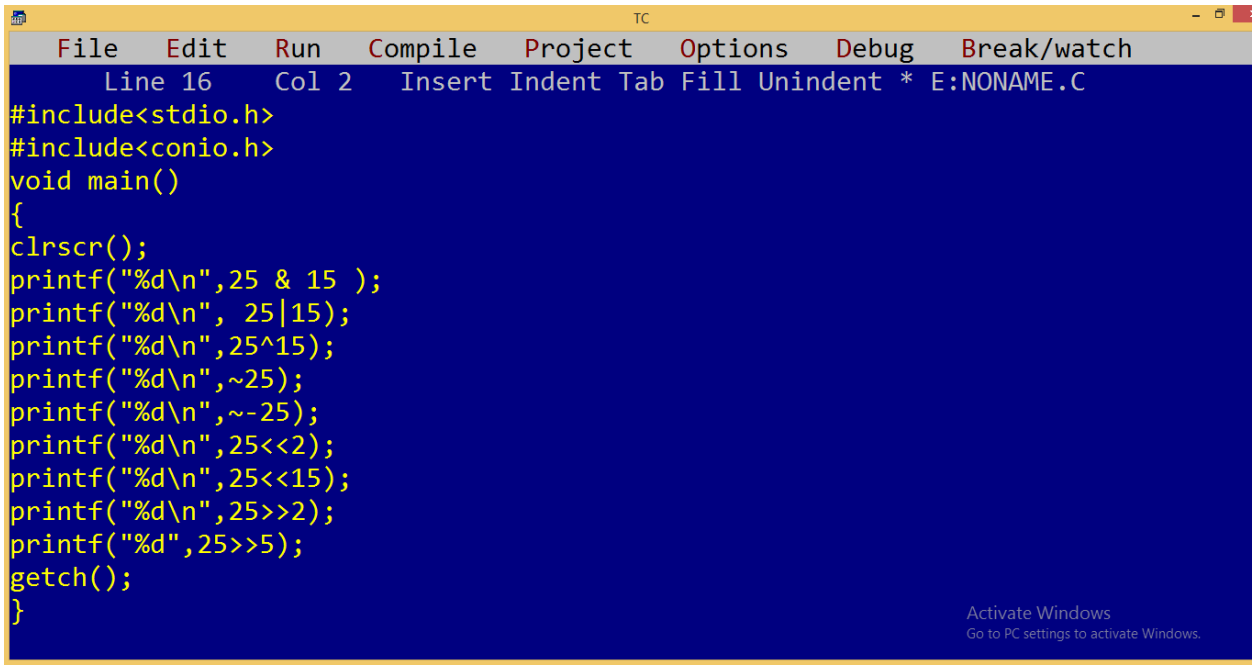
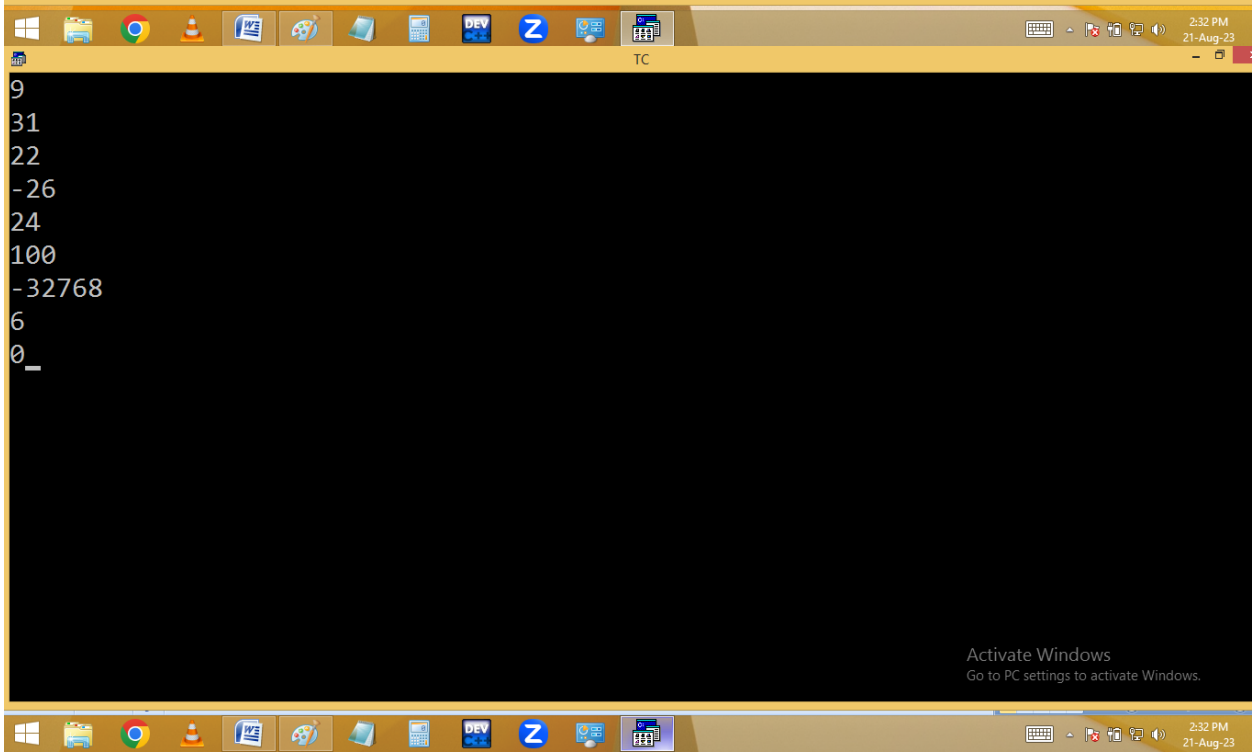
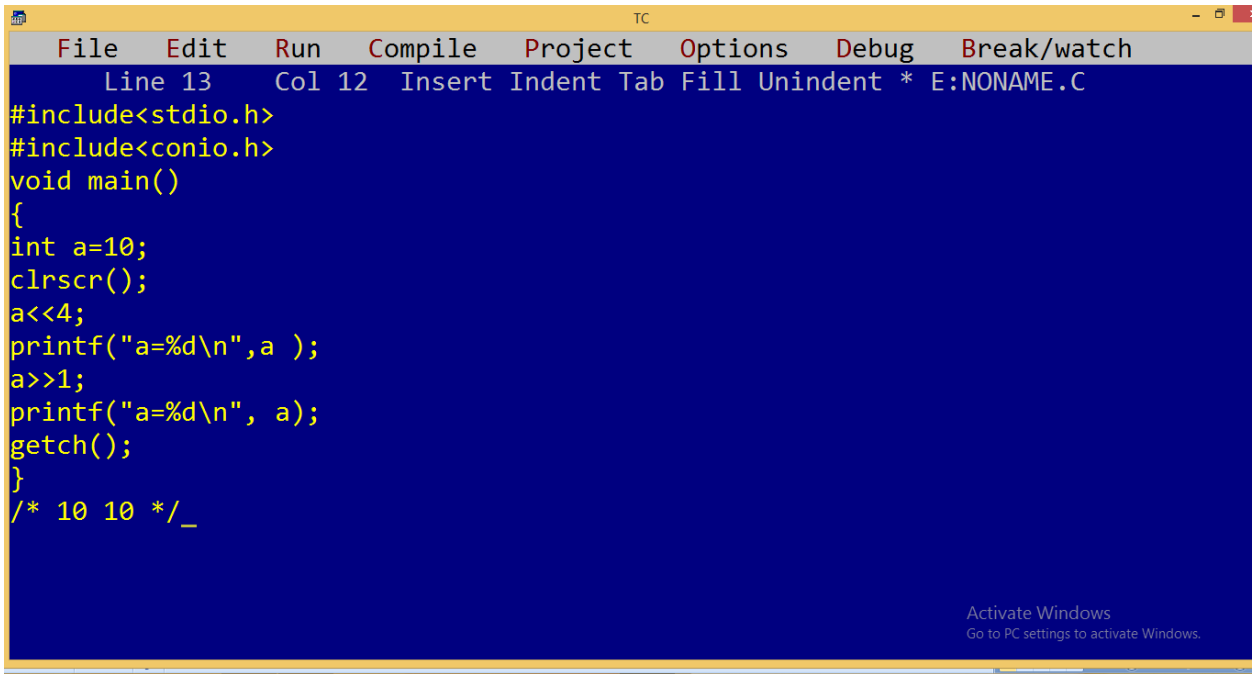
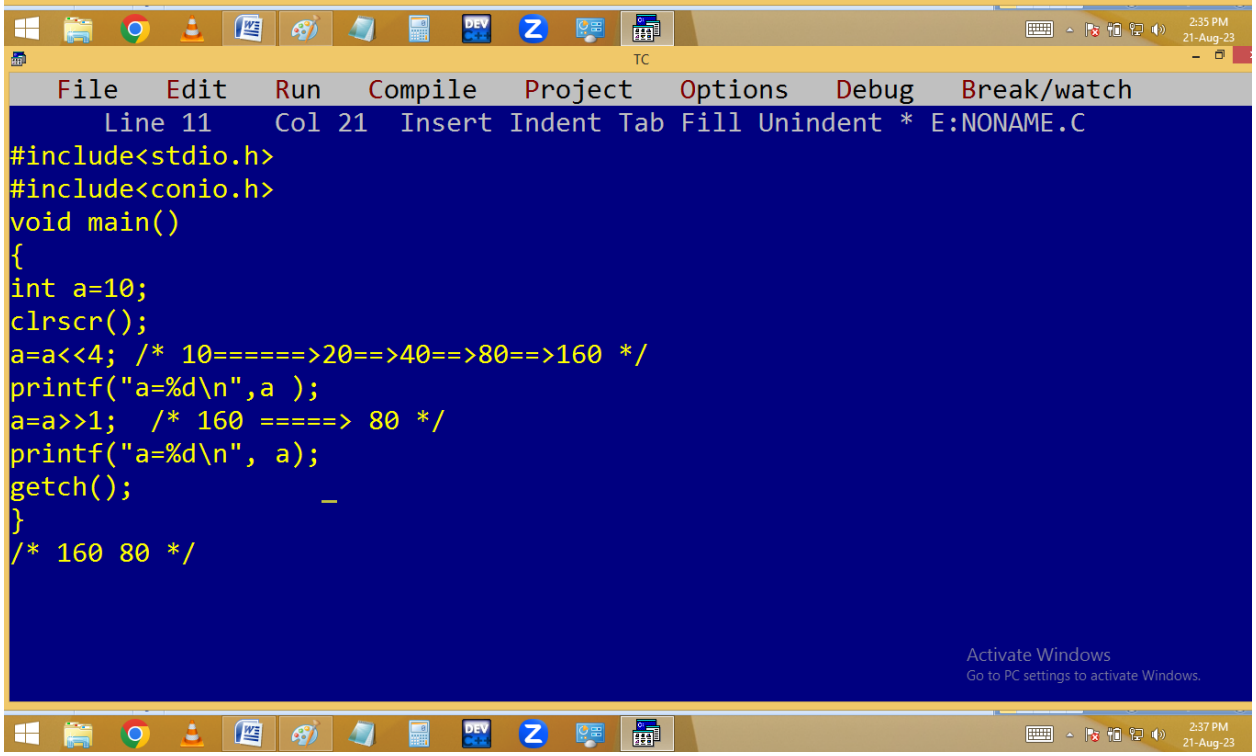
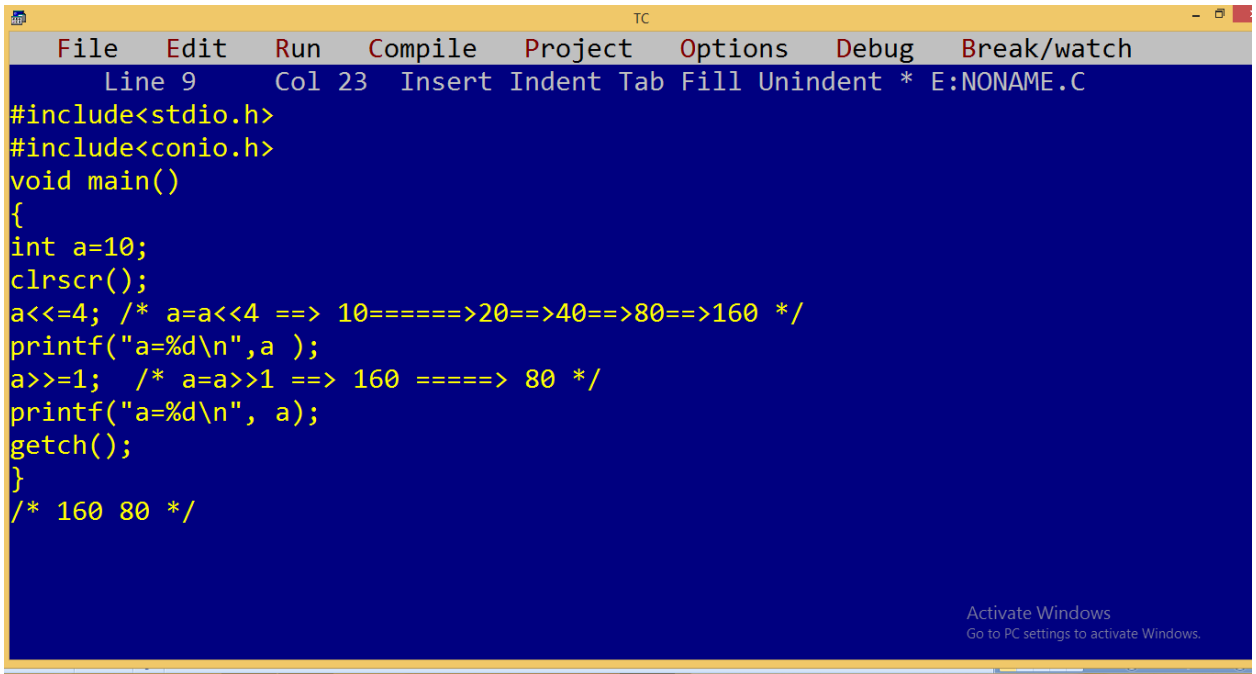
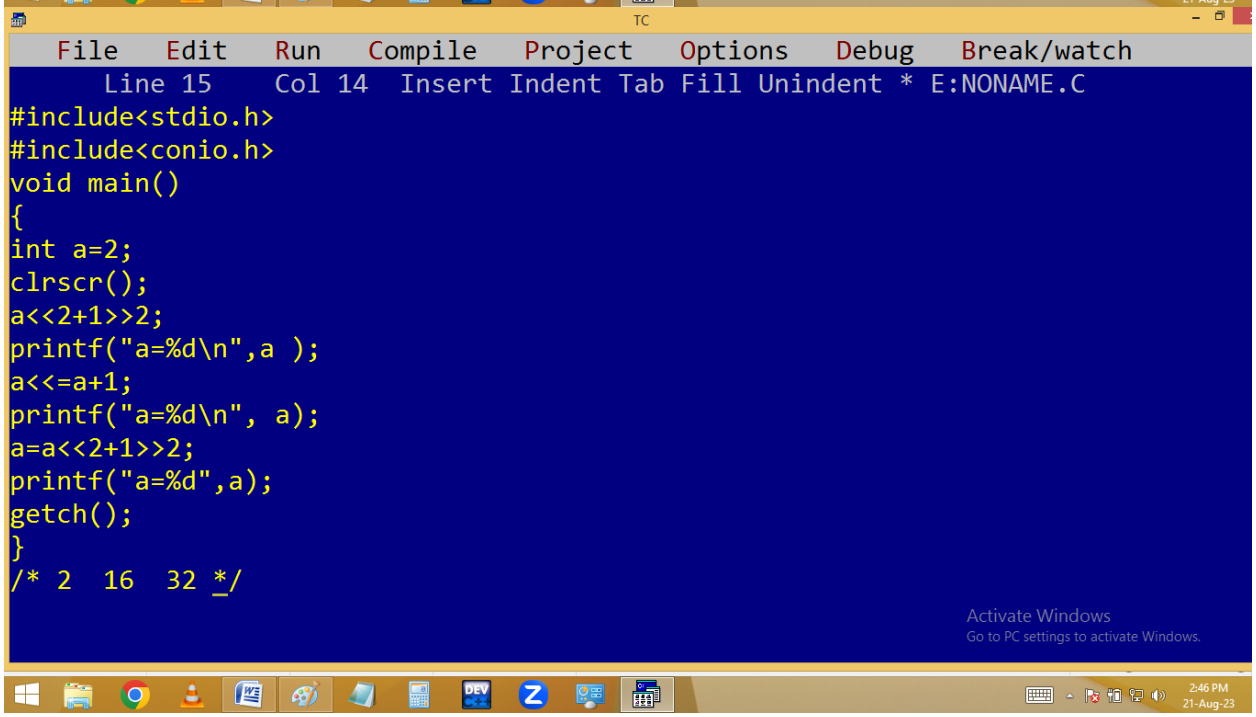

TC
File Edit Run Compile Project Options Debug Break/watch
Line 16 Col 2 Insert Indent Tab Fill Unindent * E:NONAME.C
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
printf("%d\n",25 & 15);
printf("%d\n", 25|15);
printf("%d\n",25^15);
printf("%d\n",~25);
printf("%d\n",~-25);
printf("%d\n",25<<2);
printf("%d\n",25<<15);
printf("%d\n",25>>2);
printf("%d",25>>5);
getch();
}
Activate Windows
Go to PC settings to activate Windows.


TC
9
31
22
-26
24
100
-32768
6
0
Activate Windows
Go to PC settings to activate Windows.


TC
File Edit Run Compile Project Options Debug Break/watch
Line 13 Col 12 Insert Indent Tab Fill Unindent * E:NONAME.C
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10;
clrscr();
a<<4;
printf("a=%d\n",a);
a>>1;
printf("a=%d\n", a);
getch();
}
/* 10 10 */
Activate Windows
Go to PC settings to activate Windows.
2:35 PM
21-Aug-23


TC
File Edit Run Compile Project Options Debug Break/watch
Line 11 Col 21 Insert Indent Tab Fill Unindent * E:NONAME.C
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10;
clrscr();
a=a<<4; /* 10======>20==>40==>80==>160 */
printf("a=%d\n",a);
a=a>>1; /* 160 =====> 80 */
printf("a=%d\n", a);
getch();
}
/* 160 80 */
Activate Windows
Go to PC settings to activate Windows.
2:37 PM
21-Aug-23


TC
File Edit Run Compile Project Options Debug Break/watch
Line 9 Col 23 Insert Indent Tab Fill Unindent * E:NONAME.C
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10;
clrscr();
a<<=4; /* a=a<<4 ==> 10======>20==>40==>80==>160 */
printf("a=%d\n",a);
a>>=1; /* a=a>>1 ==> 160 =====> 80 */
printf("a=%d\n", a);
getch();
}
/* 160 80 */
Activate Windows
Go to PC settings to activate Windows.
2:38 PM
21-Aug-23


TC
File Edit Run Compile Project Options Debug Break/watch
Line 15 Col 14 Insert Indent Tab Fill Unindent * E:NONAME.C
#include<stdio.h>
#include<conio.h>
void main()
{
int a=2;
clrscr();
a<<2+1>>2;
printf("a=%d\n",a);
a<<=a+1;
printf("a=%d\n", a);
a=a<<2+1>>2;
printf("a=%d",a);
getch();
}
/* 2 16 32 */
Activate Windows
Go to PC settings to activate Windows.
2:46 PM
21-Aug-23

a=2

a<<=a+1; i.e. a=a<<2+1; ==> a=a<<3 ==> 2==> 4 ==> 8 ==> 16

a=a<<2+1>>2 ==> a=a<<3>>2

a=a<<3 ==> 16 ===> 32 ==> 64 ==> 128

a=a>>2 ==> 128 ===> 64 ==> 32

C - PROGRAM STRUCTURE

It illustrates[Describes] how to write a program in c-language.

Every programming language is having a particular structure and we should have to follow this structure.

C-Programming structure is divided into the following parts.

- **[documentation section]**
- **Header files / Proto types / Preprocessor**
- **[global variables]**
- **[function declarations & definitions]**

- **void main() / main() / int main()**
- **Other statements.**

Generally documentation section consists of program headings, definitions etc and They should be represented with comments.

The statements that are enclosed in between /* and */ are called comments.

Comments never participate in program execution. They are only for user understandability or display purpose.

C-Language supports comment block only.

Eg:

/*

……;

……;

*/

C++ supports comment block and single line comments.

Eg: // ……

Header files consists of function definitions, global variables, macros etc.

We can declare the header files at any place of our program. But before going to use the relevant function, its header file should be declared. It is recommended to declare the header files at the top of the program.

Every header file should be started with **#include**. Here **#** is a **preprocessor** indicator.

We can place header files in angled brackets

< > or double quotes **“ “**.

Header file never ends with **semicolon**(**;**).

Note: In C^{++}, we should have to declare header files at the top only.

The variables that are declared before main() or top of the program are called **global variables** and they can be accessed from anywhere in our program. They are optional.

Function declarations and definitions contain function header and body.

* Every C-Program execution starts from main() function and travel towards down. Hence it is also called **top-down** approach.

* Without main(), C-Program never executed but compiled.

* main() is predefined function with user defined body. main() doesn't have any header file. One program have to maintain one main() only. We can create alternate for main(). Other

statements are changed from program to program.

Note: It is recommended to write C programs in lower case only. Every statement should have to end with semicolon except header files, control statements, main().