EECE.3220: Data Structures

Spring 2020

Exam 1

February 20, 2020

Name: ____

For this exam, you may use two 8.5" x 11" double-sided pages of notes. All electronic devices (e.g., calculators, cell phones) are prohibited. If you have a cell phone, please turn off your ringer before the start of the exam to avoid distracting other students.

The exam contains 5 sections for a total of 100 points, as well as a 10 point extra credit question. Please answer all questions in the spaces provided. If you need additional space, use the back of the page on which the question is written and clearly indicate that you have done so.

Please read each question carefully before you answer. In particular, note that:

- In Section 4, you must complete several short functions. In some cases, we have provided comments to describe what your function should do and written some of the code.
  - You can complete each function using only the variables that have been declared, but you may declare and use other variables if you want.
- Each question in Section 5 is a multiple choice question. Questions 5b has at least one correct answer, but may have more than one correct answer. Questions 5a and 5c each have exactly one correct answer.

You will have 2 hours to complete this exam.

S1: C++ input/output	/ 24
S2: Functions	/ 12
S3: C++ strings	/ 12
S4: Classes	/ 32
S5: Multiple choice: dynamic allocation, vectors	/ 20
TOTAL SCORE	/ 100
S6: EXTRA CREDIT	/ 10

1. (24 points) ***C++ input/output***
For each short program shown below, list the output exactly as it will appear on the screen. Be sure to clearly indicate spaces between characters when necessary.

You may use the available space to show your work as well as the output; just be sure to clearly mark where you show the output so that I can easily recognize your final answer.

a. (12 points)

```
int main() {
   int ival = 2020;
   double dval1 = 2.19;
   double dval2 = 3.9;

   cout << ival << ", " << dval1;

   cout << showpoint << dval2 << endl;

   cout << setprecision(2) << "dval1 = " << dval1 << '\n';

   cout << fixed << "dval2 = " << dval2 << endl;

   cout << setprecision(5) << "ival = " << ival << endl;

   return 0;
}
```

1 (continued)
b. (12 points)

For this program, assume the user inputs the three lines below. The number `'2'` in `2.20.2020` is the first character the user types. There are no spaces on the first line, one space (`' '`) between `12-2` and `PM` on the second line, and two spaces on the third line—one between `EECE.3220` and `Exam`, and one between `Exam` and `1`. Assume each line ends with a newline character (`'\n'`).

You must determine how the program handles this input and then print the appropriate results. Note that the program may not read all characters on the input lines, but no input statement fails to read data—an appropriate value is assigned to every variable.

```
2.20.2020
12-2 PM
EECE.3220 Exam 1
```

```
int main() {
   int i1, i2;
   double d1, d2;
   char c1, c2, c3;
   string str1, str2;

   cin >> d1 >> c1 >> i1;
   cin >> d2 >> i2 >> c2 >> str1;
   cin.ignore(9);
   cin.get(c3);
   getline(cin, str2);

   cout << i1 << ' ' << i2 << endl;
   cout << d1 << ' ' << d2 << endl;
   cout << c1 << c2 << c3 << endl;
   cout << str1 << endl;
   cout << str2 << endl;

   return 0;
}
```

2. (12 points) ***Functions***

For the short program shown below, list the output exactly as it will appear on the screen. Be sure to clearly indicate spaces between characters when necessary.

You may use the available space to show your work as well as the output; just be sure to clearly mark where you show the output so that I can easily recognize your final answer.

```
void f1(int &v1, int *v2) {
   v1 = v1 + 10;
   *v2 = *v2 - 10;
}

void f2(int &v3, int &v4) {
   int temp = v4;
   v4 = v3 * 3;
   v3 = temp / 4;
}

int f3(int* v5, int v6) {
   v6++;
   *v5 -= 4;
   return v6 - *v5;
}

int main() {
   int x(32), y(20), z(45);

   f1(z, &y);
   cout << x << " " << y << " " << z << endl;
   f2(y, x);
   cout << x << " " << y << " " << z << endl;
   z = f3(&x, z);
   cout << x << " " << y << " " << z << endl;

   return 0;
}
```

3. (12 points) ***Strings***

For the short program shown below, list the output exactly as it will appear on the screen. Be sure to clearly indicate spaces between characters when necessary.

You may use the available space to show your work as well as the output; just be sure to clearly mark where you show the output so that I can easily recognize your final answer.

```
int main() {
   string s1("data"), s2("structures"), s3("spring");
   unsigned i, nr, nt;

   cout << "s1 = " << s1 << ", s2 = "
         << s2 << ", s3 = " << s3 << endl;

   if (s1.length() > s3.length())
      cout << s2[0] + s3.substr(3) << '\n';
   else
      cout << "de" + s2.substr(0, 6) + s3.substr(3) << '\n';

   i = 0;
   nr = nt = 0;
   while (s2.find('r', i) != string::npos ||
          s2.find('t', i) != string::npos) {
      if (s2.find('r', i) != string::npos)
          nr++;
      if (s2.find('t', i) != string::npos)
          nt++;
      i = i + 2;
   }
   cout << i << ", " << nr << ", " << nt << endl;
   return 0;
}
```

4. (20 points) ***Classes***

Complete each member function described below for the `PointList` class. The `Point` and `PointList` classes and some of their functions are defined on the extra sheet with the exam.

a. (10 points) `PointList::printList(ostream &out)`: Print every `Point` in the calling object to the output stream `out`, with each point on a single line. If the `PointList` is empty, print `"List empty\n"`.

```
void PointList::printList(ostream& out) {

   // Print appropriate message if list is empty

   if (_______________________________________________________)
      out << "List empty\n";

   // Otherwise, print every Point in list, 1 per line
   else {









   }
}
```

b. (6 points) `PointList::isFull()`: Returns true if the list is full, false otherwise.

```
bool PointList::isFull() {









}
```

4 (continued)

Complete each member function described below for the `PointList` class. <u>The `Point` and `PointList` classes and some of their functions are defined on the extra sheet with the exam.</u>

c. (16 points) `PointList::isLine()`: Examine all points in the calling object, returning true if those points represent a straight line and false otherwise.

   A list of points represents a straight line if all consecutive pairs of points have the same slope between them. To calculate the slope between two points, divide the difference in Y coordinates by the difference in X coordinates. For example, given points (2, -1) and (4, 5), the slope is:

   (5 – (-1)) / (4 – 2) = 6 / 2 = 3

```
bool PointList::isLine () {
  unsigned i;       // Loop index
  double slope;     // Slope between first two points, used for
                    //  comparison

  // Simple case—a list with 0 or 1 points can't be a line

  if (_________________________________________________________)
    return false;

  // Otherwise, find slope between first two points

  slope =


  // Check slope between all remaining pairs of points
  // If any pair has different slope than first pair, not a line

  for (______________________________________________________) {

    if (_______________________________________________________

        _______________________________________________________)
          return false;
  }

  // If you reach end of loop, must be a line
  return true;

}
```

5. (20 points, 5 points each) ***Multiple choice: dynamic allocation, vectors***

For each of the multiple choice questions below, clearly indicate your response(s) by circling or underlining all choices you think best answer the question.

a. Which of the following choices contains statements that correctly allocate an array of doubles of size X, then later deallocate the space for that array? (The … characters represent unspecified code that comes between these statements. **This question has exactly one correct answer.**

i.
```
double *arr = new double;
…
delete arr;
```

ii.
```
double *arr = new double(X);
…
delete () arr;
```

iii.
```
double *arr = new double[X];
…
delete arr;
```

iv.
```
double *arr = new double[X];
…
delete [] arr;
```

b. Which of the following choices creates a vector of integers in which N represents the vector size and M represents the value stored in each element of the vector **This question has at least one correct answer, but may have more than one correct answer! Circle ALL choices that correctly answer the question.**

i.
```
vector <int> v1(N) = M;
```

ii.
```
vector <int> v2(N, M);
```

iii.
```
vector <int> v3 = {N, M};
```

iv.
```
vector <int> v4;
for (unsigned i = 0; i < N; i++)
  v4.push_back(M);
```

5 (continued)

c. What does the short code sequence below print? **This question has exactly one correct answer.**

```
vector <int> v(2, 5);
v.push_back(2);
v.push_back(20);
v.resize(8);
for (i = 0; i < 4; i++)
   v.at(7 - i) = i;
for (int val : v)
   cout << val << ' ';
```

i. `2 5 2 20 8`

ii. `5 5 2 20 8`

iii. `5 5 2 20`

iv. `5 5 2 20 3 2 1 0`

v. None of the above

d. Which of the following statements accurately reflect your opinion(s)? Circle all that apply (but please don't waste too much time on this "question")!

i. "This course is moving too quickly."

ii. "This course is moving too slowly."

iii. "I've attended very few lectures, so I don't really know what the pace of the course is."

iv. "I hope the next exam is as easy as this question."

6. (10 points) ***EXTRA CREDIT***

Complete the parameterized PointList constructor as described below, which uses a vector of Point objects to initialize a PointList object. The `Point` and `PointList` classes and some of their functions are defined on the extra sheet with the exam. Your constructor should:

- Fill the `PointList` as follows:
  - If the vector holds more `Point`s than the `PointList` can hold, copy as many `Point`s as you can into the `PointList`, starting with the first value in the vector.
  - Otherwise, copy all `Point` objects from the vector to the `PointList`
- Set all `PointList` data members appropriately.

```
PointList::PointList(vector <Point> pts) {








}
```