EECE.3220: Data Structures

Fall 2019

Exam 1

October 3, 2019

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 Question 3b, you must complete a short function. We have provided comments to describe what your function should do and written some of the code.
  - You can complete the 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 5a, 5b, and 5e each have at least one correct answer, but may have more than one correct answer. Questions 5c and 5d 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	/ 20
S4: Algorithmic complexity	/ 20
S5: Multiple choice: classes, dynamic allocation, vectors	/ 24
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 iv1 = 6;
   int iv2 = 36;
   double dval = 103.2019;

   cout << iv1 << endl << iv2;
   cout << setprecision(4) << dval << '\n';
   cout << fixed << "Now, iv1 = "
          << iv1 << ", iv2 = " << iv2
          << ",\ndval = " << setprecision(3)
          << dval << endl;
   return 0;
}
```

1 (continued)
b. (12 points)

For this program, assume the user inputs the three lines below. The letter `'E'` in `Exam` is the first character the user types. There is one space (`' '`) between `Exam` and `1` on the first line, one space between `2-4` and `PM` on the second line, and no spaces on the third line. 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.

```
    Exam 1
    2-4 PM
    10.3.2019
```

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

   cin.get(c1);
   cin.ignore(1);
   cin >> str1 >> i1 >> d1 >> i2 >> c2;
   getline(cin, str2);     // Slightly different getline syntax to
                           //  read line into string object, but
                           //  it works like in-class examples
                           // If str2 was a character array, we'd
                           //  call cin.getline(str2);
   cin >> d2;

   cout << i1 << ", " << i2 << '\n';
   cout << d1 << ", " << d2 << '\n';
   cout << c1 << ", " << c2 << '\n';
   cout << str1 << ", " << str2 << '\n';

   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.

```
int f1(int &arg1, int arg2) {
   arg1 = arg1 * 2;
   arg2 = arg2 / 2;
   return arg1 + arg2;
}

int f2(int *arg3, int arg4) {
   *arg3 = arg4 + 5;
   arg4 = arg4 - 5;
   return *arg3 + arg4;
}

int f3(int &arg5, int &arg6) {
   arg5 = arg6;
   arg6 = arg5 / 4;
   return arg5 + arg6;
}

int main() {
   int v1, v2(5), v3(8);

   v1 = f1(v2, v3);
   cout << v1 << ' ' << v2 << ' ' << v3 << '\n';
   v2 = f2(&v3, v1);
   cout << v1 << ' ' << v2 << ' ' << v3 << '\n';
   v3 = f3(v1, v2);
   cout << v1 << ' ' << v2 << ' ' << v3 << '\n';

   return 0;
}
```

3. (20 points) ***Strings***

a. (10 points) 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, s2("exam"), s3("October");

   if (s1.empty() && s2.empty()) {
      s1 = s3;
      s2 = "test";
   }
   else if (s1.length() < 5) {
      s1 = "test";
   }
   else
      s3 = "test";

   cout << s1 << ' ' << s2 << ' ' << s3 << '\n';

   s1[2] = s2.at(1);
   s2 += "ple";
   s3 = s2.substr(2, 2) + s3.substr(4);

   cout << s1 << ' ' << s2 << ' ' << s3 << '\n';

   return 0;
}
```

3 (continued)

b. (10 points) Complete the function described below:

```
void removeDuplicates(string &str);
```

Removes all instances of consecutive, repeated characters from the argument `str`, leaving just one of each character in the string. For example:

- If `string s1 = "aa bb cc dd"`, after calling `removeDuplicates(s1)`, `s1 = "a b c d"`
- If `string s2 = "no repeats"`, after calling `removeDuplicates(s2)`, `s2 = "no repeats"`
- If `string s3 = "1223334444"`, after calling `removeDuplicates(s3)`, `s3 = "1234"`

The comments in the function provide hints about one potential implementation of this function; if you have a different solution, feel free to ignore the comments.

```
void removeDuplicates(string &str) {
   unsigned i;            // Loop index
   string tmp;            // Temporary string

   // Identify repeated characters in str and only copy one of
   //    each to tmp











   // Copy temporary string to str before end of function


}
```

4. (20 points) ***Algorithmic complexity***

For each function in this problem, determine (a) an equation for the worst-case computing time T(n) (expressed as a function of n*, i.e.* 2n + 4) and (b) the order of magnitude (expressed using big O notation, *i.e.* O(n)). Note that:

- Each executable line of code is numbered so you can refer to it by number if necessary.
- A `for` loop may be treated as a single statement, not three separate statements.

a. (10 points)

```
int functionA(int n){
    int i, j;
1   int total = 1;
2   int count = 0;
3   for (i = 0; i <= n; i++) {
4       count++;
5       j = i;
6       while (j > 0) {
7           total = total * 2;
8           j--;
        }
    }
9   return total;
}
```

4 (continued)

For each function in this problem, determine (a) an equation for the worst-case computing time T(n) (expressed as a function of n, *i.e.* 2n + 4) and (b) the order of magnitude (expressed using big O notation, *i.e.* O(n)). Note that:

- Each executable line of code is numbered so you can refer to it by number if necessary.
- A `for` loop may be treated as a single statement, not three separate statements.

b. (10 points)

```
void functionB(int arr[], int n){
    int i, j;
1   if (n >= 20) {
2       for (i = 0; i < 20; i++)
3           arr[i]++;
    }
4   if (n >= 30) {
5       for (i = 0; i < 30; i++) {
6           for (j = i; j < 30; j++) {
7               arr[i] += arr[j];
            }
        }
    }
8   if (n >= 40) {
9       for (i = 39; i >= 0; i--)
10          arr[i] = arr[i] * 2;
    }
}
```

5. (24 points, 4 points each) ***Multiple choice: classes, 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.

Parts a, b, and c refer to the following class definition:

```
class MyClass {
public:
   MyClass();
   MyClass(int i, string s);
   void setMem(int arg1, int arg2, string arg3);
   void display(ostream &out);
private:
   int mem1, mem2;
   string mem3;
};
```

a. Which of the following statements is a valid declaration for an object of type `MyClass`? **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. `MyClass mc1;`

ii. `MyClass mc2(1, "two");`

iii. `MyClass mc3(3, 4, "five");`

iv. `MyClass mc4.mem1 = 3;`

v. `MyClass(0, "zero");`

5 (continued)

Part b uses the same class definition as part a:

```
class MyClass {
public:
   MyClass();
   MyClass(int i, string s);
   void setMem(int arg1, int arg2, string arg3);
   void display(ostream &out);
private:
   int mem1, mem2;
   string mem3;
};
```

b. Which of the following choices represents a valid implementation of a `MyClass` constructor?

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.

```
MyClass() {
   mem1 = 0;
   mem2 = 1;
   mem3 = "blank";
}
```

ii.

```
MyClass::MyClass() : mem1(10), mem2(3), mem3("Exam 1")
{}
```

iii.

```
MyClass::MyClass(int i, string s) : mem1(i), mem2(i),
   mem3(s)
{}
```

iv.

```
MyClass::setMem(int arg1, int arg2, string arg3) {
   mem1 = arg1;
   mem2 = arg2;
   mem3 = arg3;
}
```

5 (continued)

Part c uses two class definitions:

```
class MyClass {
public:
   MyClass();
   MyClass(int i, string s);
   void setMem(int arg1, int arg2, string arg3);
   void display(ostream &out);
private:
   int mem1, mem2;
   string mem3;
};

class NewClass {
public:
   NewClass();
   NewClass(int i1, int i2, string s1, string s2);
   void someFunction(MyClass &randomReference);
private:
   MyClass x, y;
}
```

c. Which of the following represents a valid implementation of the `NewClass` parameterized constructor? **This question has exactly one correct answer.**

i.
```
NewClass::NewClass(int i1, int i2, string s1, string s2) {
   x = (i1, s1);
   y = (i2, s2);
}
```

ii.
```
NewClass::NewClass(int i1, int i2, string s1, string s2) {
   x.mem1 = x.mem2 = i1;
   x.mem3 = s1;
   y.mem1 = y.mem2 = i2;
   y.mem3 = s2;
}
```

iii.
```
NewClass::NewClass(int i1, int i2, string s1, string s2) :
  x(i1, s1), y(i2, s2) {}
```

iv.
```
NewClass::NewClass(int i1, int i2, string s1, string s2) {
   someFunction(x);
   someFunction(y);
}
```

5 (continued)

d. Which of the following choices dynamically allocates an array of doubles in which `N` represents the array size and `M` represents the initial value assigned to each element in the array? **<u>This question has exactly one correct answer.</u>**

i.
```
double *arr = new double(N, M);
```

ii.
```
double *arr = new double[N, M];
```

iii.
```
double arr[N] = { M };
```

iv.
```
double *arr = new double[N];
for (unsigned i = 0; i < N; i++)
   arr[i] = M;
```

e. Which of the following choices creates a vector of integers containing 5 elements and assigns the value 3220 to every element? **<u>This question has at least one correct answer, but may have more than one correct answer! Circle ALL choices that correctly answer the question.</u>**

i.
```
vector <int> v1(5) = 3220;
```

ii.
```
vector <int> v2(5, 3220);
```

iii.
```
vector <int> v3 = {3220, 3220, 3220, 3220, 3220};
```

iv.
```
vector <int> v4;
for (unsigned i = 0; i < 5; i++)
   v4.push_back(3220);
```

f. 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) ***<u>EXTRA CREDIT</u>***

Complete the function below, which returns a string formed by combining all substrings within `s1` that start with the character `c` and have length `len`. When building the new string, add a space after each added substring. For example:

- `buildStr("testing string", 't', 3)` returns `"tes tin tri"`
- `buildStr("accident", 'c', 2)` returns `"cc ci"`
- `buildStr("EECE.3220", 'x', 5)` returns `""` (empty string)

```
string buildStr(string s1, char c, int len) {
   unsigned i;            // Character position within string
   string result;         // Final result

   // GO THROUGH s1 AND FIND SUBSTRINGS TO CONCATENATE









   return result;
}
```