Comments in C

- Comments in C language are used to provide information about lines of code. It is widely used for documenting code. There are 2 types of comments in the C language.

Single Line Comments

- Single line comments are represented by double slash \\. Let's see an example of a single line comment in C.

```
#include<stdio.h>
int main()
{
        //printing information
        printf("Hello C");
        return 0;
}
```

- **Mult Line Comments**
- Multi-Line comments are represented by slash asterisk /* ... */. It can occupy many lines of code, but it can't be nested. Syntax:

```
/*
code
to be commented
*/
```

Example:

```
#include<stdio.h>
int main()
{
   /*printing information
     Multi-Line Comment*/
   printf("Hello C");
   return 0;
}
```

C Format Specifier

- The Format specifier is a string used in the formatted input and output functions. The format string determines the format of the input and output. The format string always starts with a '%' character.
- **The commonly used format specifiers in printf() function are:**

Format specifier	Description
%d or %i	It is used to print the signed integer value where signed integer means that the variable can hold both positive and negative values.
%u	It is used to print the unsigned integer value where the unsigned integer means that the variable can hold only positive value.
%o	It is used to print the octal unsigned integer where octal integer value always starts with a 0 value.
%x	It is used to print the hexadecimal unsigned integer where the hexadecimal integer value always starts with a 0x value. In this, alphabetical characters are printed in small letters such as a, b, c, etc.

%X	It is used to print the hexadecimal unsigned integer, but %X prints the alphabetical characters in uppercase such as A, B, C, etc.
%f	It is used for printing the decimal floating-point values. By default, it prints the 6 values after '.'.
%e/%E	It is used for scientific notation. It is also known as Mantissa or Exponent.
%g	It is used to print the decimal floating-point values, and it uses the fixed precision, i.e., the value after the decimal in input would be exactly the same as the value in the output.
%p	It is used to print the address in a hexadecimal form.
%c	It is used to print the unsigned character.
%s	It is used to print the strings.
%ld	It is used to print the long-signed integer value.

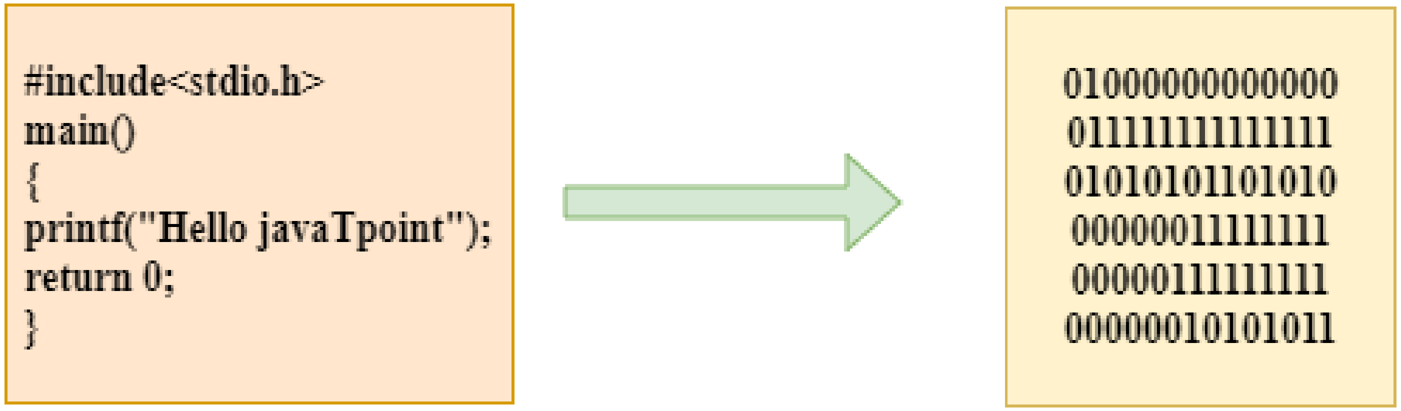
Example

```
int main()
{
 int b=6;
  int c=8;
  printf("Value of b is:%d", b);
  printf("\nValue of c is:%d",c);
  return 0;
}
```

In the above code, we are printing the integer value of b and c by using the %d specifier.

Compilation Process in C

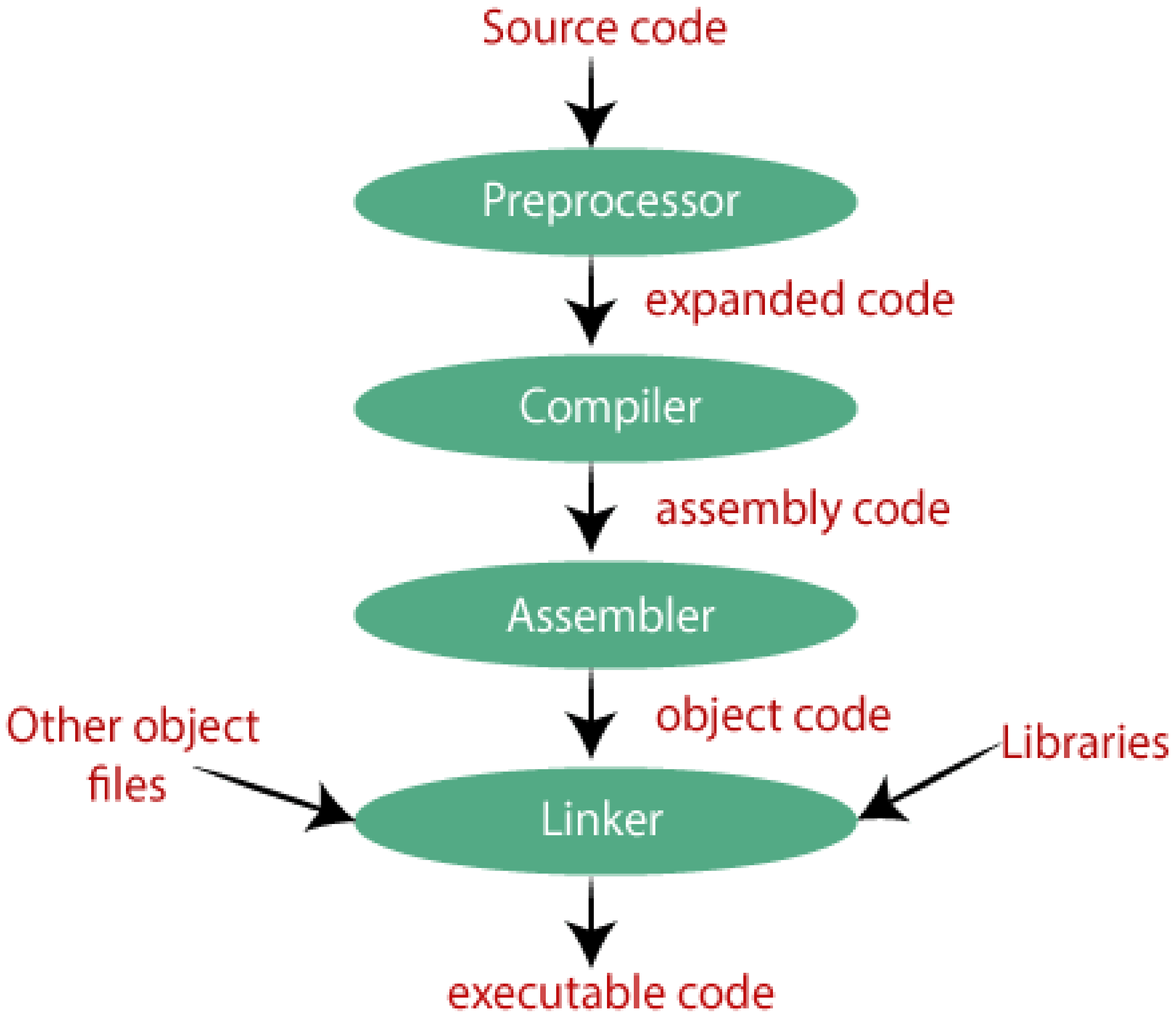
- The compilation is a process of converting the source code into object code. It is done with the help of the compiler.
- The compiler checks the source code for the syntactical or structural errors, and if the source code is error-free, then it generates the object code.
- The c compilation process converts the source code taken as input into the object code or machine code.



```
#include<stdio.h>
main()
{
printf("Hello javaTpoint");
return 0;
}
```

```
01000000000000
01111111111111
01010101101010
00000011111111
00000111111111
00000010101011
```

- The compilation process can be divided into four steps, i.e., Pre-processing, Compiling, Assembling, and Linking


Source code
Preprocessor
expanded code
Compiler
assembly code
Assembler
object code
Other object files
Libraries
Linker
executable code

Preprocessor

- The preprocessor takes the source code as an input, and it removes all the comments from the source code.
- The preprocessor takes the preprocessor directive and interprets it. For example, if **<stdio.h>,** the directive is available in the program, then the preprocessor interprets the directive and replace this directive with the content of the **'stdio.h'** file.
- The source code is the code which is written in a text editor and the source code file is given an extension ".c".
- This source code is first passed to the preprocessor, and then the preprocessor expands this code. After expanding the code, the expanded code is passed to the compiler.

Compiler

- The code which is expanded by the preprocessor is passed to the compiler. The compiler converts this code into assembly code. Or we can say that the C compiler converts the pre-processed code into assembly code.

Assembler

- The assembly code is converted into object code by using an assembler
- The name of the object file generated by the assembler is the same as the source file.
- The extension of the object file in DOS is '.obj,' and in UNIX, the extension is 'o'.
- If the name of the source file is **'hello.c',** then the name of the object file would be '**hello.obj**'.

Linker

- Mainly, all the programs written in C use library functions. The main working of the linker is to combine the object code of library files with the object code of our program.
- The output of the linker is the executable file. The name of the executable file is the same as the source file but differs only in their extensions.
- In DOS, the extension of the executable file is '.exe', and in UNIX, the executable file can be named as 'a.out'.