C Preprocessor

- The **C Preprocessor** is not a part of the compiler, but is a separate step in the compilation process. In simple terms, a C Preprocessor is just a text substitution tool and it instructs the compiler to do required pre-processing before the actual compilation.
- The **C preprocessor** is a *macro processor* that is used automatically by the C compiler to transform your program before actual compilation. It is called a macro processor because it allows you to define *macros*, which are brief abbreviations for longer constructs.

C Preprocessor

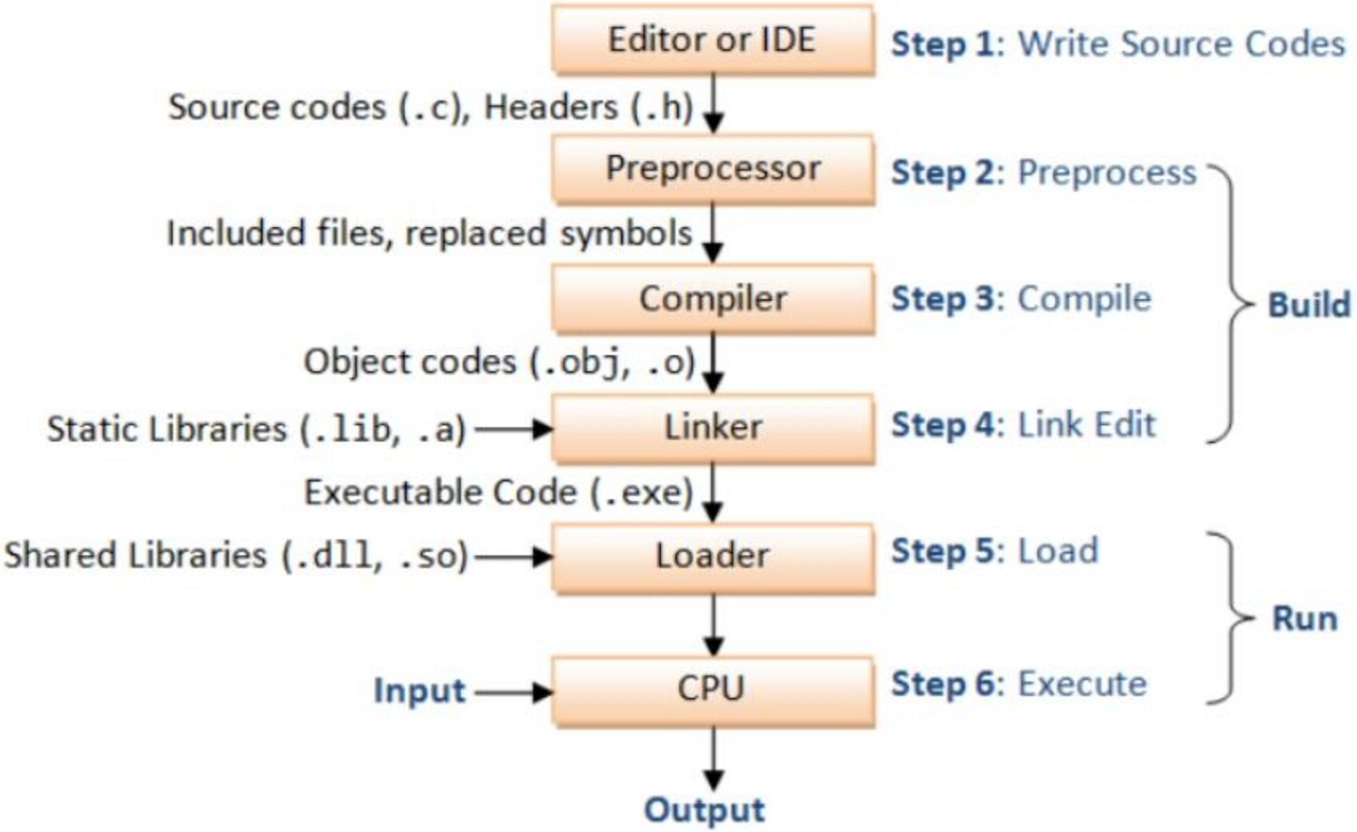
The C preprocessor provides 4 types as follows:

- Inclusion of header files - These are files of declarations that can be substituted into your program.
- Macro expansion - You can define *macros*, which are abbreviations for arbitrary fragments of C code, and then the C preprocessor will replace the macros with their definitions throughout the program.
- Conditional compilation - Using special preprocessing directives, you can include or exclude parts of the program according to various conditions.
- Line control - If you use a program to combine or rearrange source files into an intermediate file which is then compiled, you can use line control to inform the compiler of where each source line originally came from.

C Preprocessor

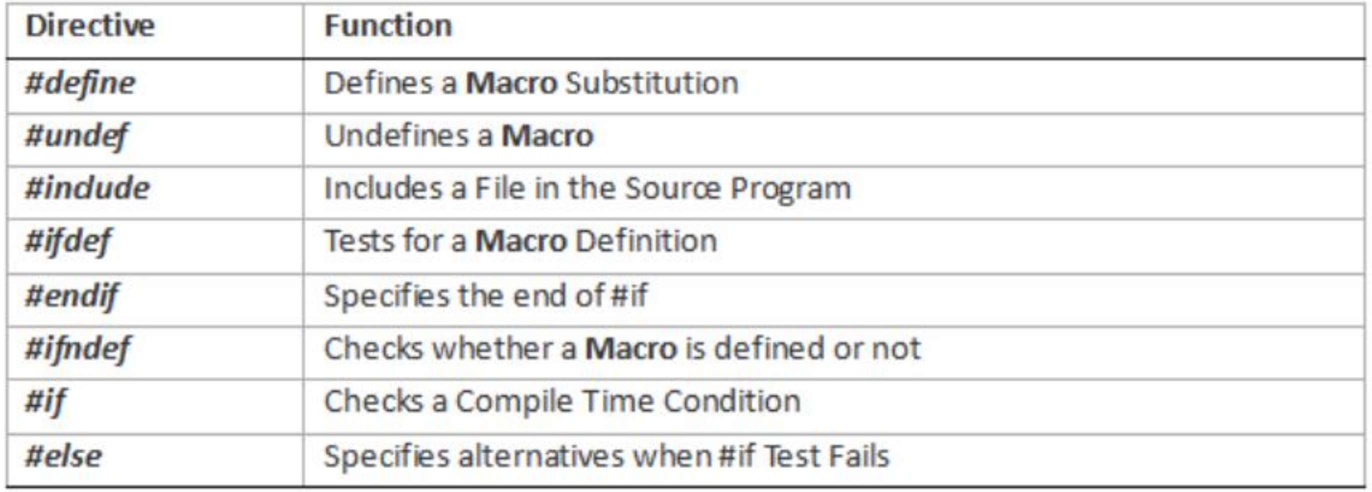
Q. How the C Preprocessor works?

➢



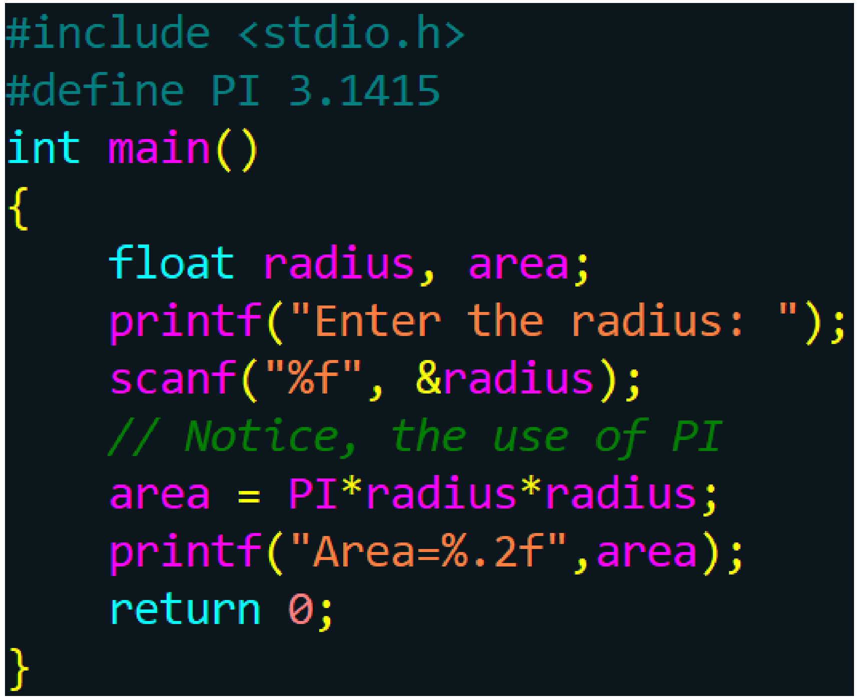
C Preprocessor

- Following are the different preprocessor directives:



Directive	Function
#define	Defines a **Macro** Substitution
#undef	Undefines a **Macro**
#indude	Includes a File in the Source Program
#ifdef	Tests for a **Macro** Definition
#endif	Specifies the end of #if
#ifndef	Checks whether a **Macro** is defined or not
#if	Checks a Compile Time Condition
#else	Specifies alternatives when #if Test Fails

Example.



```c
#include <stdio.h>
#define PI 3.1415
int main()
{
    float radius, area;
    printf("Enter the radius: ");
    scanf("%f", &radius);
    // Notice, the use of PI
    area = PI*radius*radius;
    printf("Area=%.2f",area);
    return 0;
}
```