Computer Programming, M 2022
Lab 1

Objective: To be able to write simple C programs

Part A: Practice

1. A first program in C

```c
/* Fig. 2.1: fig02_01.c
   A first program in C */
#include <stdio.h>

/* function main begins program execution */
int main( void )
{
   printf( "Welcome to C!\n" );

   return 0; /* indicate that program ended successfully */
} /* end function main */
```

```
Welcome to C!
```

Commands to compile & run

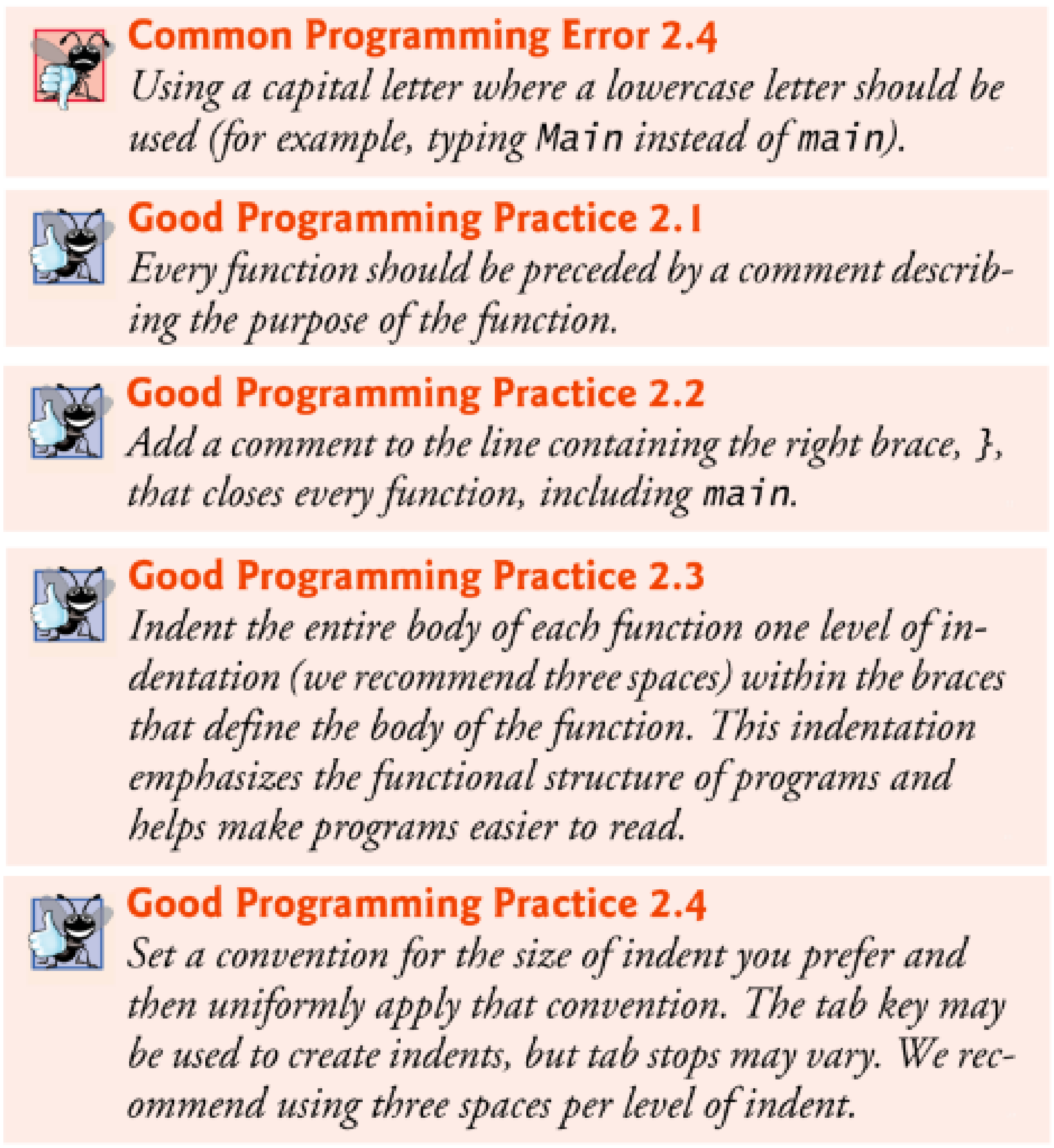
Copy and paste the above code in a file named ***myfile.c.*** To compile and run your code, use the following commands:

```
$ gcc myfile.c            # to compile
$ ./a.out                 # to run
```

Common Programming Error 2.1
*Forgetting to terminate a comment with */.*

`

Common Programming Error 2.3
Typing the name of the output function `printf` as `print` in a program.



Common Programming Error 2.4

Using a capital letter where a lowercase letter should be used (for example, typing `Main` instead of `main`).

Good Programming Practice 2.1

Every function should be preceded by a comment describing the purpose of the function.

Good Programming Practice 2.2

Add a comment to the line containing the right brace, }, that closes every function, including `main`.

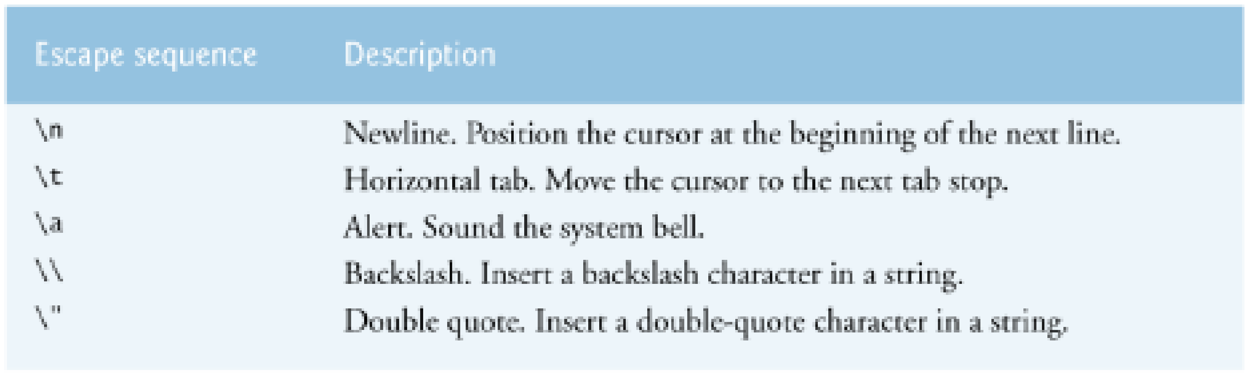
Good Programming Practice 2.3

Indent the entire body of each function one level of indentation (we recommend three spaces) within the braces that define the body of the function. This indentation emphasizes the functional structure of programs and helps make programs easier to read.

Good Programming Practice 2.4

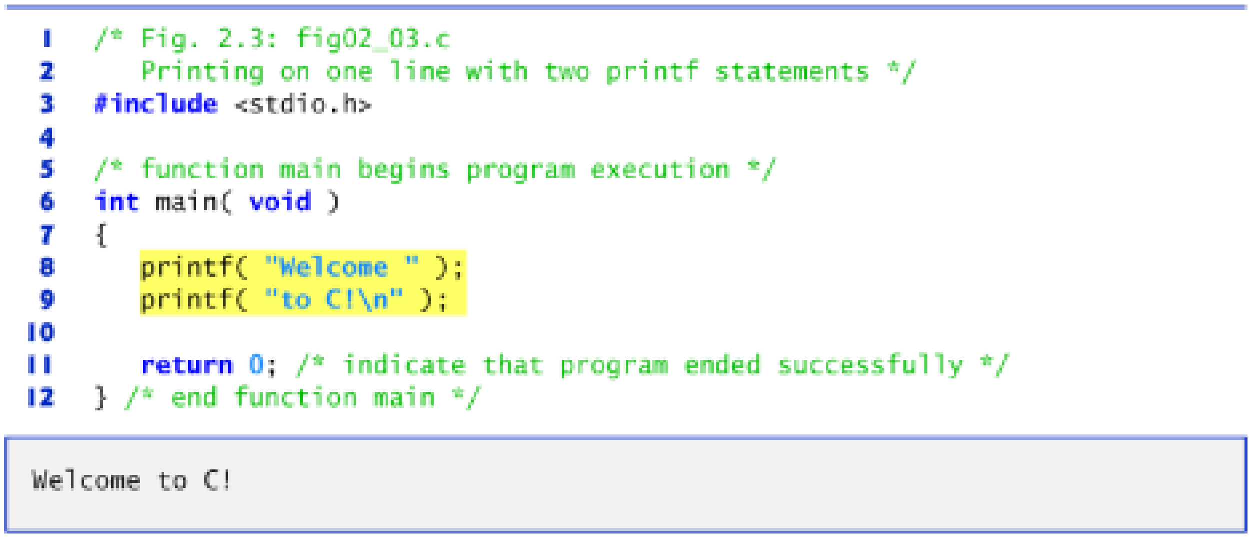
Set a convention for the size of indent you prefer and then uniformly apply that convention. The tab key may be used to create indents, but tab stops may vary. We recommend using three spaces per level of indent.

Some common escape sequences:



Escape sequence	Description
\n	Newline. Position the cursor at the beginning of the next line.
\t	Horizontal tab. Move the cursor to the next tab stop.
\a	Alert. Sound the system bell.
\\	Backslash. Insert a backslash character in a string.
\"	Double quote. Insert a double-quote character in a string.

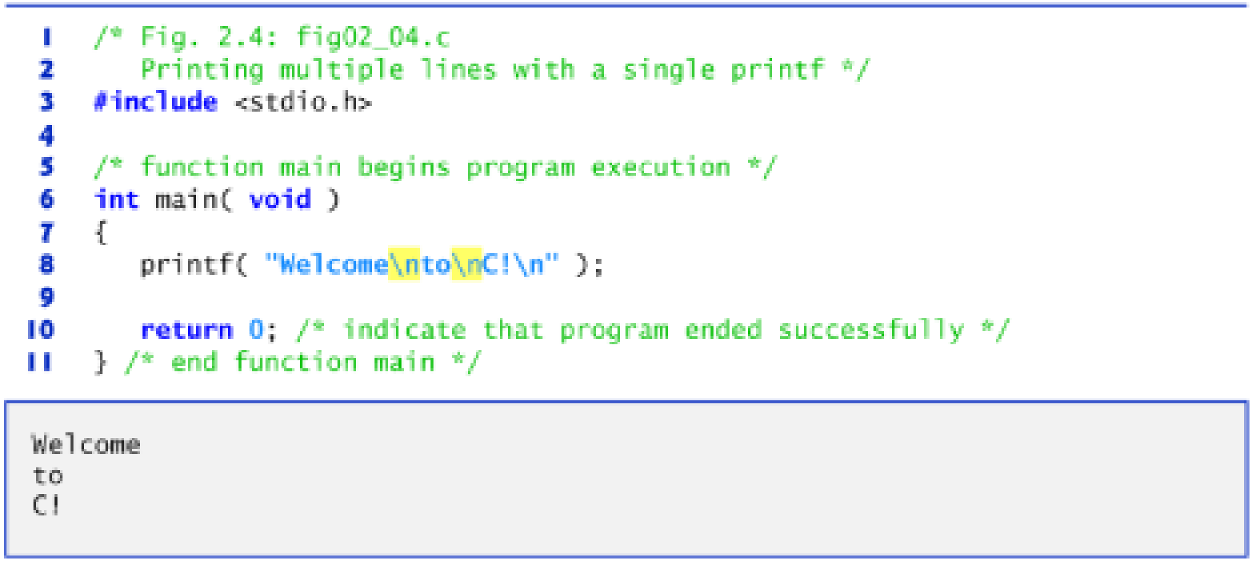
2. Printing on one line with two **printf** statement.



```c
/* Fig. 2.3: fig02_03.c
   Printing on one line with two printf statements */
#include <stdio.h>

/* function main begins program execution */
int main( void )
{
   printf( "Welcome " );
   printf( "to C!\n" );

   return 0; /* indicate that program ended successfully */
} /* end function main */
```

```
Welcome to C!
```

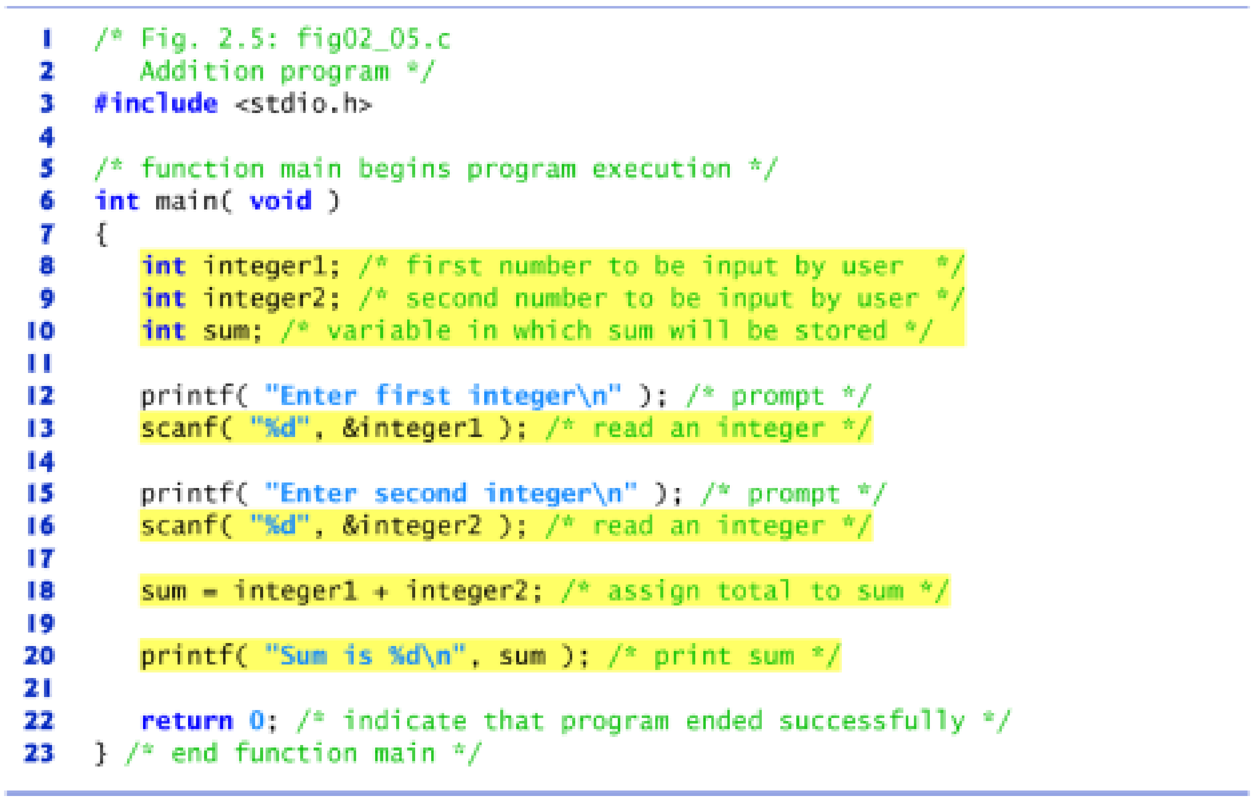
3. Printing multiple lines with single **printf**.



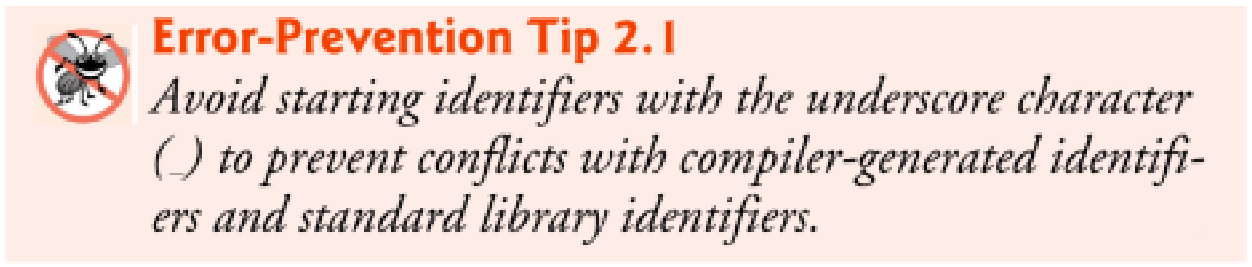
```c
/* Fig. 2.4: fig02_04.c
   Printing multiple lines with a single printf */
#include <stdio.h>

/* function main begins program execution */
int main( void )
{
   printf( "Welcome\nto\nC!\n" );

   return 0; /* indicate that program ended successfully */
} /* end function main */
```

```
Welcome
to
C!
```

4. Addition Program



```c
/* Fig. 2.5: fig02_05.c
   Addition program */
#include <stdio.h>

/* function main begins program execution */
int main( void )
{
   int integer1; /* first number to be input by user  */
   int integer2; /* second number to be input by user */
   int sum; /* variable in which sum will be stored */

   printf( "Enter first integer\n" ); /* prompt */
   scanf( "%d", &integer1 ); /* read an integer */

   printf( "Enter second integer\n" ); /* prompt */
   scanf( "%d", &integer2 ); /* read an integer */

   sum = integer1 + integer2; /* assign total to sum */

   printf( "Sum is %d\n", sum ); /* print sum */

   return 0; /* indicate that program ended successfully */
} /* end function main */
```



Error-Prevention Tip 2.1

Avoid starting identifiers with the underscore character (_) to prevent conflicts with compiler-generated identifiers and standard library identifiers.

Part B: Exercise

1. Write a program to print your details in the format given below:

   Name: XYZ

   Roll No. xxxxxxxxxx

   Branch: Computer Science and Engineering (or ECE)

   You should not take any input from the user. To get the output above, just use 4 printf statements.

2. Write a program to perform the following operations on two numbers:
    a. Addition
    b. Subtraction
    c. Multiplication
    d. Division
    e. modulus

   Your program should prompt the user to enter the two numbers. (Hint: use scanf)

3. Write a program to calculate the hypotenuse of a right angled triangle, given its base and height. Your program should prompt the user to enter the base and the height of the triangle.
   (Note: for this you may have to use the math.h library, Tha math library has the *'sqrt'* and *'pow'* functions, which will help you. Click on the below link for learning how to use the math library)
   https://www.includehelp.com/c-programming-questions/compiling-program-with-math-library-linux.aspx#:~:text=h%20library%3F-,To%20compile%20C%20program%20with%20math.,h%20library.

4. Write a program to print the sum of the first **100** terms of the given arithmetic progression.

   Let the first two numbers of the sequence are ***a*** and ***b***, say the difference between ***a*** and ***b*** is ***d.*** (in other words $d = b - a$)
   Then the sum of the first 100 terms:

   $$\mathbf{S = [100\ (2a + 99d)]/2}$$

   Your program should prompt the user to input a and b. Then in return your program should print the sum of the first 100 terms of the sequence.
   [Hint: Find the difference d, and just use the formula, it's that simple.]

[Each question carries 5 Marks]

****Read the below instructions very carefully****

What to turn in

1. **Four c-program** files with the extension .C (C program files). Please name the files as follows
   a. Program1_2022xxxx.c
   b. Program2_2022xxxx.c
   c. Program3_2022xxxx.c
   d. Program4_2022xxxx.c
2. You also need to submit **Four text files** with the extension .txt (text files showing the output of your programs). Name the files as follows:
   a. Output1_2022xxxx.txt
   b. Output2_2022xxxx.txt
   c. Output3_2022xxxx.txt
   d. Output4_2022xxxx.txt
   (Note: You can easily create the above files by copy pasting text from your command line into the text files)
3. Don't forget to replace the **'2022xxxx'** in the name of the files with your full roll number.
4. Submit the above via Google Classroom classwork. **DO NOT** submit as a message in the stream. If you don't know how to submit an assignment in google classroom, ask your teacher. He/She will show you an example.
5. **DO NOT ZIP** or create any other compressed file. You should submit the files directly as .C and .TXT files. If you submit with any other extension, your assignment will not be graded. When you submit successfully, there should be 8 files in your submission.

Warning: Missing to follow the instructions will result in reduction of points/marks.