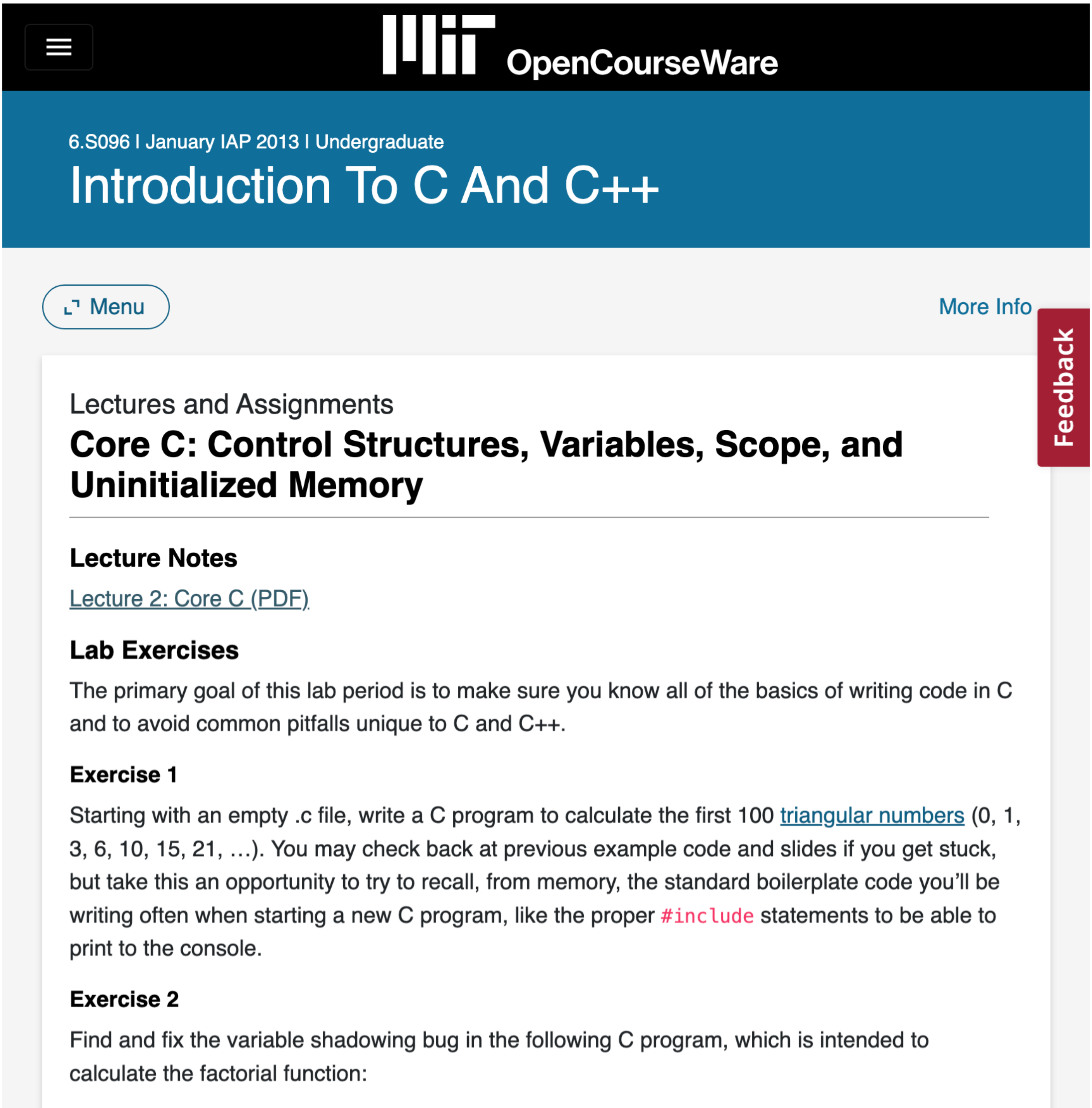

6.S096 | January IAP 2013 | Undergraduate

Introduction To C And C++

Lectures and Assignments

Core C: Control Structures, Variables, Scope, and Uninitialized Memory

Lecture Notes

Lecture 2: Core C (PDF)

Lab Exercises

The primary goal of this lab period is to make sure you know all of the basics of writing code in C and to avoid common pitfalls unique to C and C++.

Exercise 1

Starting with an empty .c file, write a C program to calculate the first 100 triangular numbers (0, 1, 3, 6, 10, 15, 21, …). You may check back at previous example code and slides if you get stuck, but take this an opportunity to try to recall, from memory, the standard boilerplate code you'll be writing often when starting a new C program, like the proper `#include` statements to be able to print to the console.

Exercise 2

Find and fix the variable shadowing bug in the following C program, which is intended to calculate the factorial function:

```
#include <stdio.h>

int factorial (int n) {
        int i = 1;
        while (n > 1) {
                i = i * n;
                int n = n - 1;
        }
        return i;
}

int main (int argc, char *argv[]) {
        int fac4 = factorial(4);
        int fac5 = factorial(5);
        printf("4! = %d, 5! = %d\n", fac4, fac5);
        return 0;
}
```

Assignment 2

Problem 1

Rewrite the program below, replacing the `for` loop with a combination of `goto` and `if` statements. The output of the program should stay the same. That is, the program should print out the arguments passed to it on the command line.

```
#include <stdio.h>

int main(int argc, char ** argv){
    for (int i = 1; i < argc; i++) {
        printf("%s\n", argv[i]);
    }
    return 0;
}
```

If you ran the program like this: `./prog one two three`, it would print

```
one
two
three
```

Problem 2

In lecture, we covered a variety of control structures for looping, such as `for`, `while`, `do/while`, `goto`, and several for testing conditions, such as `if` and `switch`.

Your task is to find seven different ways to print the odd numbers between 0 and 10 on a single line. You should give them names like `amaze1`, `amaze2`, etc. Here's a bonus amaze (that you can use as one of your seven if you like):

```
void amazeWOW() {
        int i;
        printf("amazeWOW:\t");
        for (i = 0; i <= 10; i++) {
                if (i % 2 == 1) {
                        printf("%d ", i);
                }
        }
        printf("\n");
}
```

You may need to get a little creative!

Put all of your functions in the same C file and call them in order from the `main()` function. We should be able to compile your program with this command and see no errors:

```
gcc -Wall -std=c99 -o amaze amaze.c
```

Solutions

Solutions are not available for this assignment.