### **Lecture 4: Calling printf & Including stdio.h**

**printf** is a predefined function; call printf whenever something needs to be printed on the monitor.

**How to use**:

* printf("... text here");
* To print any value: printf("%d", z); %d is a format specifier, and z is the value.

**What is a format specifier?**

* A format specifier prints values as per the specified format.
  + %d for int
  + %f for float
  + %c for characters

**Example Code**:

c

Copy code

int main() {

int x, y, z;

x = 10;

y = 10;

z = x + y;

printf("%d", z);

return 0;

}

**Some points about the above code**:

* **Which language?** English
* **Which programming language?** C

The operating system cannot execute eg1.c because the OS does not understand English; it only knows machine language. So, we need something that knows both English and machine language, called a **Compiler**. Here, the compiler should know about the C language, so it should be a **C Compiler**.

For every language, there is a separate compiler.

**Source Code**:

* eg1.c is the source code.
* We will give the source code to the compiler. If the code is correct, it will produce **compiled code**; if incorrect, it will print an error message.

**Source Code -> C Compiler -> if correct compile -> produce compiled code file**

We will give that compiled code file to the OS to execute.

Here, the compiler knows nothing about the printf function.

**What does the compiler know?**

* Keywords
* Operators
* Reserved symbols

The compiler does not know about the printf function, so we have to teach the compiler about the printf function.

**What do we need to teach the compiler?**

1. The syntax to place a call to the printf function.
2. How printf works (not needed in this case).

**Header File**:

* The header file contains the prototype of functions.

Those who created C have made lots of functions like printf, scanf, cos, sin, strlen, strcpy, and so on, and then they categorized them:

* Input/Output: printf, scanf
* Math: cos, sin
* String: strlen, strcpy

**Header Files**:

* Input/Output: stdio.h
* Math: math.h
* String: string.h

Only the prototype of printf is present in stdio.h, not the entire definition.

**Note**: stdio.h contains the prototype of the functions related to standard input and output devices. The library file contains the definition of printf.

As of now, the above code will not compile because the code does not know anything about printf. We have to use stdio.h.

**Code with stdio.h**:

c

Copy code

#include <stdio.h>

int main() {

int x, y, z;

x = 10;

y = 20;

z = x + y;

printf("%d", z);

return 0;

}

**Explanation**:

* #include <stdio.h>: The hash symbol (#) indicates a preprocessor directive. < and > are less than and greater than symbols, not angular brackets.

Whatever we write with # is called a **preprocessor directive**, like #include, #define, #ifndef.

During compilation, it will start line by line. When the compilation reaches #include <stdio.h>, it will open the header file and load all prototypes. Then, it will move to int main(), and then to the printf line. It will check the syntax, and if all is good, it will continue; otherwise, it will show an error. Then the compiler will create the executable file.