EECE 230 — Introduction to Programming Using C++, Sections 10,11 and 12 — Quiz I

March 8, 2018

- The duration of this exam is 2 hours and 45 minutes. Keep in mind that you need around 10 minutes at the end of the duration of the exam to submit your answers. It is your responsibility to make sure your files are correctly submitted.
- The exam consists of 4 problems for 130 points (+35 bonus points).
- You can use all the material available on "Volume D:" in the EECE230Files folder. In particular, the folder contains lecture slides, source code from the lectures, programming assignments, and their solutions.
- A directory "D:\EECE230Files\Submit" will be created on your machine and will be dedicated to collect your solutions. Make sure you copy your **cpp files** (files ending with the .cpp extension) to that directory. Failure to do so may lead to a failing grade on the exam.
- You are **NOT** allowed to use the **web**. You are not allowed to use **USB's** or files previously stored on your machine other than files in the EECE230Files directory.
- If you get caught violating the above rules or if you communicate with a person other than the exam proctors during the exam, you will immediately get zero and you will be referred to the appropriate disciplinary committee.
- Cell phones and any other unauthorized electronic devices are absolutely not allowed in the exam rooms. They should be turned off and put away.
- The problems are of varying difficulty. Below is rough ordering estimate of the problems in order of increasing difficulty.
  - Level 1 (50 points): Problem 1, Part 2.a, and Parts 4.a and 4.b
  - Level 2 (35 points): Part 2.b and Part 3.a
  - Level 3 (30 points): Inefficient solutions of Parts 2.c and 4.c.
  - Level 4 (15 points + 15 bonus points): Parts 3.b and 3.c.
  - Level 5 (20 bonus points): Efficient Part 2.c, and array-free solution of Problem 4.

  Interpretation of bonus points: The grading concept of Level 5 is asymmetric. If you skip them all, you loose no points points. On the other extreme, if you solve them all correctly, you will get 20 points out of 0 points. You are advised to leave them till the end.
- Detailed comments are worth partial credit.
- Plan your time wisely. Do not spend too much time on any one problem. Read through all of them first and attack them in the order that allows you to make the most progress.
- Submit your solutions each part in a separate file as indicated in the booklet. Include your name and ID number in each file.
- Good luck!

Problem 1 (30 points). Cartesian and Polar coordinates

a. **(5 points).** Prompt the user to enter the polar coordinates ρ and ϕ of a point in a polar coordinate system. Consider the angle entered in radians.

b. **(7 points).** Convert coordinates into a Cartesian coordinate system where $x = \rho\ cos(\phi)$ and $y = \rho\ sin(\phi)$. Print the Cartesian coordinates of the point. You may use the `cos` and `sin` functions from the `<cmath>` library. Both functions consider the angle given in radians.

c. **(8 points).** Check whether the entered point is inside the box whose top left and bottom right corners are at $(1.1, 5.5)$ and $(6.6, 0.9)$ in the Cartesian coordinate system, respectively. If the point is inside the box, print "Inside box", otherwise print "Outside box".

d. **(10 points).** Take another point with polar coordinates ρ_1 and ϕ_1 and compute the distance between the two points. The distance between two Cartesian points (x_1, y_1) and (x_2, y_2) is given by $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$.

Submit your solution in a file called `Prob1.cpp` including your name and ID number. Example runs follow.

```
Please enter the radius (rho) coordinates in cm:3.2
Please enter the angle (phi) coordinate in radians:1.1
The Cartesian coordinates are (1.45151,2.85186)
Inside box
Please enter the radius (rho) coordinates in cm:2.3
Please enter the angle (phi) coordinate in radians:1.5
The distance between the two points is: 1.40427cm.
```

```
Please enter the radius (rho) coordinates in cm:2.5
Please enter the angle (phi) coordinate in radians:2.1
The Cartesian coordinates are (-1.26212,2.15802)
Outside box
Please enter the radius (rho) coordinates in cm:3.4
Please enter the angle (phi) coordinate in radians:1.1
The distance between the two points is: 2.93681cm.
```

Problem 2 (55 points + 10 bonus points). Sum, diving score and auto-complete

a. **(15 points). Sum of an interval.**

Write a program that takes two integer numbers x_1 and x_2 from the user, computes the sum of all the integers between x_1 and x_2 inclusive and prints the result. For example, given $x_1 = 4$ and $x_2 = 8$ the result is $4 + 5 + 6 + 7 + 8 = 30$.

Submit your solution in a file called Prob2a.cpp including your name and ID number. Sample runs follow.

```
Enter two integers:2 10
The sum of numbers between 2 and 10 is: 54

Enter two integers:-4 12
The sum of numbers between -4 and 12 is: 68
```

```
Enter two integers:12 5
The sum of numbers between 5 and 12 is: 68

Enter two integers:-10 -2
The sum of numbers between -10 and -2 is: -54
```

b. **(20 points). Diving score.**

In the sport of diving, seven judges award a score between 0 and 10, where each score may be a real number. The highest and lowest scores are ignored. The remaining five scores are added together. The sum is then multiplied by the degree of difficulty for that dive. The degree of difficulty ranges from 1.2 to 3.8 points. The total is then multiplied by 0.6 to determine the diver's score.

Write a program that prompts the user for a degree of difficulty, and for seven scores from seven judges. The program then computes and outputs the overall score for the dive. The program should ensure that all inputs are within the allowable data ranges, otherwise, the program issues an error and exits.

Submit your solution in a file called Prob2b.cpp including your name and ID number. Sample runs follow.

```
Enter the difficulty level of the dive: 1.6
Enter score of judge 0: 2
Enter score of judge 1: 2.3
Enter score of judge 2: 5
Enter score of judge 3: 6.6
Enter score of judge 4: .7
Enter score of judge 5: 8.9
Enter score of judge 6: 8.1
The diving score is: 23.808
```

```
Enter the difficulty level of the dive: 12
Exit! Difficulty out of the [1.2,3.8] range.


Enter the difficulty level of the dive: 2.8
Enter score of judge 0: 7.3
Enter score of judge 1: 6.5
Enter score of judge 2: 4.5
Enter score of judge 3: 10.1
Exit! Scores[3] is out of [0.0,10.0] range.
```

c. **(20 points + 10 bonus points). Auto-complete.**

You are given an array of strings `string words[]` with possible duplicate words. You are also given the first two characters of a word in another string `needle`. Auto-complete prints the string in array `words` that starts with the two characters in `needle` and that occurs the most. For example, when the two characters are "ap" and the words are {"apple", "lime", "apple", "banana", "application"} your program should print "apple" since it occurs twice while the other matching string "application" occurs only once.

Any correct solution is worth 20 points. More efficient solutions are worth up to 10 additional points. (*Hint:* First find a way to have all duplicate words next to each other.)

Submit your solution in a file called Prob2c.cpp including your name and ID number. Example runs follow.

```
Words: act, ace, beep, act, ace, act, cat
Needle: ac
Suggestion: act

Words: act, ace, beep, act, ace, act, cat, ace
Needle: ac
Suggestion: act
```

```
Words: act, ace, beep, act, ace, act, ace, cat, ace
Needle: ac
Suggestion: ace

Words: act, ace, beep, act, ace, act, ace, cat, ace
Needle: bo
No Suggestion exists
```

Problem 3 (30 points + 15 bonus points). Array intersection

a. **(15 points). Intersection of two arrays.**

You are given two arrays of integers a and b of size n and m, respectively. Your program should compute an array of integers c that is the intersection of a and b. Array c should have no duplicates. For example, for $a = \{5, 1, 3, 2, 4, 5, 9, 5, 6\}$ and $b = \{5, 3, 5, 7, 6, 7, 9, 8\}$ the resulting array c should be $\{5, 3, 9, 6\}$. Your program should print arrays a, b and c. Any correct answer is worth 15 points.

Submit your solution in a file called Prob3a.cpp including your name and ID number. Sample runs follow.

```
a: 5 1 3 2 4 5 9 5 6
b: 5 3 5 7 6 7 9 8
c: 5 3 9 6
```

```
a: 5 1 3 2 4 5 9 5 6
b: 10 11 14 10 11
c: empty
```

b. **(10 points). Intersection of two *sorted* arrays.**

Arrays a and b are now given as *sorted* and n is *much larger* than m, ($n \gg m$). For example, for $a = \{1, 2, 3, 4, 4, 5, 5, 6, 9, 10, 10, \ldots, 10, 13, \ldots, 13, \ldots, 1023, \ldots, 5432\}$ and $b = \{3, 5, 5, 6, 6, 7, 8, 1023\}$ the resulting intersection with no duplicates array c should be $\{3, 5, 6, 1023\}$. Provide a program faster than that of Part 3.a that computes c for sorted arrays a and b where $n \gg m$. In particular, your program should run in proportional to $m \log n$ operations in terms of runtime.

Submit your solution in a file called Prob3b.cpp including your name and ID number.

c. **(5 points + 15 Bonus points). Intersection of *similar length* and *sorted* arrays.**

Arrays a and b are now sorted and n and m are approximately equal. For example, for $a = \{1, 2, 3, 4, 5, 5, 6, 9\}$ and $b = \{3, 5, 5, 5, 6, 6, 7, 9\}$ the resulting intersection with no duplicates array c should be $\{3, 5, 6, 9\}$. Provide a program that is faster than both programs given in 3.a and 3.b to compute c for sorted arrays a and b where $n \approx m$. The program should run in proportional to $m + n$ operations in terms of run time.

Submit your solution in a file called Prob3c.cpp including your name and ID number.

Problem 4 (15 points + 10 bonus points). Separate lower and and upper case letters

We are given a string *str* with n alphabetical characters. We would like to separate the lower case characters of *str* from the upper case characters of *str* such that all the lower case characters appear before the upper case characters.

More precisely, we want to transform *str* such that the substring $str[0 \ldots q-1]$ is all lower case characters, and the substring $str[q \ldots n-1]$ is all upper case characters where q is the number of lower case characters in *str*. The resulting string *str* must be also a rearrangement of the original string *str*.

We do not care about the order of the characters within $str[0 \ldots q-1]$ and $str[q \ldots n-1]$.

Examples follows.

- If *str* was "ABaBCDdbEF" then a valid solution would be "adbABBCDEF". Other valid solutions are "dbaBBADCEF" and "bdaFECDABB".
- If *str* was "AaAaAa" then the only valid solution is "aaaAAA".
- If *str* was "ABAC" then a valid solution is "ABAC".

a. **(3 points). Is lower case.** Write a program that checks whether a character `char c` represents a lower case letter between 'a' and 'z' inclusive.

b. **(2 points). Is upper case.** Write a program that checks whether a character `char c` represents an upper case letter between 'A' and 'Z' inclusive.

c. Write a program that takes a string *str*, computes and prints a rearrangement of *str* as described above.

Any valid solution is worth 10 points. If you do it efficiently while using an auxiliary array, you get 20 points. If you do it efficiently and without using an auxiliary array, you get 25 points.

Submit your solution is a file called Prob4.cpp including your name and ID nunber.