
 bind(fd, rp->ai_addr, rp->ai_addrlen); // add after setsocketopt
 freeaddrinfo(rp);
 listen(fd, SOMAXCONN);                  // add
 int cfd = accept(fd, pointers to client address info & length);
                      // add client fd and fix arguments (ok
                      // to leave out address pointer details.)
```

(Continued on next page)

Question 5. (cont.) Continue writing in your corrections below as needed.

```
 char buf[BUFSIZE]  // BUFSIZE is a predefined constant
 // read data from the client and write it back until the
 // socket is closed
 while (1) {
   int r_res = WrappedRead(cfd, buf, BUFSIZE);  // use client fd
   if (r_res == 0) {
     break;
   } else if (r_res < 0) {
     close(cfd); close(fd);       // close both file descriptors
     return EXIT_FAILURE;
   }
   int w_res = WrappedWrite(cfd, buf, r_res);   // use client fd
   if (w_res != r_res) {
     close(cfd); close(fd);       // close both file descriptors
     return EXIT_FAILURE;
   }
 }
 close(fd);     // close listening file descriptor
 close(cfd);    // close client file descriptor
 return EXIT_SUCCESS;
}
```

Question 6. (20 points) Too many things at once. Consider the following small program that uses pthreads. (This does compile and execute successfully.)

```
int x = 0;
int y = 0;

void *worker(void *ignore) {
  for (int k = 1; k <= 3; k++) {
    x = x + k;
    y = y + 1;
  }
  printf("x = %d, y = %d\n", x, y);
  return NULL;
}

int main() {
  pthread_t t1, t2, t3;
  int ignore;
  ignore = pthread_create(&t1, NULL, &worker, NULL);
  ignore = pthread_create(&t2, NULL, &worker, NULL);
  ignore = pthread_create(&t3, NULL, &worker, NULL);
  pthread_join(t1, NULL);
  pthread_join(t2, NULL);
  pthread_join(t3, NULL);
  printf("final x = %d\n", x);
  printf("final y = %d\n", y);
  return EXIT_SUCCESS;
}
```

When we run this program, it starts three threads that each assign values to `x` and `y` and then print the values of those variables. Then the program waits for all threads to finish, and prints the final values of the variables `x` and `y`.

(a) (4 points) What output would this program print if the threads are executed sequentially, not concurrently? In other words, what output is produced if thread 1 executes first and then thread 2 executes after thread 1 terminates and then thread 3 executes last?

```
x = 6, y = 3
x = 12, y = 6
x = 18, y = 9
final x = 18
final y = 9
```

(continued on next page)

Question 6. (cont.) (b) (6 points) When the threads run concurrently, is it possible to get different values for the numbers in the output if the program is executed multiple times? If it is, give two possible outputs that could be produced by the program that are different from the sequential output in part (a). If there is only one other possible set of output numbers besides the sequential one from part (a), write that one and indicate that it is the only possible different output.

(You should assume that the statements in each individual thread are executed in the order written, and not rearranged by the compiler or memory system to be executed out-of-order. You should also assume that the `printf` calls don't interfere with each other and that each line of output is printed correctly on a single line and separately from other output lines, and the variables being printed are read simultaneously with no other thread interrupting to change them while `printf` is running. If different executions lead to different output numbers, it is only because of the interaction between statements in the threads as they run concurrently.)

Here are a few possibilities. There are undoubtedly many others.

```
x = 6, y = 3
x = 7, y = 6
x = 12, y = 6
final x = 12
final y = 6
```

```
x = 7, y = 3
x = 13, y = 4
x = 18, y = 6
final x = 18
final y = 6
```

The question did ask if it was possible to get "different values for the numbers", so answers that just showed the output from the sequential threads in a different order did not receive full credit.

(c) (4 points) Assuming that the threads are executed concurrently, as in part (b), what are the possible final values of variables x and y? Circle all that could possibly happen on some possible execution (you can draw a big circle around several values rather than individual small circles for each value if it helps save time ☺):

Possible final values for x:

0	1	2	3	4	5	6	7	8	9	10
	11	12	13	14	15	16	17	18	19	20 or more

Possible final values for y:

0	1	2	3	4	5	6	7	8	9	10
	11	12	13	14	15	16	17	18	19	20 or more

We probably should have worded this question to ask for the lowest and highest values instead of which values are possible, since it may be the case that some of the intermediate values are not possible. When we graded the question, we only looked to see if the min and max values were correct and did not analyze all of the intermediate values.

(continued next page)

Question 6 (cont.) (d) (6 points) Now we would like to add synchronization to the initial code so that the program still has three concurrent threads, but properly synchronized this time. In the copy of the original code below, insert appropriate locking using a single `pthread_mutex_t` lock so the threads do not interfere with each other as they update shared variables `x` and `y` but also so there is as much concurrency as possible. i.e., each thread should acquire the lock just long enough to avoid synchronization problems, but should not, for example, acquire the lock and hold on to it until it has completely finished, which would effectively cause the threads to execute one at a time rather than concurrently.

Hints: `pthread_mutex_t` and related mutex init, lock, and unlock functions are useful. (See the section at the end of the exam for additional reference information.)

(Changes shown in bold green text)

```
pthread_mutex_t lock;
int x = 0;
int y = 0;
void *worker(void *ignore) {
  for (int k = 1; k <= 3; k++) {
    pthread_mutex_lock(&lock);
    x = x + k;
    y = y + 1;
    pthread_mutex_unlock(&lock);
  }
  printf("x = %d, y = %d\n", x, y);
  return NULL;
}

int main() {
  pthread_t t1, t2, t3;
  int ignore;
  pthread_mutex_init(&lock, NULL);
  ignore = pthread_create(&t1, NULL, &worker, NULL);
  ignore = pthread_create(&t2, NULL, &worker, NULL);
  ignore = pthread_create(&t3, NULL, &worker, NULL);
  pthread_join(t1, NULL);
  pthread_join(t2, NULL);
  pthread_join(t3, NULL);
  printf("final x = %d\n", x);
  printf("final y = %d\n", y);
  pthread_mutex_destroy(&lock);
  return EXIT_SUCCESS;
```

Question 7. (2 free points – all answers get the free points) Draw a picture of something you're planning to do over spring break!

Congratulations on lots of great work this quarter!!
Have a great spring break and best wishes for the future!
The CSE 333 staff