Due before:	11:59 PM (before Midnight) on Tuesday June 23, 2020

CSE278: Introduction to Systems Programming (Systems I)

Lab #5

Due:Tuesday June 23, 2020 before 11:59 PM

Email-based help Cutoff: 5:00 PM on Mon, June 22, 2020

Maximum Points: 50

Submission Instructions

This part of the homework assignment must be turned-in electronically via Canvas. Ensure you name this document Lab5_*MUID*.docx, where *MUid* is your Miami University unique ID. (Example: Lab5_ahmede.docx)

Copy pasting from online resources is **Plagiarism**. Instead you should read, understand, and use your own words to respond to questions.

Submission Instructions:
Once you have completed answering the questions save this document as a PDF file (**don't just rename the document; that is not the correct way to save as PDF**) and upload it to Canvas.

General Note: Upload each file associated with homework (or lab exercises) individually to Canvas. Do not upload archive file formats such as zip/tar/gz/7zip/rar etc.

Objective

The objective of this Lab is to review basic concepts of:

- Basics of Number Systems
- Basics of computer architecture and organization
- Pointers and Memory
- Use CODE Plugin as part of functional testing

Name: **William Mechler**

Required reading

- Lecture Slides and ClassNotes: NumberRepresentation
- Lecture Slides and ClassNotes: ComputerArchitecture
- Lecture Slides and ClassNotes: Pointers

Due before:	11:59 PM (before Midnight) on Tuesday June 23, 2020

1. Convert hexadecimal 765F to octal. (Hint: First convert 765F to binary, then convert that binary number to octal). *Show all of your mathematical work.*

2. Convert decimal 299 to binary, to octal and to hexadecimal. *Show all of your mathematical work.*

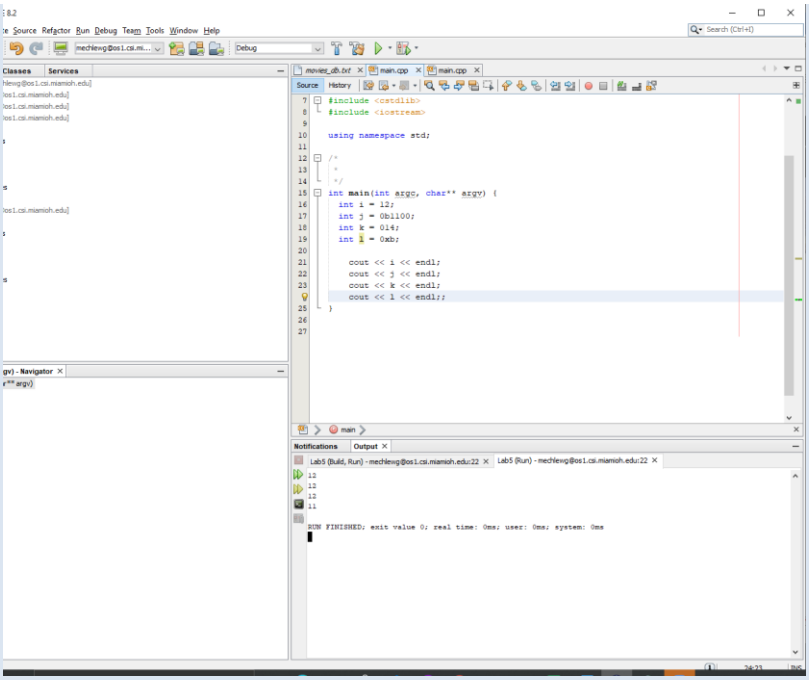
3. **Literals in C++**

   C++ supports all 4 number systems to represent numbers: – Normally numbers are treated as decimals – If number starts with 0b it is considered binary – If number starts with 0 it is considered octal – If number starts with 0x it is considered hexadecimal. Print the value of the following code snippet:

```
int main() {
        int i = 12;     // decimal
        int j = 0b1100; // 12 in binary
        int k = 014;    // 12 in octal
        int l = 0xb;    // 11 in hexadecimal
}
```

   Print the values from main suitably calling `cout`

   Attach a screen shot of the result

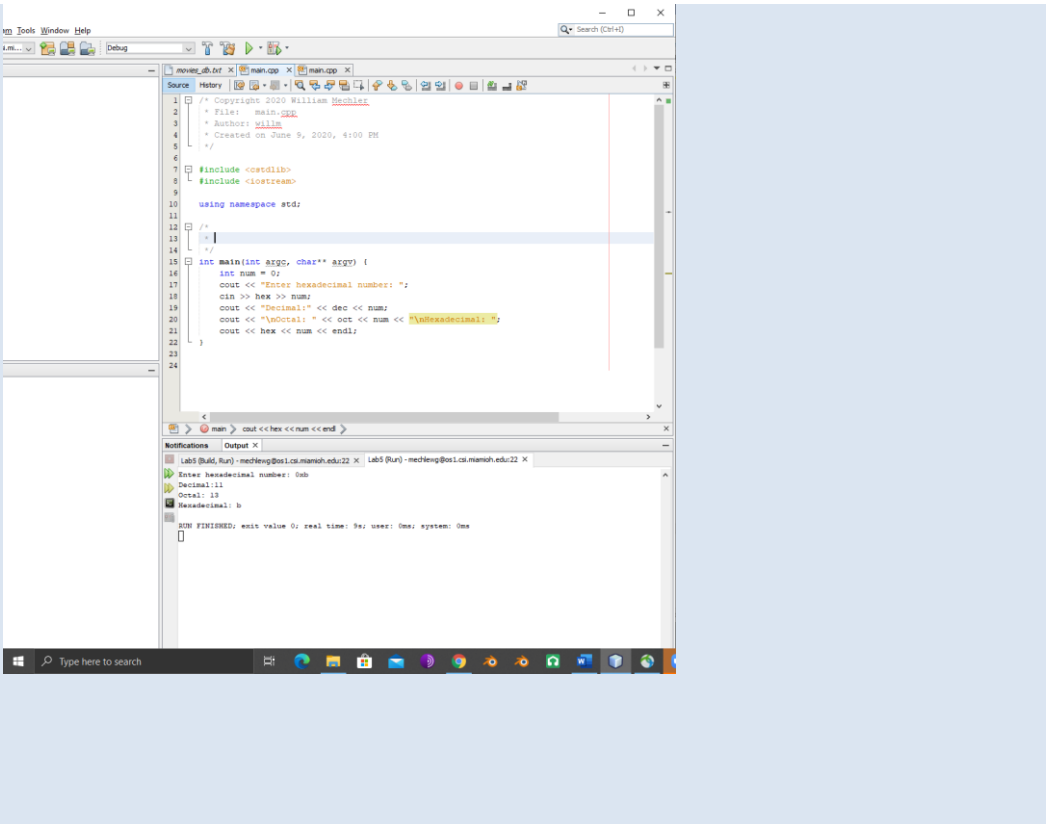


4. Number I/O in different bases

Run the following code:

```
int main() {
        int num  = 0;
        std::cout << "Enter hexadecimal number: ";
        std::cin  >> std::hex >> num;
        // Print number in various bases
        std::cout << "Decimal: "       << std::dec << num
                  << "\nOctal: "       << std::oct << num
                  << "\nHexadecimal: " << std::hex << num
                  << std::endl;
        }
```

Attach a screen shot of the result



5. Write a Complete C++ program which can produce the same result as shown below, i.e. covert number *procedurally* from decimal to binary, octal and hexadecimal. Microsoft Office 365 Excel support number conversion from one base to another, for example from Decimal to Binary, Octal, Hexadecimal etc. One such example screen shot is attached here:

Decimal	Binary	Octal	Hexadecimal
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D

14	1110	16	E
15	1111	17	F

6. Briefly (max 3 sentences each) describe the functionality of following key components of the Central Processing Unit (CPU) [**2 points**]

- ALU: ALU stands for arithmetic & logic unit which performs all operations in the CPU as a number.
- Registers: The fastest temporary storage in the CPU and tightly integrated with registers.
- Caches: Special high speed memory, and maintains copy of subset of data in RAM to ensure fast access.

7. What is the "Von Neumann" or "Stored Program" architecture? [**1 point**]

Von Newmann is the concept of where data and instructions are stored in the same memory.

8. What is a memory address? [**1 points**]

A memory address is the index into RAM typically written in hexadecimal.

9. What is assembly language? State 1 advantage and 1 disadvantage of assembly language [**2 points**]

Assembly language is high level mnemonic representation using variables. One advantage is it enables development of ultra-optimized code tailored for a specific microprocessor. However one disadvantage is that tailored code for specific microprocessor and operating system is less portable then general solutions.

Q9. Write a complete C++ program which reads one *hexadecimal* number from user and print its equivalent *decimal* number. You should write a function **hex2dec** which can take one argument command line parameter argv[1] as string that can represent the hex number. The function should define *int* variable and return the value of the variable as the decimal number. As for your hints, I am attaching the code **bin2Dec**() that we covered earlier in the Lab0:

```
int bin2dec (int binaryNum) {
 int decimalNum = 0;
 int remainder;
int counter = 0;

    while (binaryNum > 0){
       remainder = binaryNum % 10;
       decimalNum += remainder * pow(2, counter);
       counter++;
       binaryNum /= 10;
    }
    return decimalNum;
}
```

The process of computing the decimal number from hexadecimal is similar, here the power to be used is 16. You need to know the length of the string that will be used as the highest power, then inside a *for/while* loop, accumulate the equivalent multiplied by the respective power of 16.

Functional Requirements of the Code are as follows:

a. main() function should be able to handle *one* command line string parameter
b. Declare a function prototype: int hex2Dec(string input)
c. Implement the function (use `stoi()` and other string processing API)
d. Call the implemented function from main()
e. Print the result using `std::cout`

SAMPLE I/O

./MyHex2Dec 123
291

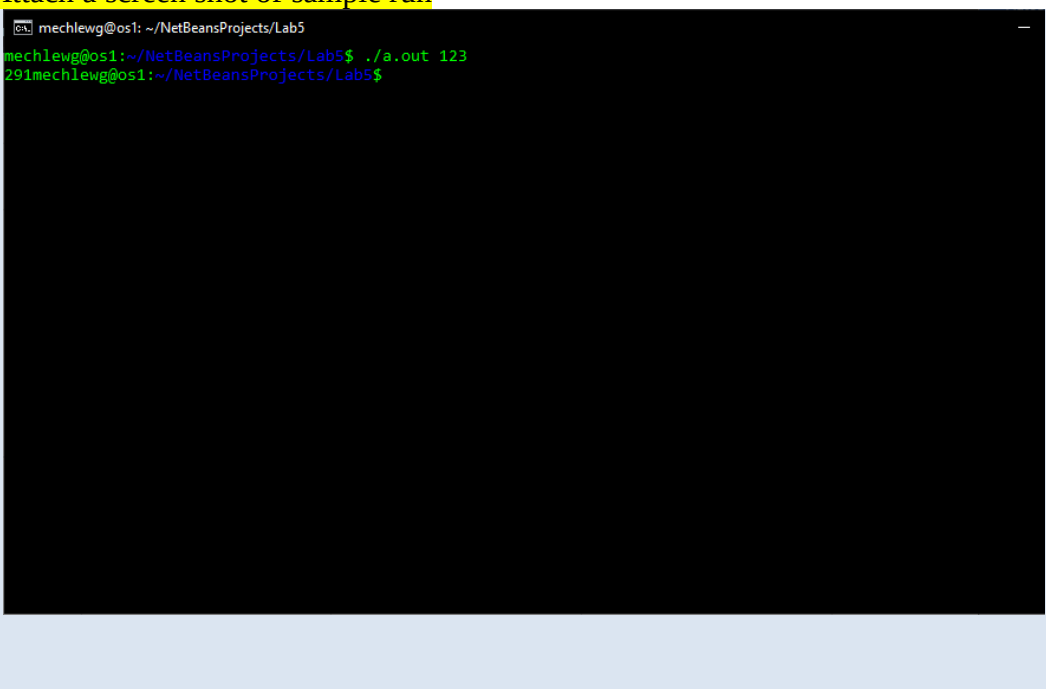
./MyHex2Dec 0x123
291

./MyHex2Dec 0X123
291

./MyHex2Dec 0x765f
30303

./MyHex2Dec 0X765F
30303

Attach a screen shot of sample run



Q10: Using Pointers to Iterate Loops
[10 points]

Program: In this lab, the goal is to write a program that utilizes points to traverse an array of numbers specified by a user. The program will get a set of numbers from standard input. It should use pointers to insert the numbers into the array and then use pointers to traverse the array in reverse while printing the memory address and the factors of the number located at that memory location to which it points. A sample output is provided. **The memory address and the even/odd should be comma separated and printed to standard output.**

The program will:

1. Read one command-line argument: the number of elements in the array **[1 point]**
    a. Example, the following command would ask the user for 5 numbers:
        ./lab5 5
2. Insert the numbers entered by the user into an integer array using pointers. **[2 points]**
3. Traverse the list in reverse using pointers. **[2 points]**

4. Print the value at the memory address to which it points and then print the factors of the value at the memory address to which it points (sample output below). **[2 points]**
    a. **NOTE: The output should *not* contain 1 and the number itself**
5. ***Code submission MUST be done through the code plug-in on Canvas for credit (instructions on the next page)***
6. Pass all the Test Cases [3 points]

SAMPLE OUTPUT:

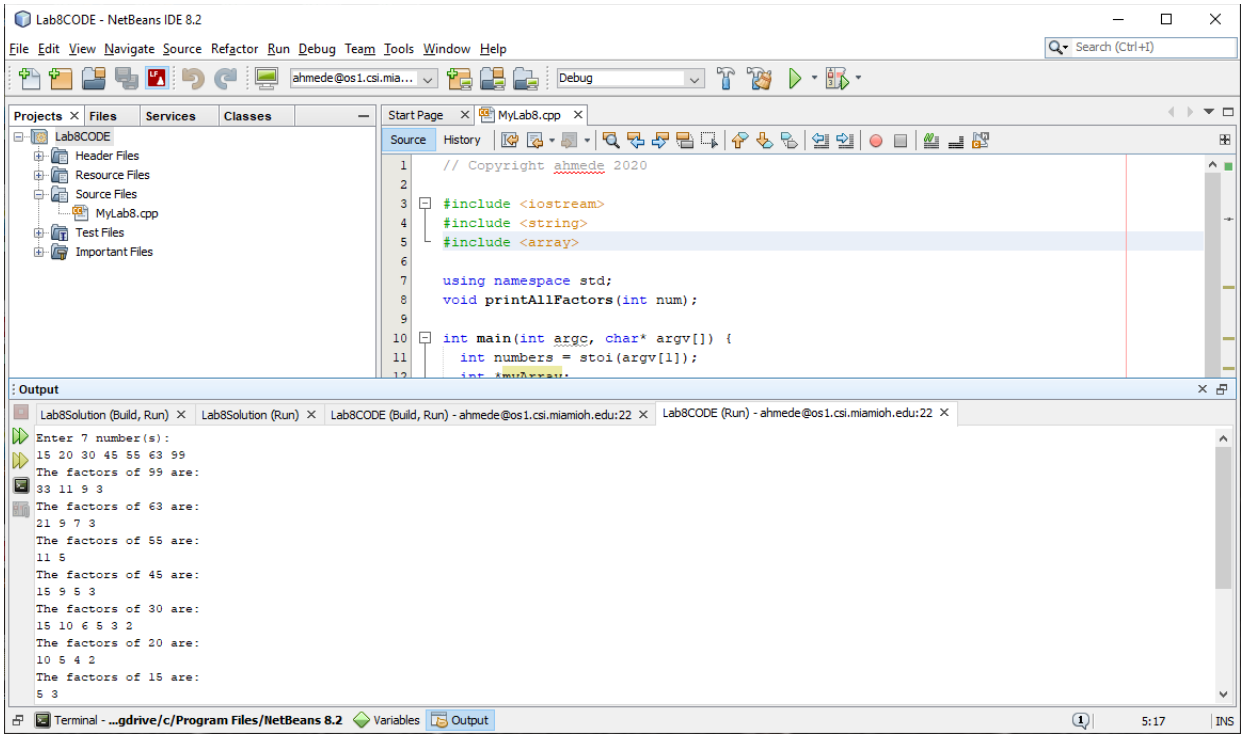
```
cvendome@Dagobah:TestLab9$./Lab9 5
Enter 5 number(s):
2 23 21 10 100
The factors of 100 are:
50 25 20 10 5 4 2
The factors of 10 are:
5 2
The factors of 21 are:
7 3
The factors of 23 are:

The factors of 2 are:

cvendome@Dagobah:TestLab9$
```

NOTE: Operating on the array without using pointers will NOT receive partial credit.

Due before:	11:59 PM (before Midnight) on Tuesday June 23, 2020



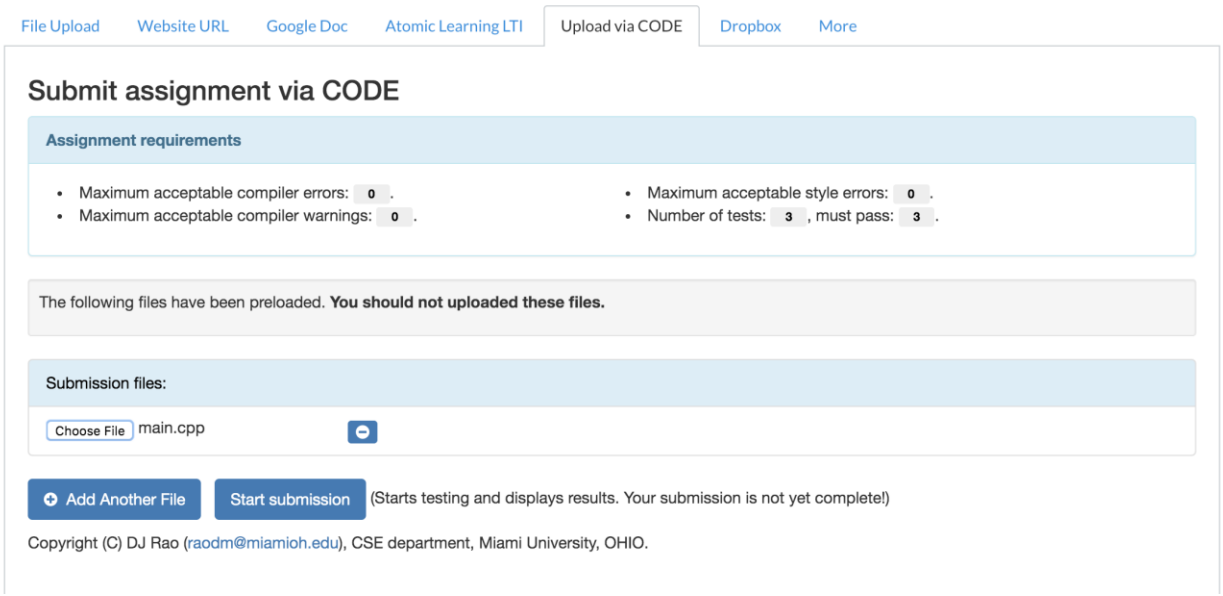
Submit to Canvas

There are **THREE** assignments on canvas – one for the program and *two* for code submission

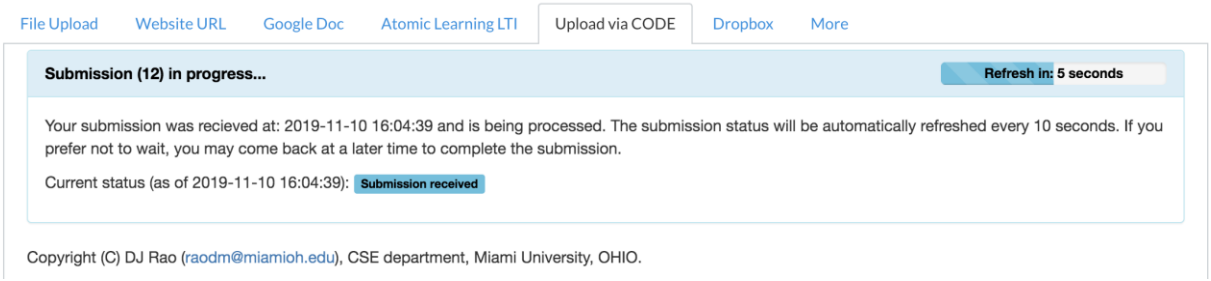
1. The document must be submitted as a PDF (30 points)
2. Q9 cpp code must be submitted through the CODE Plug-in (10 points)
3. Q10 cpp code must be submitted through the CODE Plug-in (10 points)

Submit to Code Plug-in

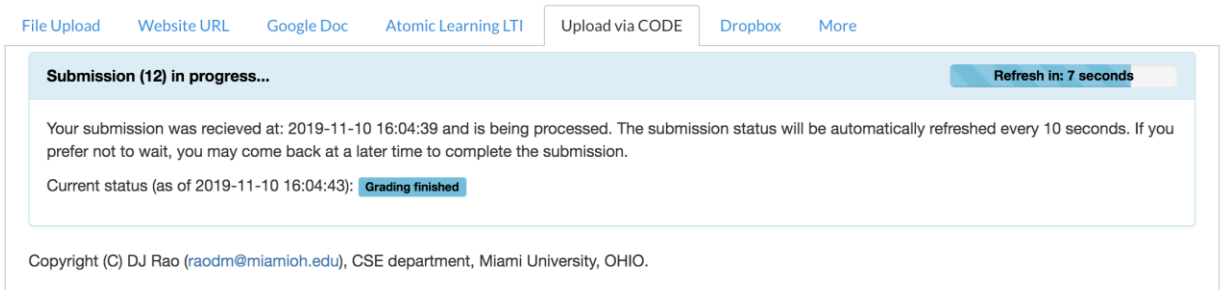
To submit, click the "Upload via CODE" tab then click "Choose File" to upload the source code



It will then submit the file

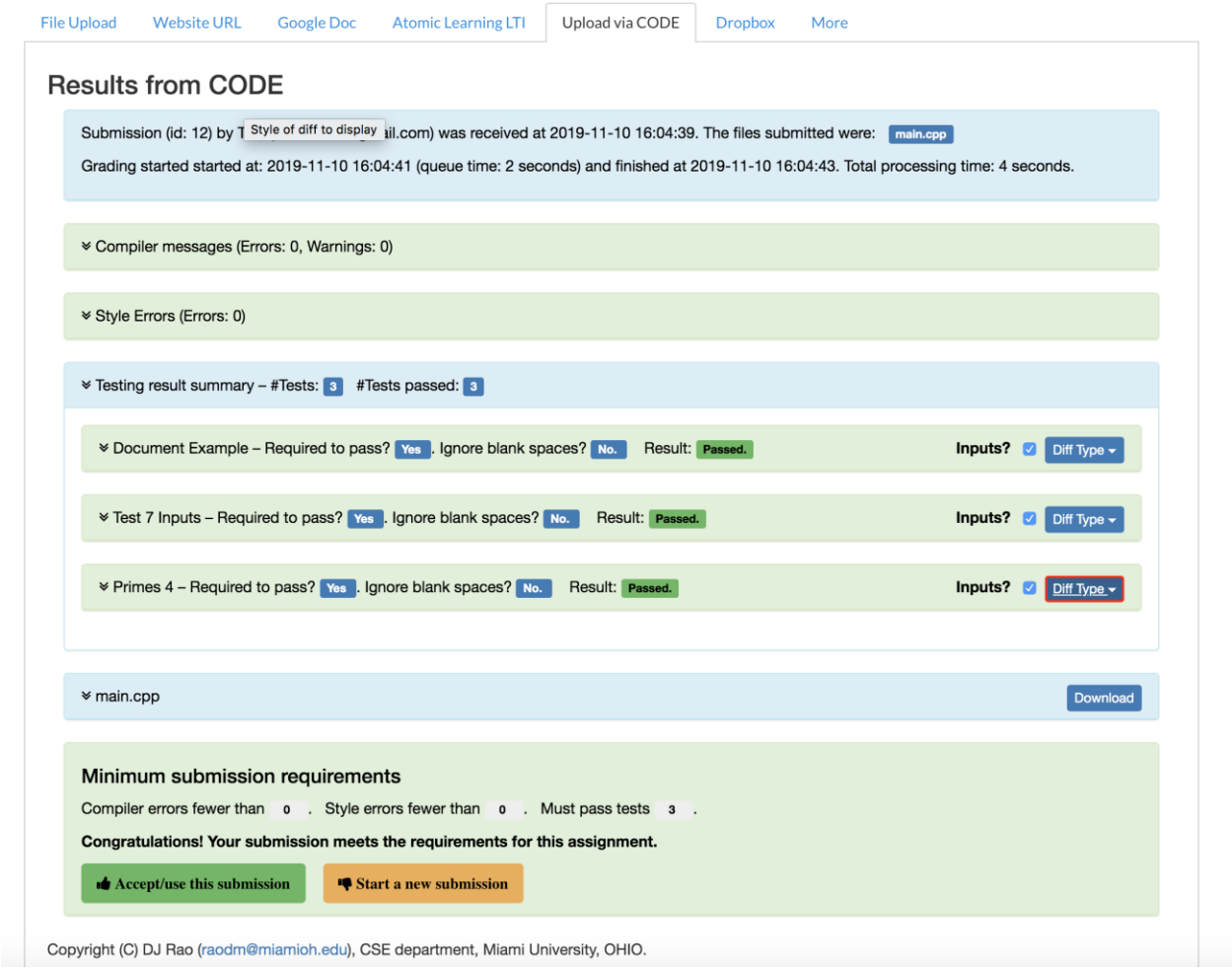


And subsequently run test cases on the program

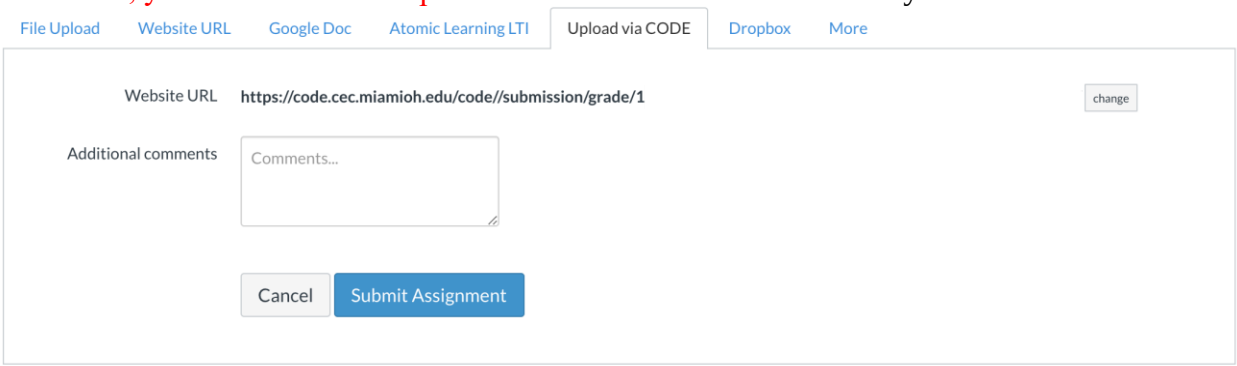


The results of the tests will be displayed and can be expanded:

Due before:	11:59 PM (before Midnight) on Tuesday June 23, 2020



To submit, you must click "Accept/use this submission" and it will let you submit:



Click submit assignment and it will submit the program.

Submission

- The submission file will be saved with the name ***Lab5_yourMUID.pdf***
- Assignment is due before Midnight Tuesday June 23, 2020.
- On or before the due time, drop the *electronic copy* of your work in the *canvas*
- Don't forget to Turn in the files! Lab5_yourMUID.pdf & Lab5_yourMUID*.cpp
- Also please submit the complete cpp code solution for Q9 and Q10 via CODE plugin