**ECE 1310 Final Part 2 Group**

**Work Alone: \_\_\_X\_\_\_\_ (please mark)**

**Or Work in a group: \_\_\_\_\_\_\_\_\_\_ (write down the group #)**

**Names in the group:**

**Date:** 12/4/24, Wednesday, 10 a.m., week 15

**Due Date:** 12/11/24, Wednesday, 11 p.m..

**Total Points**: 80 points (without extra credits) or 92 points with extra credits.

**Total Grade Points: 15 grade points** (80 points = 15 grade points, 92 points = 17.25 grade points.)

You can work on this exam for a group of up to **5 (five) students** (clearly indicate who and who are in the group. It is open books as usual. Submit your Word document answer (**just one, not severa**l) together with your source code etc. (supposed to be no bigger than 1 M bytes) to the folder ECE 1310 Final Part 2 Group.

If you choose to work alone but your are in a group. Please clearly indicate that in the beginning.

**Play with arrays (37%)**

1. (12%) Determine array size and perform some operations.

Consider a simple array int a[] = {10, 30, 50, 70, 90, 100, 80, 60, 40, 20};

You can see clearly that the size of a is 10 by counting *manually*.

1. **(3%) Write** a C program to find the size of array a using the keyword sizeof (sizeof is explained in your textbook and powerpoint file). Turn in the C code plus the output to prove the size of a is 10.
2. **(3%) Modify** the program in part (a) using a different approach, by int b[20] = {10, 30, 50, 70, 90, 100, 80, 60, 40, 20};

Now write a for loop looping 20 times and count the # of nonzero elements. It should be 10. Verify that by turning in your source code (enhanced or modified from part (a)) and show in the output that the size of b is also 10.

The following is output from my computer

A black background with white text

Description automatically generated

You should show something similar

1. (2%) Compute the maximum of array a and display that.
2. (4%) Sort array a (using any sorting algorithm you know such as insertion sort) and display the output.
3. (25%) Redo Q1 with all 4 parts, but this time, use a much bigger array with area code:

In US, long distance phone call is made by dialing the three digits area code first, followed by 7 digits phone number (ECE department office number is 909-869-4609 for example).

The following is a list of some US area code using JavaScript code (not the most current. There are more)

var AreaCode =new Array (205,251,659,256,334,907,403,780,264,268,520,928,480,602,623,501,479,870,242,246,441,250,604,778,284,341,442,628,657,669,747,752,764,951,209,559,408,831,510,213,310,424,323,562,707,369,627,530,714,949,626,909,916,760,619,858,935,818,415,925,661,805,650,600,809,345,670,211,720,970,303,719,203,475,860,959,302,411,202,767,911,239,386,689,754,941,954,561,407,727,352,904,850,786,863,305,321,813,470,478,770,678,404,706,912,229,710,473,671,808,208,312,773,630,847,708,815,224,331,464,872,217,618,309,260,317,219,765,812,563,641,515,319,712,876,620,785,913,316,270,859,606,502,225,337,985,504,318,318,204,227,240,443,667,410,301,339,351,774,781,857,978,508,617,413,231,269,989,734,517,313,810,248,278,586,679,947,906,616,320,612,763,952,218,507,651,228,601,557,573,636,660,975,314,816,417,664,406,402,308,775,702,506,603,551,848,862,732,908,201,973,609,856,505,575,585,845,917,516,212,646,315,518,347,718,607,914,631,716,709,252,336,828,910,980,984,919,704,701,283,380,567,216,614,937,330,234,440,419,740,513,580,918,405,905,289,647,705,807,613,519,416,503,541,971,445,610,835,878,484,717,570,412,215,267,814,724,902,787,939,438,450,819,418,514,401,306,803,843,864,605,869,758,784,731,865,931,423,615,901,325,361,430,432,469,682,737,979,214,972,254,940,713,281,832,956,817,806,903,210,830,409,936,512,915,868,649,340,385,435,801,802,276,434,540,571,757,703,804,509,206,425,253,360,564,304,262,920,414,715,608,307,867)

Note you need to convert Javascript array (205, 251,… 307, 867) to the format {205, 251, .., 307, 867}; this is straightforward by changing “(“ to “{“ etc.

1. (13%) Compute the size of this very long array using sizeof like Q1(a) or use an integer n large enough to contain all the entries of the Javascript array (n equal to 1000, or safer 1200 should be enough since area codes are all 3 digits, so the total number can not be bigger than 1000).
2. (4%) Compute the maximum of the area code array
3. (8%) Sort the area code array

**Print Calendar** (25 points)

A screenshot of a calendar

Description automatically generated

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  | |  | | |  | |  | |  | |  |
|  | Dec, 2024 | | | |  |  | |  | |  | |
|  |  | |  | |  |  | |  | |  | |
| Su | Mo | | Tu | | We | Th | | Fr | | Sa | |
| 1 | 2 | | 3 | | 4 | 5 | | 6 | | 7 | |
| 8 | 9 | | 10 | | 11 | 12 | | 13 | | 14 | |
| 15 | 16 | | 17 | | 18 | 19 | | 20 | | 21 | |
| 22 | 23 | | 24 | | 25 | 26 | | 27 | | 28 | |
| 29 | 30 | |  | |  |  | |  | |  | |

Figure 1 Figure 2

1. (25%) **Write** a C program that takes an integer m between 1 and 12 as the month number and **prints** out a calendar like Figure 2 above. **Show** your design and code clearly in the Word file and also turn in your cpp code. **Do not use 7 cout statements to print out this calendar (use for loop).** (Figure 1 is how Windows 11 prints, but the instructor wants you to print out like Figure 2)

When you enter m = 6 (the sixth month or June), your program will display on top June, 2024, then a row of Su, Mo, Tu, etc (Sunday, Monday, Tuesday…), and then 5 rows of numbers 1, 2, .., through 7 in first row, 8, 9, .., through 14 in second row, .., , and finally 29, and 30 in the last row.

**Run** your program for m = 6 (June), m = 2 (February) and m = 12 (December) and also another month of your choice. Note you can use the fact that December 1, 2024 starts on Sunday.

Note: Windows 11 seems to have a problem of printing the calendar correctly (while Windows 10 did that correctly).

The following Figure 3 shows how June 2024 is printed in Windows 11. June 1 prints on the 2nd rows (1st row prints the last few days of May 2024) and the calendar only prints up to June 15, which is wrong! The real calendar of June should start with June 1 in the 1st row on Saturday, and prints up to June 30 on Sunday, on the 6th row. Print your June calendar like what the instructor asks you to do in Figure 2, NOT like in Figure 3 (or Figure 1), the wrong way as in Windows 11.

A calendar with numbers and letters

Description automatically generated

Figure 3.

Computation of mathematical constant .( 18%)

1. (18%) Computation of  using infinite series.

  3.14159 is a mathematical constant you have learned since elementary school. Here we will use infinite series you learned in calculus to approximate the value of  using the formula . (here , is also known as arctan (x) as the inverse function of tan (x). In programming, you have to use atan (x) and include <cmath> in C programming)

* 1. **Derive**  = x – x3/3 + x5 / 5 + … from 1/ (1 + x2) = 1 – x2 + x4 – x6 ….
  2. What is the range that 1/ (1 + x2) = 1 – x2 + x4 – x6 …. expansion holds true? It is obviously this is true for x = 0 and not true for x = -3 since the expansion gets bigger and bigger. This is also true for x = 0.1 since it is a geometric series with ratio < 1.
  3. Write down how the n-th term looks like. Especially show the coefficient. Is that + or – or depends?

Since tan (/4) = 1 from trigonometry, we have  = 4 - +….

From the theory of infinite alternating series, we know that when we stop at the nth term, the error EPSILON is at most the absolute value of nth term. Hence you may compute n based on the value of the nth term.

* 1. **Write** a C / C++ program to compute the value of  using for loop. The for loop will look like for (i = 0; double (4.0/(2\*i + 1)) >= EPSILON ; i ++) with EPSILON = 1.0e-7 (=0.0000001) or 10-7.
  2. **Show** how your computed  looks like if you set EPSILON = 1.0e-5 (=0.00001)**.**
  3. How many terms n do you need to compute with EPSILON = 1.0e-7? Note n is at least 1,000.

1. (12%) **Fast** computation of p

I heard about this trigonometric formula from Dr. Chandra.

(this can be derived from the sum and difference angle formula of sine and cosine functions that you learned in high school).

From this we have 

Mathematically = ½ - (1/2)3/3 + (1/2)5/5 + …

* 1. Compute manually or programmatically when the k th term of  has absolute value less than EPSILON = 1.0e-7 = 0.0000001. Note this k must be < 50 (much smaller than n >= 1,000 for  to get EPSILON = 1.0e-7).
  2. Compute manually or programmatically when the m th term of  has absolute value less than EPSILON = 1.0e-7 = 0.0000001. Note this m must be < 20 (much smaller than n >= 1,000 for  to get EPSILON = 1.0e-7).

From the results of (a) and (b) we know that using arctan of ½ and 1/3, we can compute p precisely to 10-7 inless than 50 terms, which is much better than over 1,000 terms of #6.