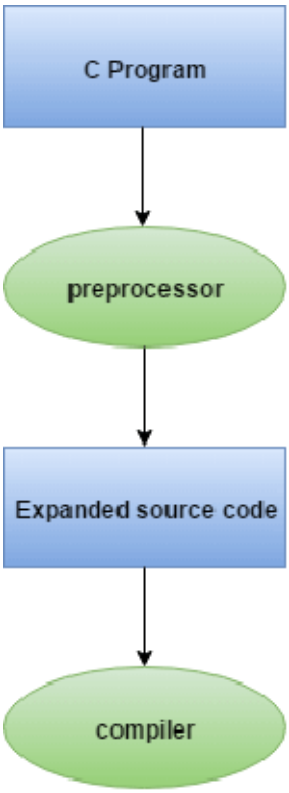
C Preprocessor Directives

The C preprocessor is a micro processor that is used by compiler to transform your code before compilation. It is called micro preprocessor because it allows us to add macros.

> Note: Proprocessor direcives are executed before compilation.



All preprocessor directives starts with hash # symbol.

Let's see a list of preprocessor directives.

- #include
- #define
- #undef
- #ifdef
- #ifndef
- #if
- #else

- #elif
- #endif
- #error
- #pragma

What is Macro

C Macros

A macro is a segment of code which is replaced by the value of macro. Macro is defined by #define directive. There are two types of macros:

1. Object-like Macros
2. Function-like Macros

Object-like Macros

The object-like macro is an identifier that is replaced by value. It is widely used to represent numeric constants. For example:

```
#define PI 3.14
```

Here, PI is the macro name which will be replaced by the value 3.14.

Function-like Macros

The function-like macro looks like function call. For example:

```
#define MIN(a,b) ((a)<(b)?(a):(b))
```

Here, MIN is the macro name.

Visit #define to see the full example of object-like and function-like macros.

C Predefined Macros

ANSI C defines many predefined macros that can be used in c program.

No.	Macro	Description
1	_DATE_	represents current date in "MMM DD YYYY" format.
2	_TIME_	represents current time in "HH:MM:SS" format.
3	_FILE_	represents current file name.
4	_LINE_	represents current line number.
5	_STDC_	It is defined as 1 when compiler complies with the ANSI standard.

C predefined macros example

File: simple.c

```
1. #include<stdio.h>
2.  int main(){
3.    printf("File :%s\n", __FILE__ );
4.    printf("Date :%s\n", __DATE__ );
5.    printf("Time :%s\n", __TIME__ );
6.    printf("Line :%d\n", __LINE__ );
7.    printf("STDC :%d\n", __STDC__ );
8.    return 0;
9.  }
```

Output:

```
File :simple.c
Date :Dec 6 2015
Time :12:28:46
Line :6
STDC :1
```