CSE 240 Homework 2 (75 points)

Introduction

The aim of this assignment is to make sure that you understand and are familiar with the concepts covered in the lectures, including programming paradigms, the structure of programming languages, and the differences between a macro and a procedure. By the end of the assignment, you should have

- exercised typing systems and operations of different typing systems.
- understood differences between the execution models of a macro and a function.
- gotten started with the programming environments Visual Studio and GCC.

This assignment is related to the outcomes 1-2 and 1-3 listed in the syllabus:

- learn strong vs. weak typing in computer programming languages
- understand the control structures of functional, logic, and imperative programming languages.
- understand the execution of functional, logic, and imperative programming languages.

Reading: Read text chapter 1, chapter 2 (sections 2.1, 2.2, and 2.3), appendix (sections B.1 andB.2), and course notes (slides).

Exercising: Complete the multiple choice questions in Textbook Section 1.7. The answers of the questions are available in course Web page.

You are expected to do the majority of the assignment outside the class meetings. Should you need assistance, or have questions about the assignment, please contact the instructor or the TA during their office hours.

You are encouraged to ask and answer questions on the course discussion board. (However, do not share your answers in the course discussion board.)

Programming Exercise (50 points)

Q&A Portion:

1) Review the lecture slides which discuss Very Simple Programming Languages (VSPL). Next, observe the VSPL defined below and identify which sequences are valid.

```
letter     ::= a | b | c | d | e
LETTER     ::= V | W | X | Y | Z
number     ::= 1 | 3 | 5 | 7 | 9
letters    ::= <letter> | <letter> <letters>
LETTERS    ::= <LETTER> | <LETTER> <LETTERS>
numbers    ::= <number> <number> | <number> <numbers>
sequence   ::= <letters> (LETTERS) <numbers> |
                  <LETTERS> (letters) <numbers>
```

Which of the following are valid sequences? You must clearly identify for each of the following sequences, which are valid and which are invalid. Each sequence is worth 1 point. Submit your answer in text document labeled hw02q1.txt. [10 points]

1. CSE240
2. aZ1
3. VWX97
4. Zbad99
5. eY55
6. dYaZeWkZ
7. XYZcde579
8. XYZabc3
9. abcXYZ123
10. edc135790V

2) Read C_part1.pptx and/or text section 1.4.2

Macros are available in most high-level programming languages. The body of a macro is simply used to replace a macro-call during the preprocessing stage. A macro introduces a "true inline" function that is normally more efficient than an "out-line" function. However, macros suffer from the side-effect, unwanted, or unexpected modifications to variables.

Macros should be used cautiously. The main purpose of the following programs are to demonstrate the differences between a function and a macro.

Other purposes include demonstrating the differences between different programming environments, and learning different ways of writing comments, formatted input and output, variable declaration and initialization, unary operation ++, macro definition/call, function definition/call, if-then-else and loop structures, etc.

Observe each of the functions below and understand their functionality. *You can use either GNU gcc under Unix or Visual Studio to implement the code in this question.*

```
int subf(int a, int b) {
	return a - b;
}

int cubef(int a) {
	return a * a * a;
}

int minf(int a, int b) {
	if (a <= b) {
		return a;
	} else {
		return b;
	}
}

int oddf(int a) {
	if (a % 2 == 0) {
		return 0;
	} else {
		return 1;
	}
}
```

2.1 Write four macros to re-implement the given four functions. Name them: subm, cubem, minm, and oddm, respectively. [8 points]

2.2 Write the main function to test the functions and macros. Use the following test cases in the main function to call your macros: [5 points]

```
int a = 5, b = 7;
subf(a, b);
subm(a, b);
subf(a++, b--);
subm(a++, b--);
cubef(a);
cubem(a);
cubef(--a);
cubem(--a);
minf(a, b);
minm(a, b);
minf(--a, --b);
minm(--a, --b);
oddf(a);
oddm(a);
oddf(a++);
oddm(a++);
```

Your main function must print the results of the test run.

For questions 2.1 and 2.2, submit your program labeled as hw02q2.c

2.3 Observe and compare with the outputs of the functions and the macros. Do the functions and macros generate the same results? Submit your short answer and the screenshots of the test run and label the file hw02q2.pdf [2 points]

Programming Portion:

3. You are given a file named hw02q3.c. All instructions are given in the form of comments in the file. You are to again run the file in both Visual Studio and in GCC and observe the outputs and make changes as requested. Please read all instructions very carefully, then complete and submit the updated file as hw02q3.c. [25 points]

Grading of Programming Assignment

The TA will grade your program following these steps:

(1) Compile the code. If it does not compile, 20% of the points given will be deducted. For example, if there are 20 points possible, you will earn 16 points if the program fails to compile.

(2) The TA will read your program and give points based on the points allocated to each component, the readability of your code (organization of the code and comments), logic, inclusion of the required functions, and correctness of the implementations of each function.

Rubric:

	Levels of Achievement						
Criteria	**A**	**B**	**C**	**D**	**E**	**U**	**F**
Q&A Portion **Weight 33.00%**	**100 %** All questions correct	**85 %** Mostly correct responses, some errors	**75 %** Some correct	**65 %** Partially correct	**35 %** Very little correct	**20 %** Very little correct	**0 %** Nothing correct
Specifications **Weight 34.00%**	**100 %** The program works and meets all of the specifications.	**85 %** The program works and produces the correct results and displays them correctly. It also meets most of the other specifications.	**75 %** The program produces mostly correct results but does not display them correctly and/or missing some specifications	**65 %** The program produces partially correct results, display problems and/or missing specifications	**35 %** Program compiles and runs and attempts specifications, but several problems exist	**20 %** Code does not compile and run. Produces excessive incorrect results	**0 %** Code does not compile. Barely an attempt was made at specifications.
Code Quality **Weight 17.00%**	**100 %** Code is written clearly and is well commented	**85 %** Code readability is less and/or commenting is simple	**75 %** The code is readable only by someone who knows what it is supposed to be doing. Commenting is severely lacking	**65 %** Code is using single letter variables, poorly organized and bare minimum comments	**35 %** The code is poorly organized and very difficult to read. Comments are poor	**20 %** Code uses excessive single letter identifiers. Excessively poorly organized.	**0 %** Code is incomprehensible
Efficiency **Weight 16.00%**	**100 %** The code is extremely efficient without sacrificing readability and understanding.	**85 %** The code is fairly efficient without sacrificing readability and understanding.	**75 %** The code is brute force but concise.	**65 %** The code is brute force and unnecessarily long.	**35 %** The code is huge and appears to be patched together.	**20 %** The code has created very poor runtimes for much simpler faster algorithms.	**0 %** Code is incomprehensible

What to Submit?

This homework assignment will have multiple parts. You are required to submit your solutions in a compressed format (.zip). Make sure your compressed file is label correctly - lastname_firstname2.zip. (All lowercase, do not put anything else in the name like "hw2".)

The compressed file MUST contain the following:

- hw02q1.txt
- hw02q2.c
- hw02q2.pdf
- hw02q3.c

No other files should be in the compressed folder. A .txt file is a plain text file, which is produced by tools like Notepad. Microsoft Word does not save plain text by default.

If multiple submissions are made, the most recent submission will be graded. (Even if the assignment is submitted late.)

Where to Submit?

All submissions must be electronically submitted to the respected homework link in the course web page where you downloaded the assignment.