ENGG 1410 Assignment 1

General Questions

1. Problem Solving Questions:
    - Steps required to solve programming problem
        - Understanding requirements
        - Flow chart to understand program flow
        - Write Pseudo-Code from flow chart
        - Writing code in C from Pseudo-code
    - Flow chart, Pseudo-code, diagrams
    - A flow chart is a graphical representation of the state of a program, pseudo code is a language agnostic way of representing code
2. Programming Language Questions:
    - Source code is a text file written in a programming language, Object code is a pre-linked file which contains compiled code and symbols awaiting to be linked to. Executable code is a binary ready to be run on a system
    - Types, input, output, data manipulation
    - High level, low level, machine code, assembly
    - Easier to understand, faster development
3. C Language Questions:
    - 1972
    - Designed in close relation to the Unix Operating System for use when building applications for it
    - As a established and long used (relatively) low level programming language, it is useful to learn CS fundamentals
    - The pre-processor manipulates the file (only text operations) and then passes the file to the compiler, which changes the text code into object code which can be understood by the linker.
    - The #include <stdio.h> is for including standard input and output functions and definitions
    - Comments are useful when describing esoteric or hard to understand code
    - The semicolon is used to mark the end of a statement in C
4. Steps for getting hello world running:
    - Download & install vscode and visual studio community edition
    - Run vscode from visual studio command prompt

- Open folder and create file named main.c
- Write the "Hello World!" program into the main.c file
- Click "Run C/C++ File" and select cl.exe
- File is compiled and run.

5. File will not compile due to undefined symbol printf & undefined symbol EXIT_SUCCESS. Moving line 8 to line 1 will get this program to compile
6. Program does not compile, due to a redefinition with different basic types.

Syntax Qustions

7. To include printf/scanf and other standard C input and output functions.
8. - a) long int - at least 32-bit (on modern Windows is 32-bit, on modern Unix is 64-bit) short int - at least 16-bit (on modern systems is 16-bit)
   int - at least 16-bit (on modern systems is 32-bit)
   - b) double - double precision floating point (64-bit)
   float - single precision floating point (32-bit)
9. A token that is used & reserved by the c compiler for specific uses: types, specifiers, etc.
10. There can't be half a student, so it must be an int as so:

```c
int students = 30;
```

11. Ex Program:

```c
#include <stdio.h>

int main(void) {
    printf("    * In C, lowercase letters are significant\n");
    printf("    * main( ) is where program execution begins\n");
    printf("    * main( ) is different than Main ( ) , the latter actually will not serve the purpose\n");
    printf("    * All C program statements must be terminated by a semicolon\n");
    return 0;
}
```

12.

```c
int cvg1234; // Valid
float double; // Invalid (identifiers cannot be reserved keywords)
char 1letter; // Invalid (identifiers cannot start with a number)
long hello$world; // Technically invalid by standard, but commonly supported as a compiler extension
int x_Y_4; // Valid
```

13.

```c
int Double; // Valid
int Main; // Valid
int 3Cats; // Invalid (identifiers cannot start with a number)
int he_llo; // Valid
int num 4; // Invalid (identifiers cannot have a space inside it)
int E2E; // Valid
```

I/O

14.

```c
#include <stdio.h>

int main(void) {
    printf("To C or Not to C\n");
    printf("                That is the question!\n");
    return 0;
}
```

15.

```c
#include <stdio.h>

int main(void) {
    printf("a- Start Game\n");
    printf("b- Load Game\n");
    printf("c- Save Game\n");
```

```
    printf("d- Exit\n");
    printf("Please enter your choice: ");
    return 0;
}
```

16.

```
#include <stdio.h>

int main(void) {
    printf("87 - 14 = %d\n", 87-14);
    return 0;
}
```

17.

```
int main(Void) // Should be: int main(void)
{
    INT sum; // Should be: int sum;
    /* compt Result // Should be: /* Compt Result */
    sum = 25 + 37 -1 // Should be sum = 25 + 37 -1;

    /* Display Results // // Should be: /* Display Results */
    printf("The answer is %i \n"        sum); // Should be: printf("The answer is %i \n", sum);

    return 0;
}
```

18.

```
#include <stdio.h>

int main(void) {
    int a = 0;
    int b = 0;
```

```
    printf("Input integer a: ");
    scanf("%i", &a);
    printf("Input integer b: ");
    scanf("%i", &b);
    printf("The sum is %i\n", a + b);
    return 0;
}
```

19.

```
#include <stdio.h>

int main(void) {
    for(int i = 1; i <= 10; i++) {
        printf("%d - %d\n", i, i*i);
    }
    return 0;
}
```

20. There can't be half a student, so it must be an int as so:

```
int students = 30;
```

21.

```
#include <stdio.h>
#define PI 3.1415f
int main(void) {
    float r = 0;
    printf("Input radius of circle: ");
    scanf("%f", &r);
    printf("Area of circle: %fu^2\n", PI * r * r);
    printf("Circumference of circle: %fu\n", 2 * PI * r);
    return 0;
}
```

22.

```c
#include <stdio.h>

int main(void) {
    float l = 0;
    float w = 0;
    printf("Input length of rectangle: ");
    scanf("%f", &l);
    printf("Input width of rectangle: ");
    scanf("%f", &w);
    printf("Area of rectangle: %fu^2\n", l * w);
    printf("Perimeter of rectangle: %fu\n", 2 * (l + w));
    return 0;
}
```

23.

```c
#include <stdio.h>

int main(void) {
    float d = 0;
    printf("Input distance in meters: ");
    scanf("%f", &d);
    printf("Distance in centimeters: %fcm\n", d * 100);
    printf("Distance in millimeters: %fmm\n", d * 100000);
    return 0;
}
```

24.

```c
#include <stdio.h>

int main(void) {
    printf("a-\n");
```

```
    printf("*********\n");
    printf("*  CoE  *\n");
    printf("*********\n");
    printf("\n");
    printf("b-\n");
    printf("***********\n");
    printf("***********\n");
    printf("**  CoE  **\n");
    printf("***********\n");
    printf("***********\n");
    return 0;
}
```