Macros in C programming

In a C program, all lines that start with # are processed by preprocessor which is a special program invoked by the compiler. In a very basic term, preprocessor takes a C program and produces another C program without any #.

1) When we use ***include*** directive, the contents of included header file (after preprocessing) are copied to the current file.
Angular brackets **<** and **>** instruct the preprocessor to look in the standard folder where all header files are held. Double quotes **"** and **"** instruct the preprocessor to look into the current folder (current directory).

2) When we use ***define*** for a constant, the preprocessor produces a C program where the defined constant is searched and matching tokens are replaced with the given expression. For example in the following program *max* is defined as 100.

Example

```
#include<stdio.h>
#define val 10
int main()
{
    printf("value is %d", val);
    return 0;
```

The macros can take function like arguments, the arguments are not checked for data type.

```
#include <stdio.h>
#define INCREMENT(x) ++x
int main()
{
    char *ptr = "CSEbatch";
    int x = 10;
    printf("%s  ", INCREMENT(ptr));
    printf("%d", INCREMENT(x));
    return 0;
}
```

Output

```
SEbatch 11
```

The macro arguments are not evaluated before macro expansion. For example, consider the following program.

```
#include <stdio.h>
#define MULTIPLY(a, b) a*b
int main()
{
    // The macro is expended as 2 + 3 * 3 + 5, not as 5*8
    printf("%d", MULTIPLY(2+3, 3+5));
    return 0;
}
// Output: 16
```

The macros with arguments should be avoided as they cause problems sometimes. And Inline functions should be preferred as there is type checking parameter evaluation in inline functions.

For example consider the following program. From first look the output seems to be 1, but it produces 36 as output.

```
#include <stdio.h>

#define square(x) x*x
  int main()
   {
    // Expanded as 36/6*6
    int x = 36/square(6);
    printf("%d", x);
    return 0;
  }
```

C Predefined Macros

1. **_DATE_** – Used to represent the current date in MM DD YYYY format.
2. **_STDC_** – Used to indicate when the compiler compiles the code successfully by returning the value 1.
3. **_TIME_** – Represent the current time in HH:MM:SS.
4. **_LINE_** – Represent the current line number.
5. **_FILE_** – Represent the current file name.

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

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