EECE.3220: Data Structures

Spring 2019

Exam 1 Solution

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 = 3220;
    double dval = 2.25;

    cout << ival << dval;      Prints values of ival & dval with
                                 no spaces in between

    showpoint forces decimal point to be shown for dval, with
      default precision specifying # of significant figures (6)
    cout << showpoint << ", ival = " << ival
         << ", dval = " << dval;

    fixed format means precision now determines # of digits after
      decimal point
    cout << fixed;
    cout << "\nival = " << ival << ", dval = " << dval;

    setprecision changes precision to 1, so 2.25 rounded to 2.3
    cout << setprecision(1) << endl;
    cout << "ival = " << ival << ", dval = " << dval << endl;
    return 0;
}
```

OUTPUT

```
32202.25, ival = 3220, dval = 2.25000
ival = 3220, dval = 2.250000
ival = 3220, dval = 2.3
```

1 (continued)
b. (12 points)

For this program, assume the user inputs the two lines below. The letter `'E'` in `EECE` is the first character the user types. There is one space (`' '`) between `EECE.3220` and `E1`, and no spaces on the second line of input. 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.

```
     EECE.3220 E1
     2/25/2019

int main() {
    int i1, i2;
    double d1, d2;
    char c1, c2, c3;
    char buf[10];

    cin.ignore(4);              Skips first 4 chars (EECE)
    cin >> d1 >> c1 >> i1;      d1 = .3220, c1 = 'E', i1 = 1
    cin.get(c2);                c2 = 1st char after 1 = '\n'
    cin >> i2 >> c3 >> d2;      i2 = 2, c3 = '/', d2 = 25
    cin.getline(buf, 10);       buf = rest of line = "/2019"

    cout << i1 << ' ' << i2 << endl;
    cout << d1 << ' ' << d2 << endl;
    cout << c1 << c2 << c3 << endl;      Since c2 = '\n',
                                         c1 & c3 print on
                                         different lines
    cout << buf << endl;
    return 0;
}
```

OUTPUT:

```
1 2
0.322 25
E
/
/2019
```

2. (14 points) ***Structures***

Complete the function below (the `Box` structure is defined on the extra sheet):

```
int maxVol(Box list[], int n);
```

This function takes as arguments an array of `Box` structures, `list`, as well as the number of structures in the list, `n`. The function returns the index of the structure representing the box with the greatest volume. For example, given:

```
Box arr[4] = { {1, 3, 5}, {9, 9, 9}, {4, 3, 2}, {2, 2, 2} };
```

the function call `maxVol(arr, 4)` would return 1, as `arr[1]` has the greatest volume (729, which is greater than the volumes of `arr[0]` (volume 15), `arr[2]` (24), and `arr[3]` (8)).

```
int maxVol(Box list[], int n) {
   int i;            // Loop index
   int maxI;         // Index of largest prism
   double maxVol;    // Volume of largest prism

   // Initialize variables as needed
   maxI = 0;
   maxVol = list[0].W * list[0].L * list[0].H;

   // Go through list; update max variables if larger box found
   for (i = 1; i < n; i++) {
      if (maxVol < list[i].W * list[i].L * list[i].H) {
         maxVol = list[i].W * list[i].L * list[i].H;
         maxI = i;
      }
   }

   // Return index of box with greatest volume
   return maxI;
}
```

3. (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) {
   int temp = v1;
   v1 = v2;
   v2 = temp;
}
```

Since v1 is passed by value and v2 is passed by reference, v2 will hold value of v1, while v1 is unchanged

```
void f2(int *v3, int v4) {
   int temp = *v3;
   *v3 = v4;
   v4 = temp;
}
```

Since v3 is passed by address and v4 is passed by value, whatever v3 points to will hold value of v4, while v4 is unchanged

```
void f3(int *v5, int &v6) {
   int temp = *v5;
   *v5 = v6;
   v6 = temp;
}
```

Since v5 is passed by address and v6 is passed by reference, this function swaps the values of v6 and whatever v5 points to

```
int main() {
   int x = 10;
   int y = 20;
   int z = 30;

   cout << x << " " << y << " " << z << endl;

   f1(x, y);
   f2(&y, z);
   f3(&z, x);

   cout << x << " " << y << " " << z << endl;

   return 0;
}
```

Sets y = x = 10

Sets y = z = 30

Swaps x and z, so x = 30 and z = 10

OUTPUT:

```
10 20 30
30 30 10
```

4. (14 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("question"), s2("act"), s3;
   unsigned i;

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

   i = s2.length();        i = 3

   if (s2.empty())
      cout << s2 << s1[i] << endl;
   else                                          Since s2 isn't empty,
      cout << s1.at(i + 1) << s2 << endl;          prints s1[4] = 't'
                                                   followed by s2

   if (s3.empty())                        Since s3 is empty, sets s3 =
      s3 = s2 + s1.substr(5);               "act" + "ion" = "action"
   else
      s3 = s2[2] + s1.substr(2, 3);

   s2 += "ed"   s2 = "acted"

   cout << s1.substr(0, 5) << " " << s2 << " " <<  s3 << endl;

   return 0;
}
```

OUTPUT:

```
s1 = question, s2 = act, s3 =
tact
quest acted action
```

5. (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;
1   int total = 1;                  1
2   if (n > 0) {                    1
3       for (i = n; i > 1; i--)     1 + n + (n – 1) = 2n
4           total = total * i;      n - 1
    }
5   if (n < 0) {                    1
6       for (i = n; i < 0; i++)     1 + (n + 1) + n = 2n + 2
7           total = total + i;      n
    }
8   if (n % 2 == 0) {               1
9       for (i = 0; i < n; i += 2)  1 + (n/2 + 1) + (n/2) = n + 2
10          total = total + i;      n/2
    }
11  return total;                   1
}
```

Worst case analysis: You can't just add all of these terms—although all conditions will be tested (lines 2, 5, and 8) since the if statements are independent, all 3 conditions can't be true. If the condition on line 2 is true, the one on line 5 can't be, and vice versa.

Since lines 6 and 7 execute more steps than lines 3 and 4, the worst case occurs when n is a negative number divisible by 2. The program would therefore execute every line except lines 3 and 4, giving a worst case execution time of:

$$T(n) = 1 + 1 + 1 + 2n + 2 + n + 1 + n + 2 + n/2 + 1$$
$$= 4.5n + 9$$

The order of magnitude for the worst-case execution time is therefore **O(n)**.

5 (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   for (i = 0; i < n; i++) {                  1 + (n + 1) + n
                                               = 2n + 2
2       for (j = n – 1; j > 0; j = j / 2) {    n * (1 + (log2(n) +
                                                 1) + log2(n))
                                               = 2n log2(n) + 2n
3           if (arr[i] > arr[j])               n log2(n)
4               arr[j] = arr[i];               n * X (see below)
            else
5               arr[i] = arr[j];               n * Y (see below)
        }
    }
}
```

Worst case analysis: The body of the outer loop executes n times, so the number of steps for each line in the inner loop (in blue) is multiplied by n to get the total number of steps executed as the outer loop executes.

Since your inner loop index is divided by 2 each iteration, that loop's execution time is a function of $\log_2(n)$, as shown.

We have no idea what X and Y are—without the values in an array, there's no way to tell how many times line 3 will evaluate as true and how many times the else block will execute instead. However, we do know that either line 4 or line 5 will execute every time through the inner loop, so X + Y must equal the number of inner loop iterations—$\log_2(n)$.

The worst case execution time is therefore:

$$T(n) = 2n + 2 + 2n \log_2(n) + 2n + n \log_2(n) + n \log_2(n)$$
$$= 4n \log_2(n) + 4n + 2$$

The order of magnitude for the worst case execution time is **$O(n \log_2(n))$**.

6. (16 points, 4 points each) ***Classes***

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.

NOTE: All parts of this question refer to the class `E1Class`, which is defined on the extra sheet included with the exam.

a. Which of the following statements is a valid declaration for an object of type `E1Class`? **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. ***`E1Class ec1;`***

ii. ***`E1Class ec2(1, 'a', 2.3);`***

iii. `E1Class ec3(4, 5, 'b', 'c', 6.7, 8.9);`

iv. `E1Class ec4.changeAll(0, 0, 'x', 'y', 32.20, 2.25);`

v. `E1Class ec5.E1Class();`

b. Which of the following statements will compile and execute without errors, assuming the object `E1Class ec` has been properly initialized? **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. ***`ec.changeAll(9, 8, '7', '6', 5.4, 3.2);`***

ii. `ec.m1 = ec.m2;`

iii. ***`ec.printVars(cout);`***

iv. `ec.E1Class(1, '2', 3.4);`

v. `int x = ec.someHelp(5);`

6 (continued)

c. Which of the following choices represents a valid implementation of one of the E1Class constructors? **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.
```
E1Class() {
   m1 = 0;
   m2 = 0;
   m3 = 'e';
   m4 = '1';
   m5 = 1.5;
   m6 = 9.9;
}
```

ii.
```
E1Class::E1Class() {
   m1 = 5;
   m2 = 6;
   m3 = 'x';
   m4 = m3;
   m5 = m1;
   m6 = m2;
}
```

iii.
```
E1Class::E1Class(int i, char c, double d) {
   m1 = i;
   m2 = i + 2;
   m3 = c;
   m4 = c + m1;
   m5 = d;
   m6 = d / 2;
}
```

iv.
```
E1Class::E1Class(int i1, int i2, char c1, char c2,
                 double d1, double d2) {
   m1 = i1;
   m2 = i2;
   m3 = c1;
   m4 = c2;
   m5 = d1;
   m6 = d2;
}
```

6 (continued)

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."

All of the above are "correct."

7. (10 points) ***EXTRA CREDIT***

Complete the function below, which finds the longest substring shared by its two string arguments, `s1` and `s2`. For example:

- `maxSubMatch("test 1", "test2")` returns `"test"`
- `maxSubMatch("EECE.3220", "19320")` returns `"32"`
- `maxSubMatch("test", "Exam")` returns `""` (empty string)

```
string maxSubMatch(string s1, string s2) {
   string t1, t2;         // Temporary substrings for comparison
   string longest;        // Longest matching substring

   unsigned i, j, k;      // Loop indices
   unsigned len;          // Length of shortest string

   // Go through both strings, comparing all substrings of
   //   same length, and find the longest one

   // Determine length of shortest string to limit outer loop
   if (s1.length() > s2.length())
      len = s2.length();
   else
      len = s1.length();

   // Outer loop--cover substrings of length 1 --> len
   for (i = 1; i <= len; i++) {

      // Middle loop--starting position for substrings from s1
      // Finds substrings of length i starting at position j
      for (j = 0; j <= s1.length() - i; j++) {
         t1 = s1.substr(j, i);

         // Inner loop--starting position for substrings from s2
         // Finds substrings of length i starting at position j
         for (k = 0; k <= s2.length() - i; k++) {
            t2 = s2.substr(k, i);

            // Compare substrings—is there a new longer match?
            if (t1 == t2 && t1.length() > longest.length())
                 longest = t1;
         }
      }
   }

   return longest;
}
```