Notes Session Six

Preprocessor	Syntax/Description
Macro	**Syntax: #define** This macro defines constant value and can be any of the basic data types.
Header file inclusion	**Syntax: #include <file_name>** The source code of the file "file_name" is included in the main program at the specified place.
Conditional compilation	**Syntax: #ifdef, #endif, #if, #else, #ifndef** Set of commands are included or excluded in source program before compilation with respect to the condition.
Other directives	**Syntax: #undef, #pragma** #undef is used to undefine a defined macro variable. #Pragma is used to call a function before and after main function in a C program.

Compilers	Interpreters
Compiler reads the entire source code of the program and converts it into binary code. This process is called compilation. Binary code is also referred as machine code, executable, and object code.	Interpreter reads the program source code one line at a time and executing that line. This process is called interpretation.
Program speed is fast.	Program speed is slow.
One time execution. Example: C, C++	Interpretation occurs at every line of the program. Example: BASIC

#Paragma ...

```
void india();
void usa() ;
#pragma startup india 105
#pragma startup usa
#pragma exit usa
#pragma exit india 105

void main(){
     printf("\nI am in main");
     getch();
}

void india(){
     printf("\nI am in india");
     getch();
}

void usa(){
     printf("\nI am in usa");
     getch();
}
```

pragma startup always execute the function **before the main** function **pragma exit** always execute the function **after the main** function.Function declaration of must be before startup and exit pragma directives and function must not take any argument and return void. If more than one startup directive then priority decides which will execute first.

startup:

Lower value: higher priority i.e. functions will execute first. If more than one exit directive then priority decides which will execute first.

exit:

Higher value: higher priority i.e. functions will execute first. For example

```
Output:
I am in usa
I am in India
I am in main
I am in India
I am in usa
```

#pragma startup [priority]
#pragma exit [priority]

Where priority is optional integral number.
For user priority varies from 64 to 255
For c libraries priority varies from 0 to 63
Default priority is 100.

Macros ==Text Replacement

__VA_ARGS__ & # &

```
#define Dprintf(...) printf (__VA_ARGS__);
```

Syntax

`#define` *identifier replacement-list*(optional)	(1)	
`#define` *identifier*`(` *parameters* `)` *replacement-list*	(2)	
`#define` *identifier*`(` *parameters*`, ... )` *replacement-list*	(3)	(since C99)
`#define` *identifier*`( ... )` *replacement-list*	(4)	(since C99)
`#undef` *identifier*	(5)	

```
1 //Prepared by Eng.Keroles
2 #include <stdio.h>
3
4
5 //make function factory and use it
6 #define FUNCTION(name, a) int fun_##name(int x) { return (a)*x;}
7
8 FUNCTION(quadruple, 4)
9 FUNCTION(double, 2)
10
11 #undef FUNCTION
12 #define FUNCTION 34
13 #define OUTPUT(a) printf( #a )
14
15 int main(int argt ,char**argv){
16     printf("quadruple(13): %d\n", fun_quadruple(13) );
17        printf("double(21): %d\n", fun_double(21) );
18        printf("%d\n", FUNCTION);
19        OUTPUT(Keroles);    //convert million to string using #
20 }
21
```

```
quadruple(13): 52
double(21): 42
34
Keroles
```

Predefined macros

```
1 //Prepared by Eng.Keroles
2 #include <stdio.h>
3
4 int main(int argt ,char**argv){
5      printf("__STDC__ = %ld  \n",__STDC__ );
6      printf("__STDC_VERSION__ = %ld  \n",__STDC_VERSION__ );
7      printf("__STDC_HOSTED__ = %ld  \n",__STDC_HOSTED__ );
8      printf("__FILE__ = %s  \n",__FILE__ );
9      printf("__LINE__ = %d  \n",__LINE__ );
10     printf("__DATE__ = %s  \n",__DATE__ );
11     printf("__TIME__ = %s  \n",__TIME__ );
12     printf("__func__ = %s  \n",__func__ );
13
14 return 0 ;
15 }
```

```
__STDC__ = 1
__STDC_VERSION__ = 201112
__STDC_HOSTED__ = 1
__FILE__ = ../main.c
__LINE__ = 9
__DATE__ = Apr 21 2017
__TIME__ = 11:54:28
__func__ = main
```


ENG. Keroles Shenouda
https://www.facebook.com/groups/embedded.system.KS/