Lecture 1 examples

October 7, 2019

1 Our first Program in C

We will develop and run some simple C programs to illustrate some basic elements. Most probably this is not going to be fully understandable. But for now accet it as it is.

Our first program is a famous "Hello world!" program. That is a program that just prints something and terminates:

```
In [1]: #include <stdio.h>

        int main()
        {
            printf("Hello students!!\n");
        }
```

```
Hello students!!
```

This program, while the simplest it can be already contains elements that are possibly new and will need explenation.

1.1 The structure of our program:

1.1.1 Preprocessor directives

Everything that starts with a # is a preprocessor directive. This will gouvern includes defines and other statments. For now please accept we need to have at least:
#include <stdio.h> it is responsible for providing basic input/output operations

```
In [ ]: #include <stdio.h> // need to have it
```

Than our program reguieres an entry point. This is provided by a special function. it is called **main**. We have not said anything about functions, so for now we will accept it needs to be there. This function os of type **int** (whatever that means):

```
In [2]: #include <stdio.h> // need to have it

        int main() //our main function is an entry point
        {
            // This here is the body of our main function. This is where instruction defining
        }
```

We will consider a differnt (there is much more of them) preprocessor directive. The **define**. It is uded to predefine values that will be used as constants.

```
In [ ]: #define PI 3.141592
```

This requests the preprocessor to replace all occurances of PI with 3.141592. It should be used with care, as it can lead to some unexpected results if important vaues become replaced. Consider the following:

```
In [5]: #include <stdio.h> // need to have it
        #define main da

        int main()
        {
            printf("Hello CS1 students!!\n");
        }
```

```
/tmp/tmpf_z791f8.out: /tmp/tmpqpouyi2a.out: undefined symbol: main
[C kernel] Executable exited with code 1
```

It fails since main has been replaced with da, and the acctual code now loos like this:

```
In [ ]: #include <stdio.h> // need to have it
        #define main da

        int da()
        {
            printf("Hello CS1 students!!\n");
        }
```

Which makes no sense.

1.1.2 Functions

We can add functions to our program. Consider a fution that will work with to variables of type integer (int - we will cover this...)

```
In [6]: #include <stdio.h> // need to have it
        #define PI 3.141592

        int main()
        {
            printf("Hello CS1 students!!\n");
            sum(5,6);
        }
```

```
/tmp/tmpdblz0set.c: In function main:
/tmp/tmpdblz0set.c:7:5: warning: implicit declaration of function sum [-Wimplicit-function-decl
     sum(5,6);
     ^~~
/tmp/tmpf_z791f8.out: symbol lookup error: /tmp/tmp04x_ze9f.out: undefined symbol: sum
[C kernel] Executable exited with code 127
```

We try to use function sum() which has not been defined. We get a compilation error in line 7, collumn 5 that we try to use an undefined function. Have a look at the error message. Most probably you will be seeing it often.

We need to add a definition of a function sum, e.g. as below:

```
In [7]: #include <stdio.h> // need to have it
        #define PI 3.141592

        int sum(int a, int b) // definition before use
        {
            printf("a= b= a+b= \n");
        }

        int main()
        {
            printf("Hello CS1 students!!\n");
            sum(5,6);
        }
```

```
Hello CS1 students!!
a= b= a+b=
```

It is not important what function sum() does for now. We define it and we can call it. That is what matters for now. This is not the only way to deal with a function. Try the following:

```
In [8]: #include <stdio.h> // need to have it
        #define PI 3.141592

        int sum(int a, int b); // this is a promise to the compiler

        int main()
        {
            printf("Hello CS1 students!!\n");
            sum(5,6);
        }
```

```
Hello CS1 students!!


/tmp/tmpf_z791f8.out: symbol lookup error: /tmp/tmpzn7sl4j5.out: undefined symbol: sum
[C kernel] Executable exited with code 127
```

This time we declared a function to be used (in line 4) but we did not provide the body of the function. Note that the error os different than before! This time compilation failed during linking as appropriate function could not be located for lonking.

Finally, we have declaration and definition in the following:

```
In [9]: #include <stdio.h> // need to have it
        #define PI 3.141592

        int sum(int a, int b); // this is a promise to the compiler

        int main()
        {
            printf("Hello CS1 students!!\n");
            sum(5,6);
        }

        int sum(int a, int b) // definition before use
        {
            printf("a=%d b=%d a+b=%d \n", a, b, a+b); // This needs a special formatting
        }
```

```
Hello CS1 students!!
a=5 b=6 a+b=11
```

We will now comment that function sum() doeas some printing. You might note that in the printing request some new elements are presant, namly the output formating. We use %d to print integers. We will discuss it in more detail in future.

This concludes our first round of examples.