POINTERS

Pointer is a variable, which holds the address of another variable of same type.

Pointer is a memory location, which holds the address of another memory location.

Pointer is a derived data type.

Advantages:

1. Dynamic memory allocation.
2. Program performance is increased due to preventing memory wastage.
3. They are very much used in System programming.
4. They are very much used in dynamic linked list & Stacks [data structures].
5. It allows to access a local variable outside the function i.e. data sharing between functions. [call by address].
6. To handle strings, arrays etc in functions we need pointers.
7. To handle data files we are using pointers.

8. They directly works on variable address. Due to this search time is reduced and execution speed is increased.

Disadvantage:

They are not secured and make programmer complex.

Syntax:

datatype * variable;

- * indicates it is a pointer data type.
- * is called indirection operator.
- * is called dereferencing operator.
- *** is a re-direction operator.**
- * indicates value at that address.
- * indicates pointer value.

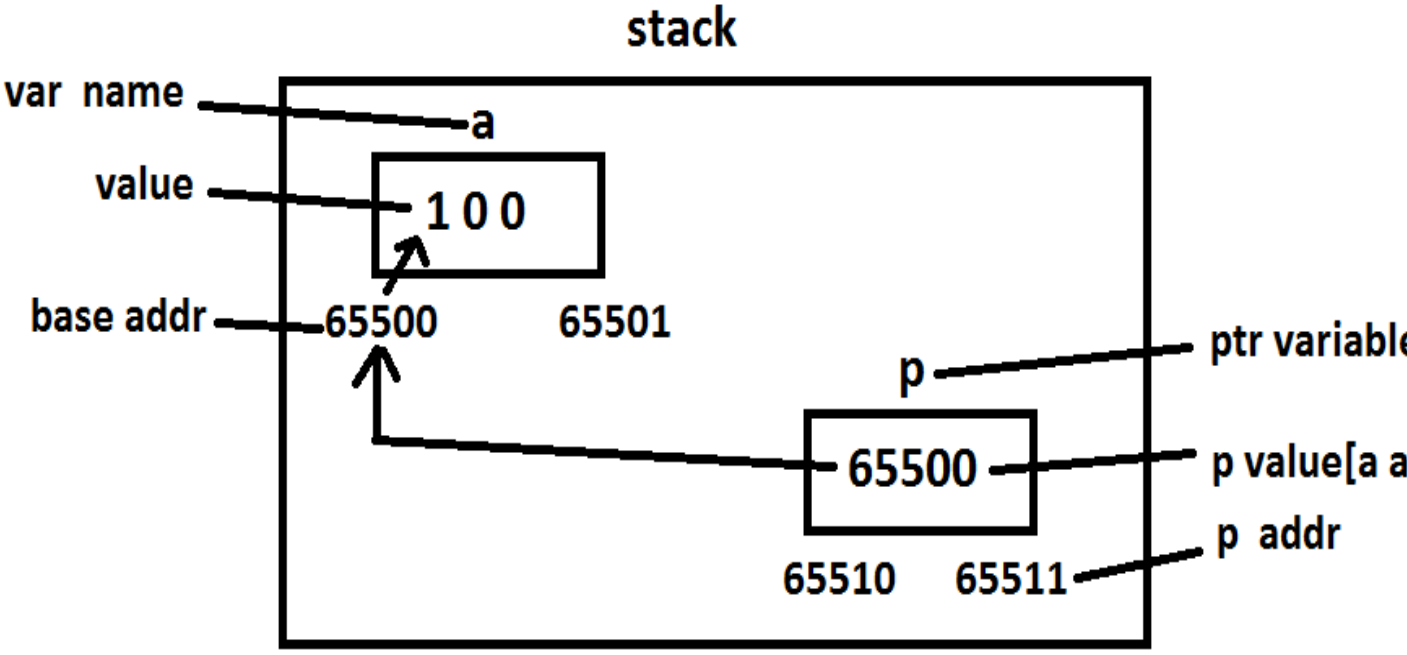
Eg:

```
int a=100, * p;
```

In the above example '**a**' is a general variable.

***** indicates '**p**' is a pointer type variable and it is able to store the address of general variable **'a'** as follows.

```
p =  &a;
```



In the above example, to pick the value of **a** through pointer variable **p**, we have to use the **printf()** as follows.

Here *p means **value of p** or **value at that addr**. i.e. **65500**. But 65500 is the addr of '**a**'. The value in **a** address is **100**.

Or

Here p means 65500. *p means value at 65500. i.e. 100.

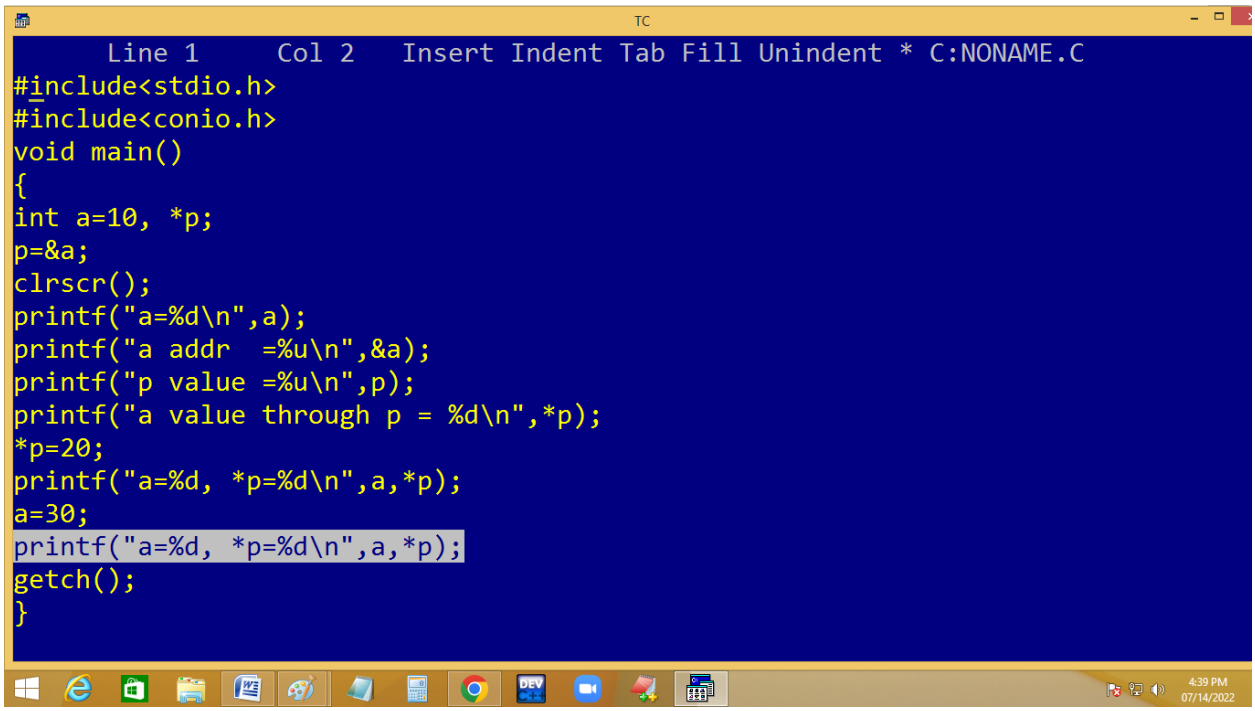
Due to this example any changes we have conducted in ***p** effects the value of **'a'**. Hence **p** is called **pointer** to **a**.

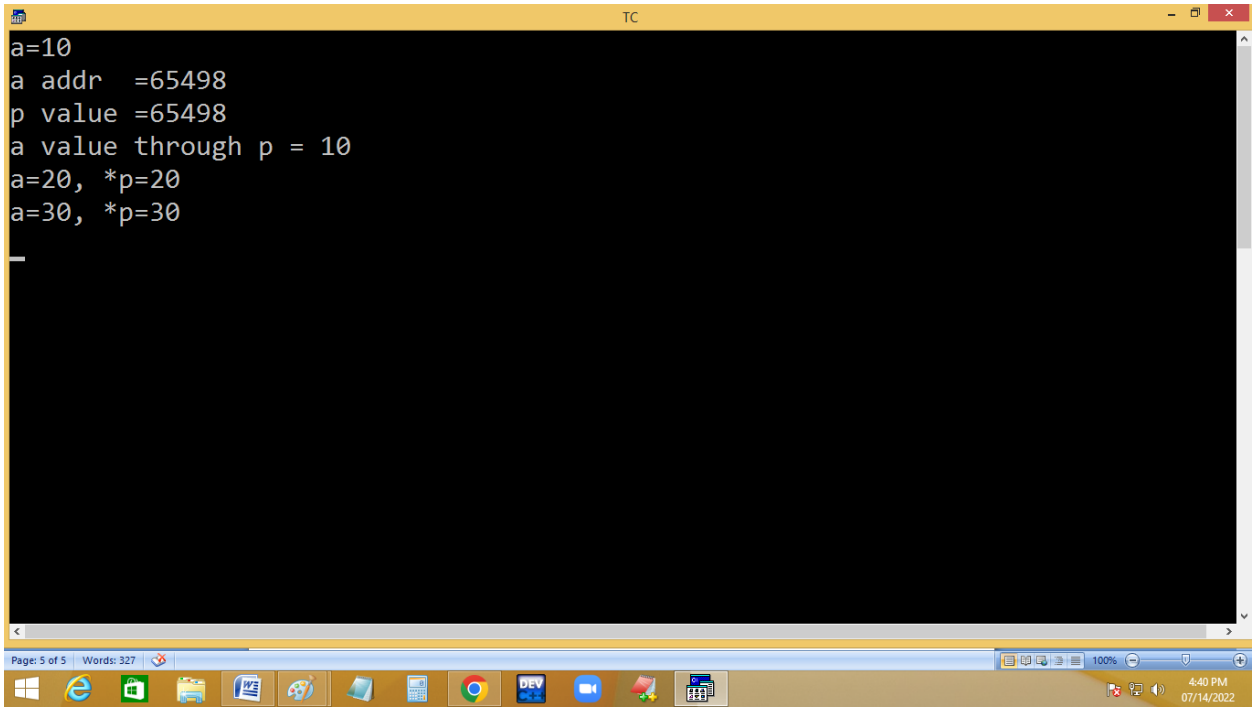
Eg: *p=200;

Now a becomes 200.

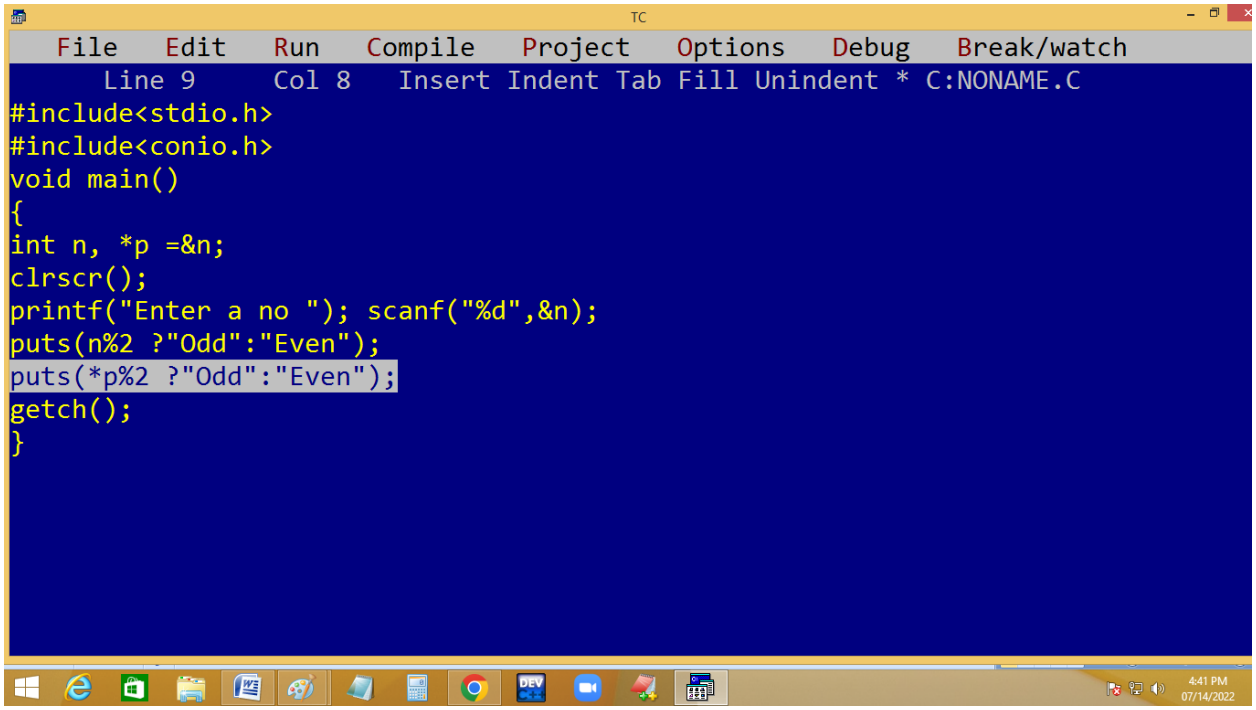
Eg:

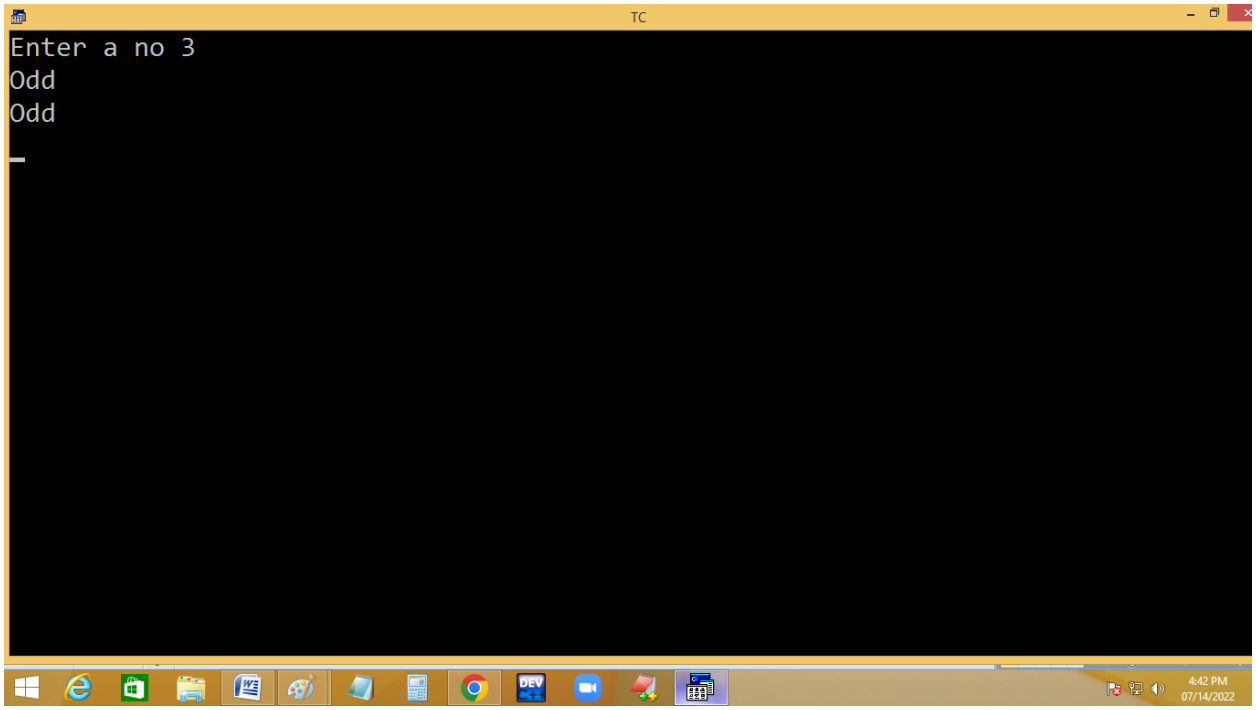
Finding a variable value and address using a pointer:

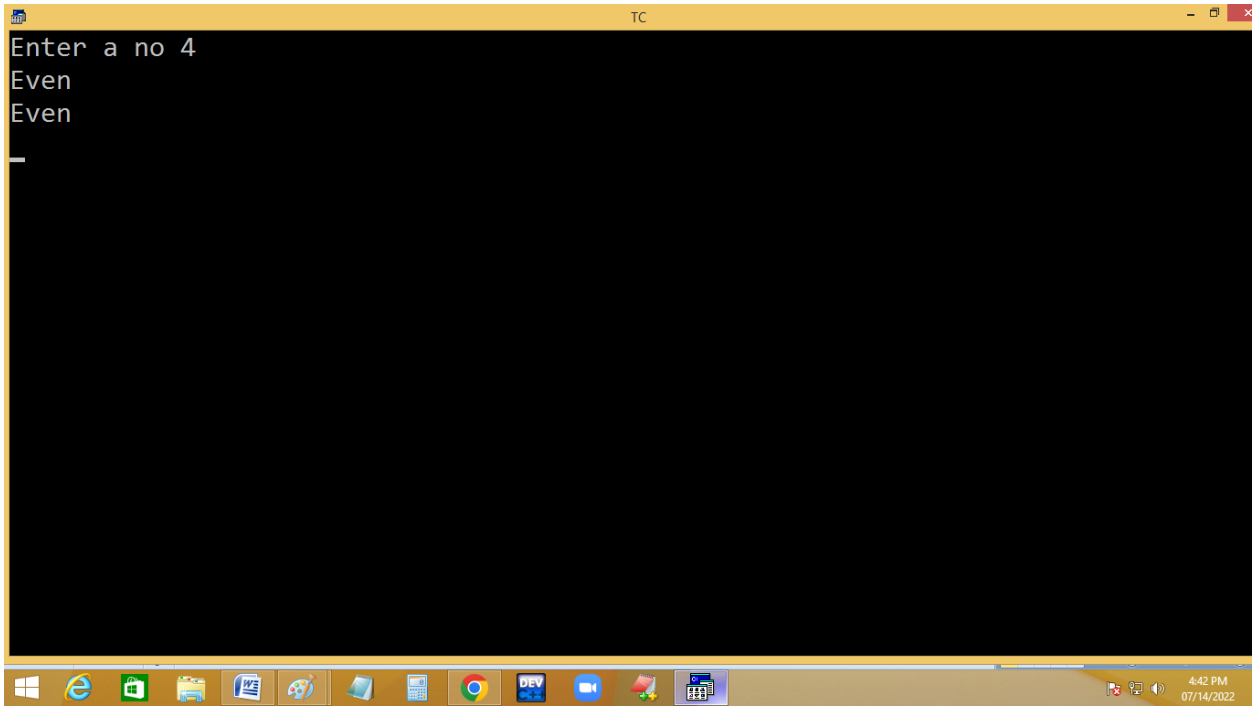

TC
Line 1 Col 2 Insert Indent Tab Fill Unindent * C:NONAME.C
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10, *p;
p=&a;
clrscr();
printf("a=%d\n",a);
printf("a addr =%u\n",&a);
printf("p value =%u\n",p);
printf("a value through p = %d\n",*p);
*p=20;
printf("a=%d, *p=%d\n",a,*p);
a=30;
printf("a=%d, *p=%d\n",a,*p);
getch();
}
4:39 PM
07/14/2022


TC
a=10
a addr =65498
p value =65498
a value through p = 10
a=20, *p=20
a=30, *p=30
Page: 5 of 5
Words: 327
100%
4:40 PM
07/14/2022

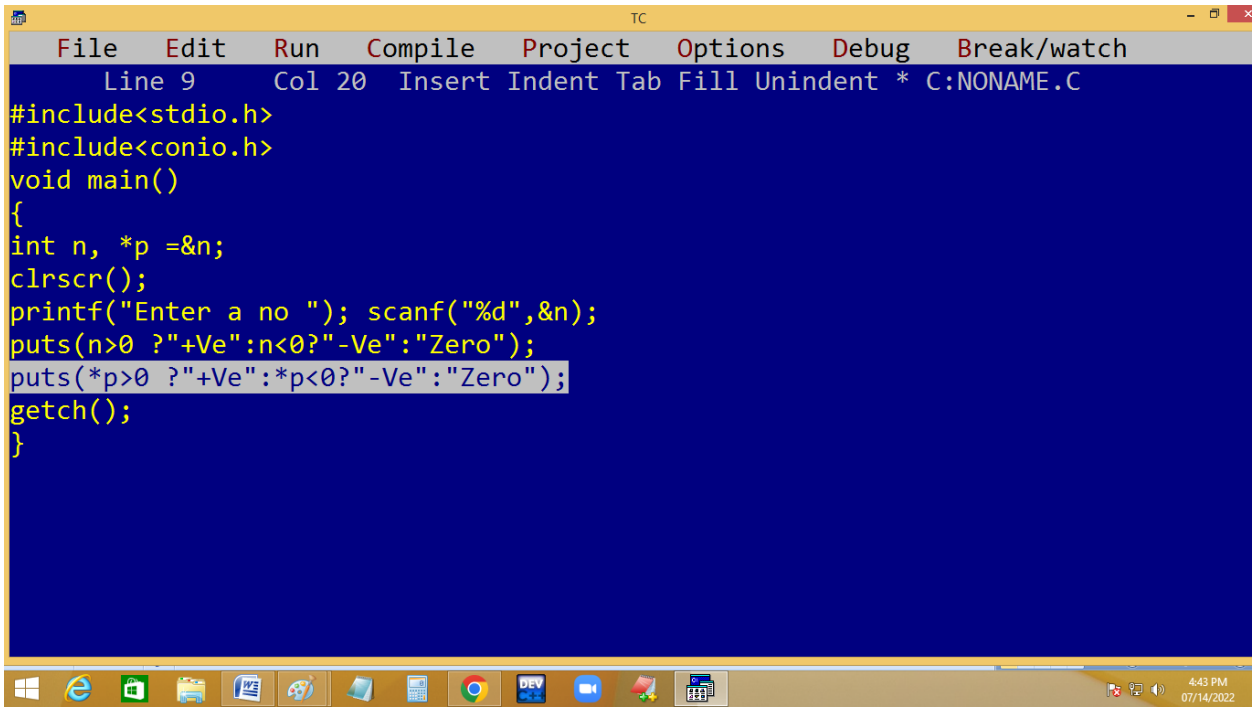
Eg. finding even/odd using pointer.

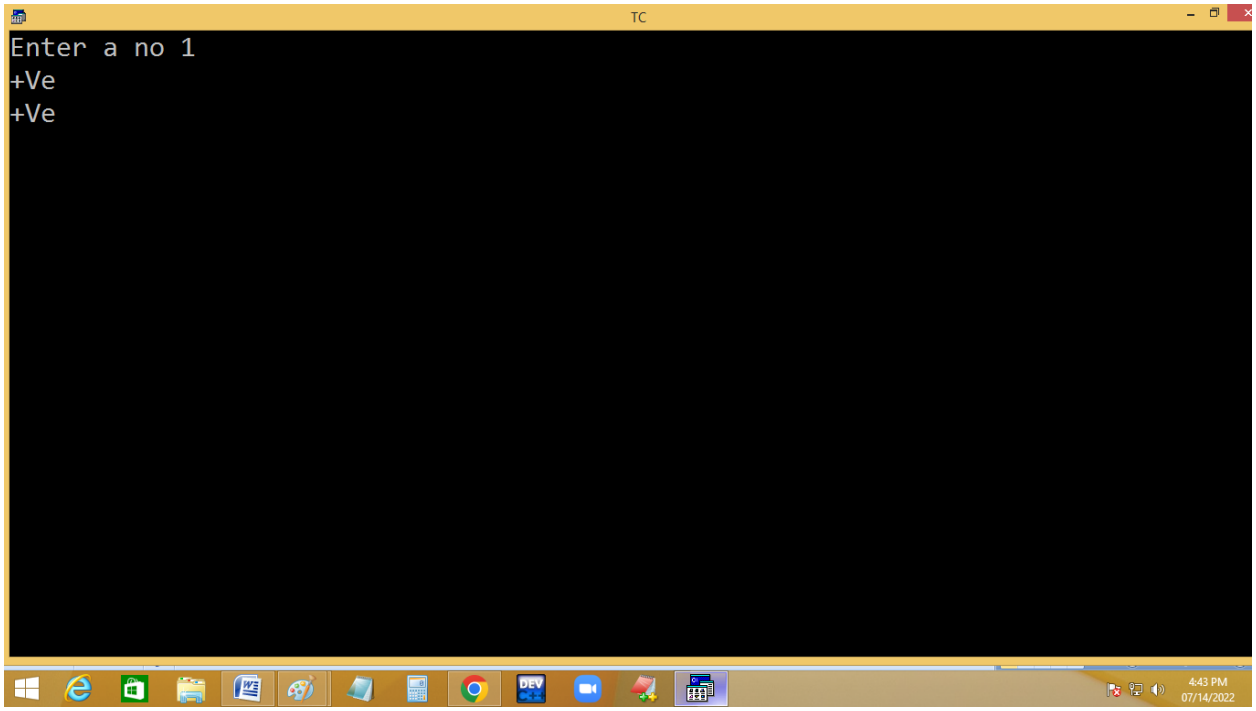


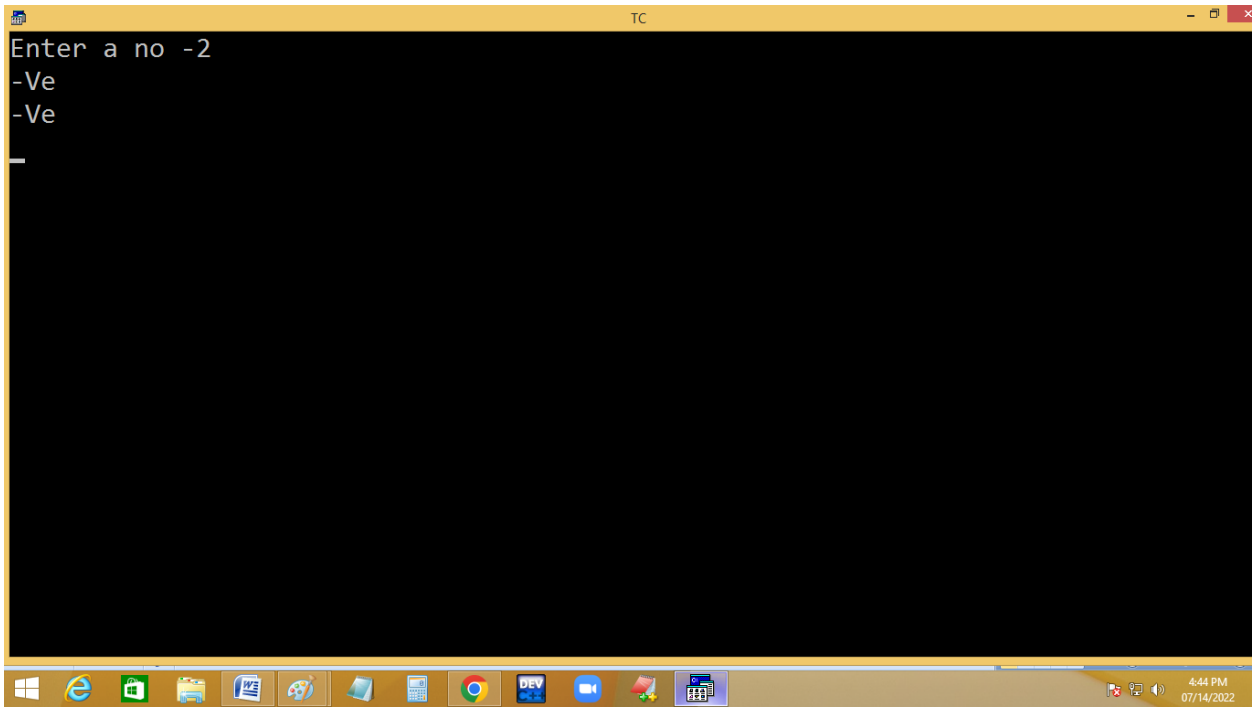


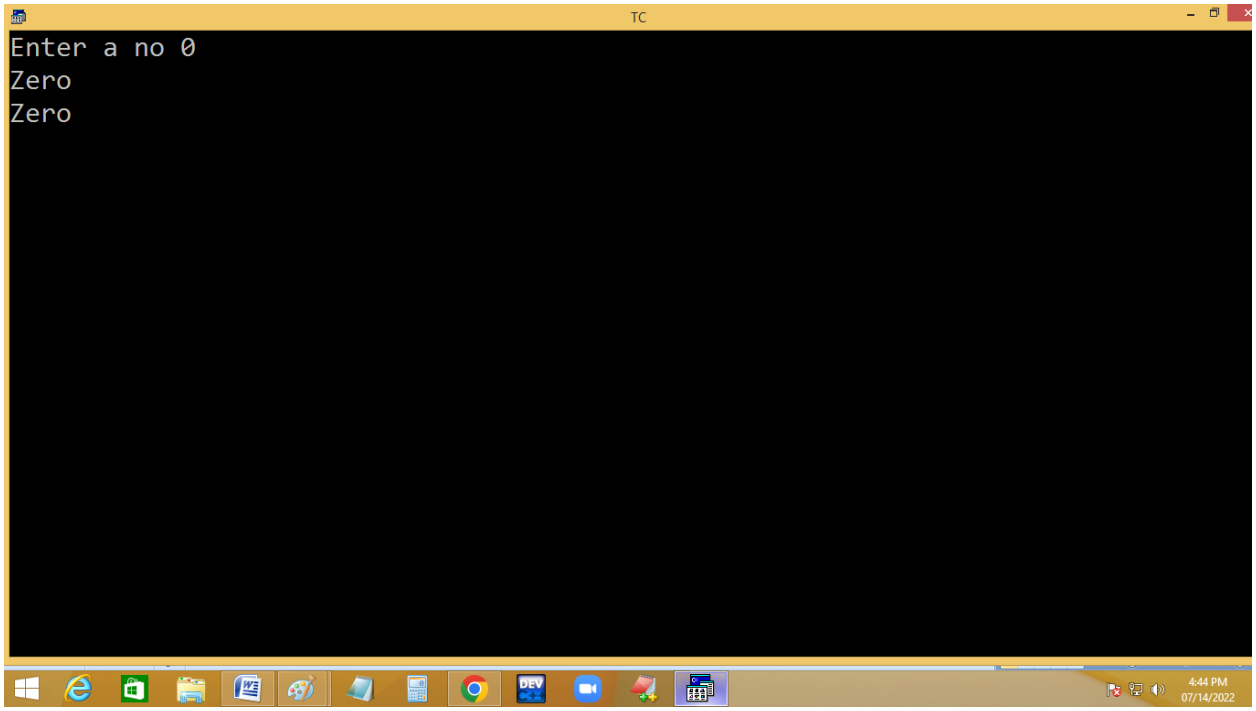


Eg. Finding +ve/-ve/0 using pointer.

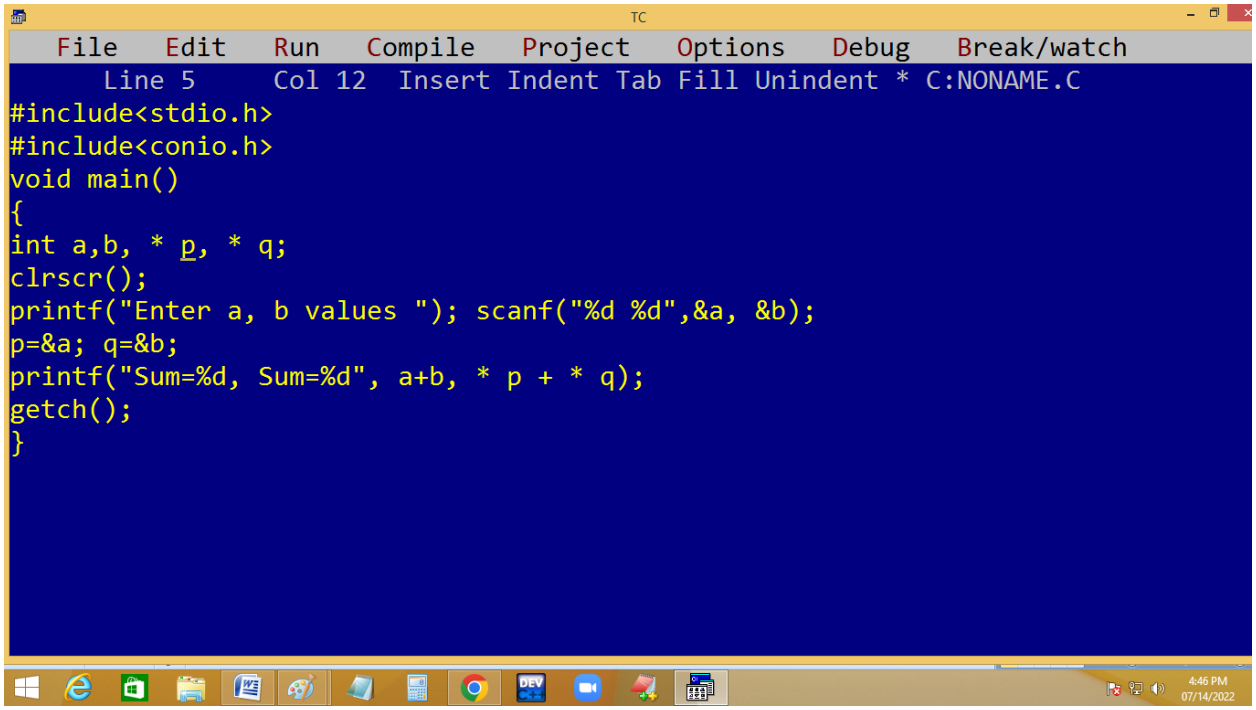

TC
File Edit Run Compile Project Options Debug Break/watch
Line 9 Col 20 Insert Indent Tab Fill Unindent * C:NONAME.C
#include<stdio.h>
#include<conio.h>
void main()
{
int n, *p =&n;
clrscr();
printf("Enter a no "); scanf("%d",&n);
puts(n>0 ?"+Ve":n<0?"-Ve":"Zero");
puts(*p>0 ?"+Ve":*p<0?"-Ve":"Zero");
getch();
}

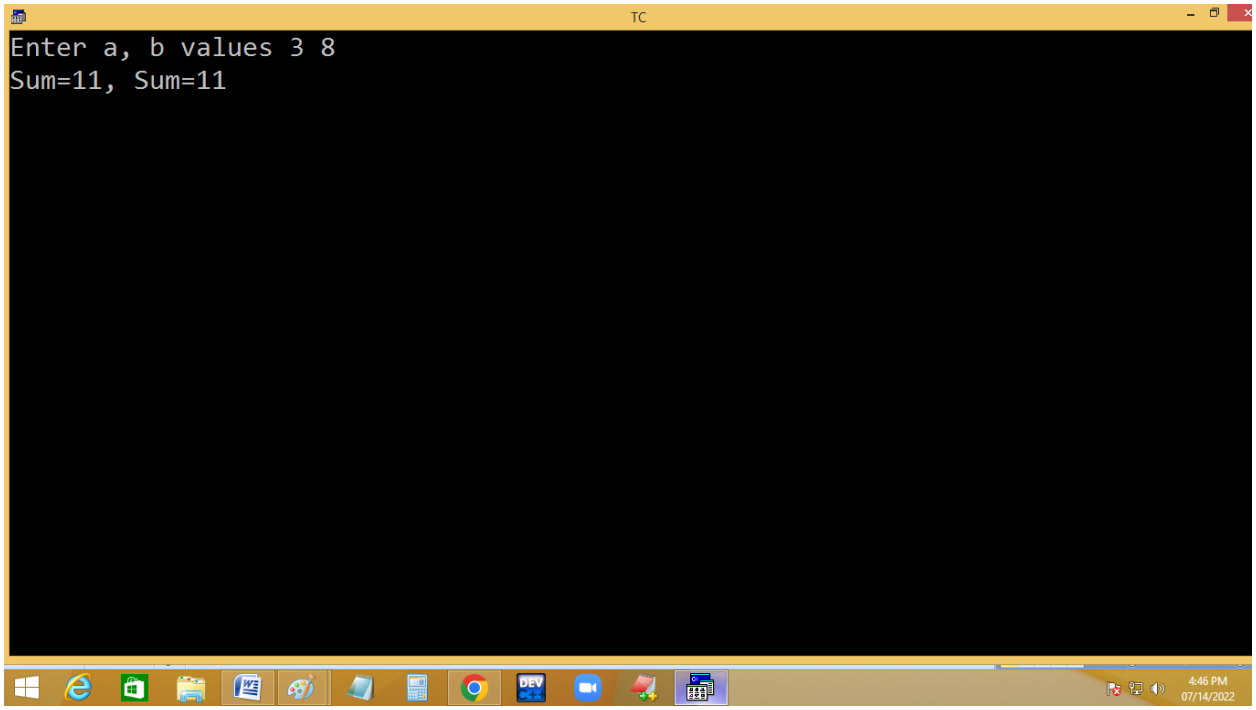

TC
Enter a no 1
+Ve
+Ve


TC
Enter a no -2
-Ve
-Ve
4:44 PM
07/14/2022

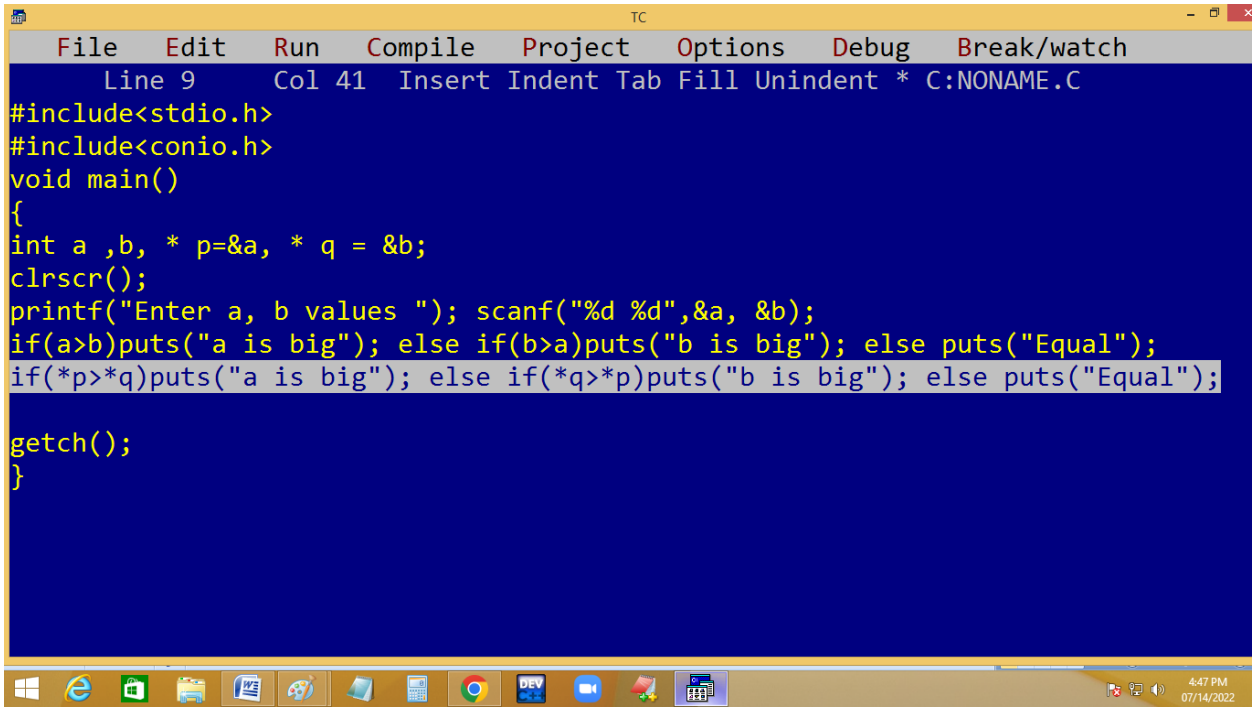

TC
Enter a no 0
Zero
Zero
4:44 PM
07/14/2022

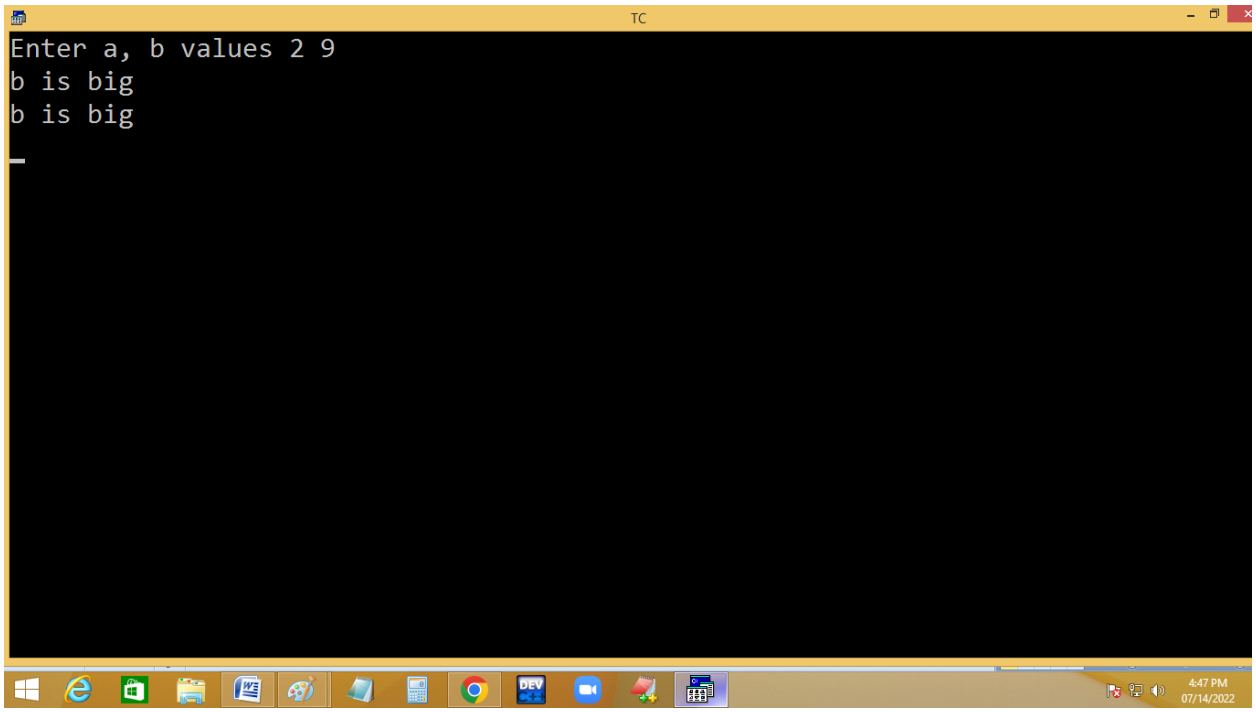
Eg. Add two numbers using pointers.

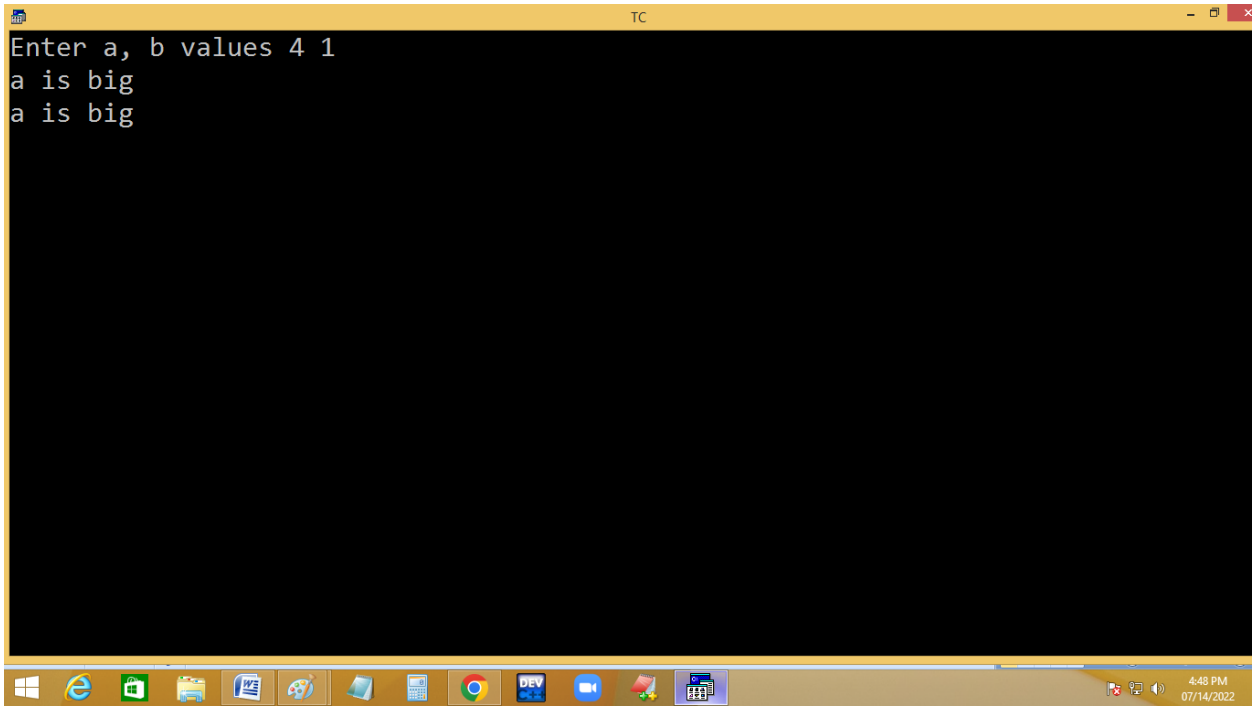




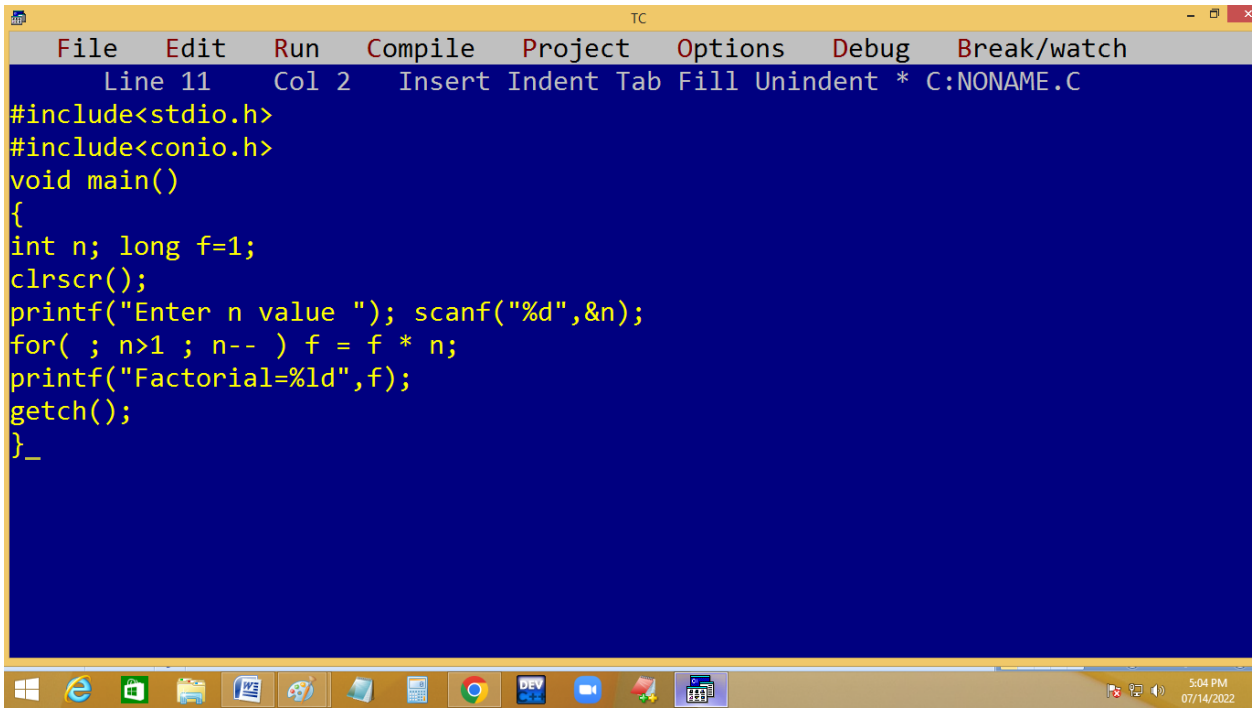
Eg. find max in 2 numbers using pointer.

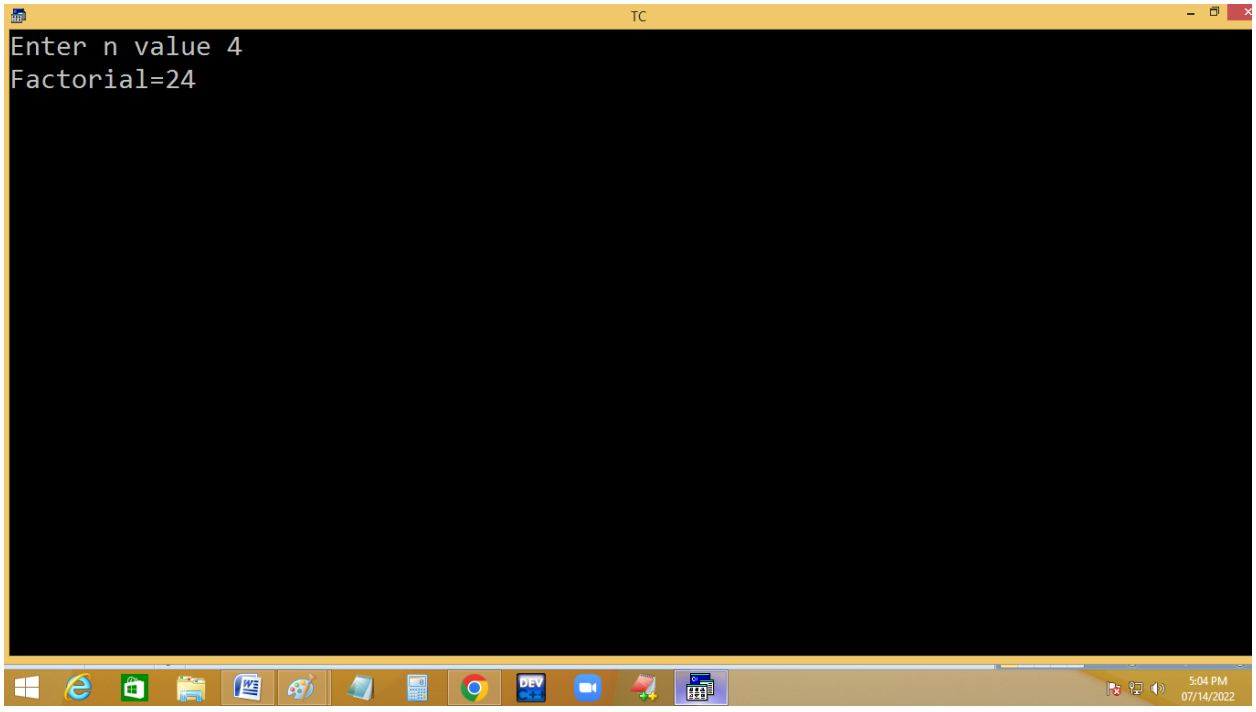


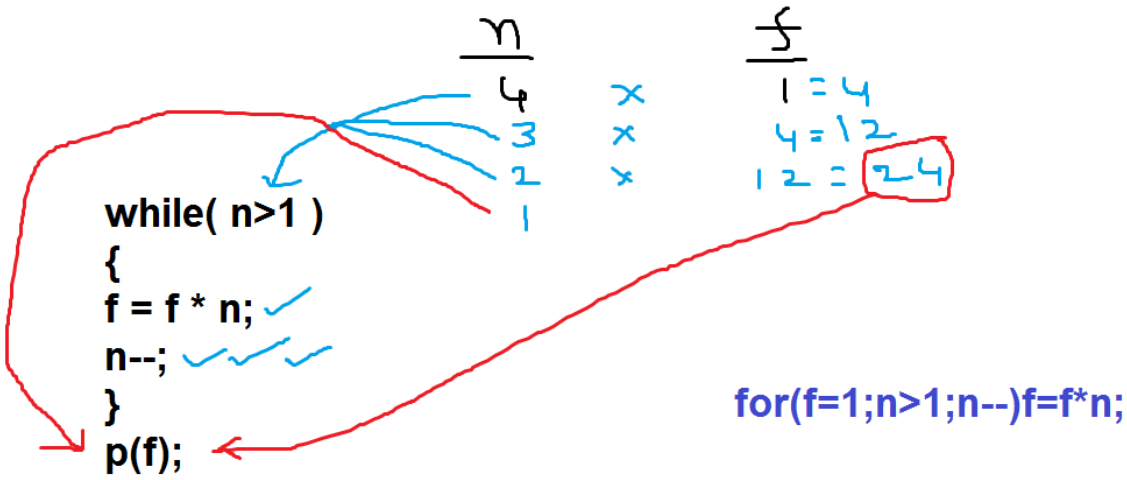


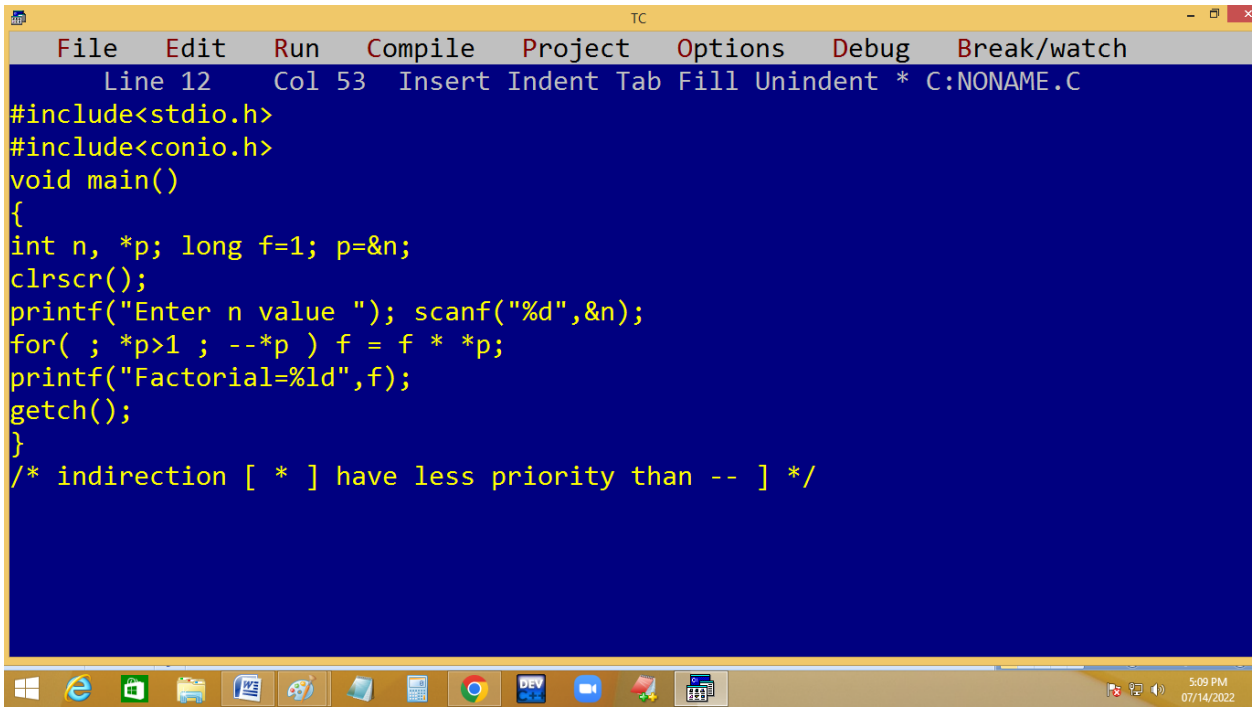

TC
Enter a, b values 4 1
a is big
a is big
4:48 PM
07/14/2022

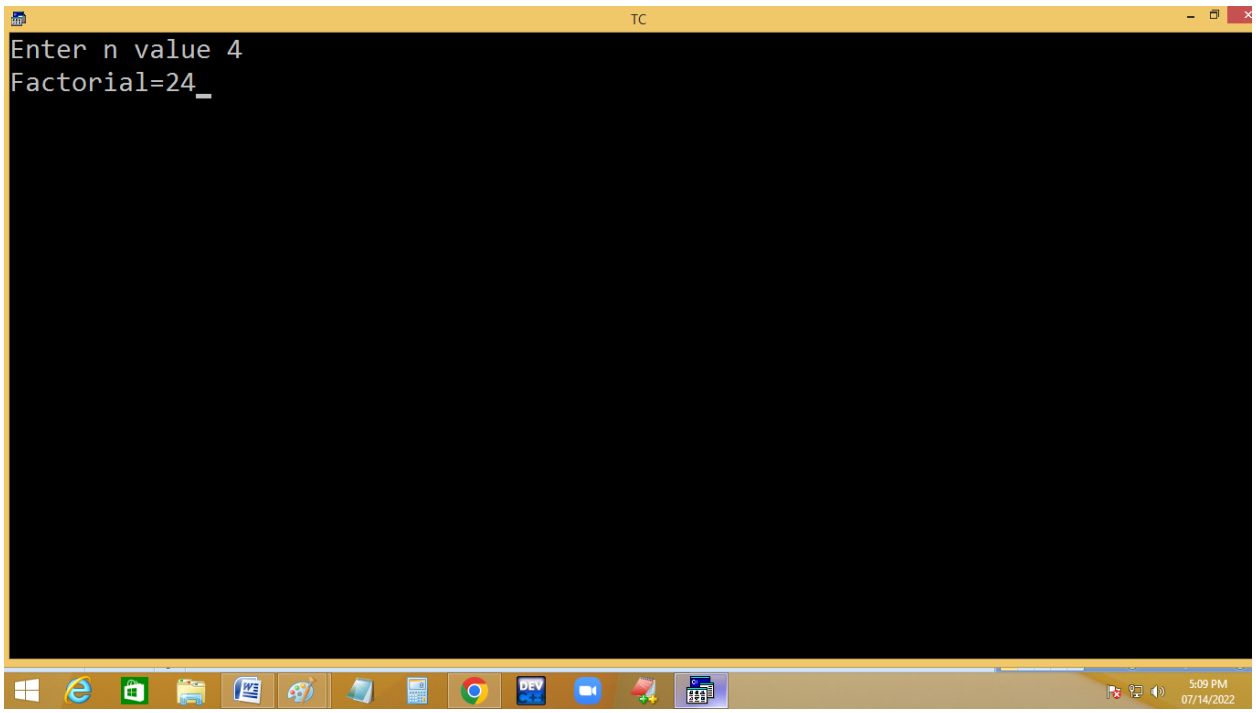
Finding factgorial using pointer.



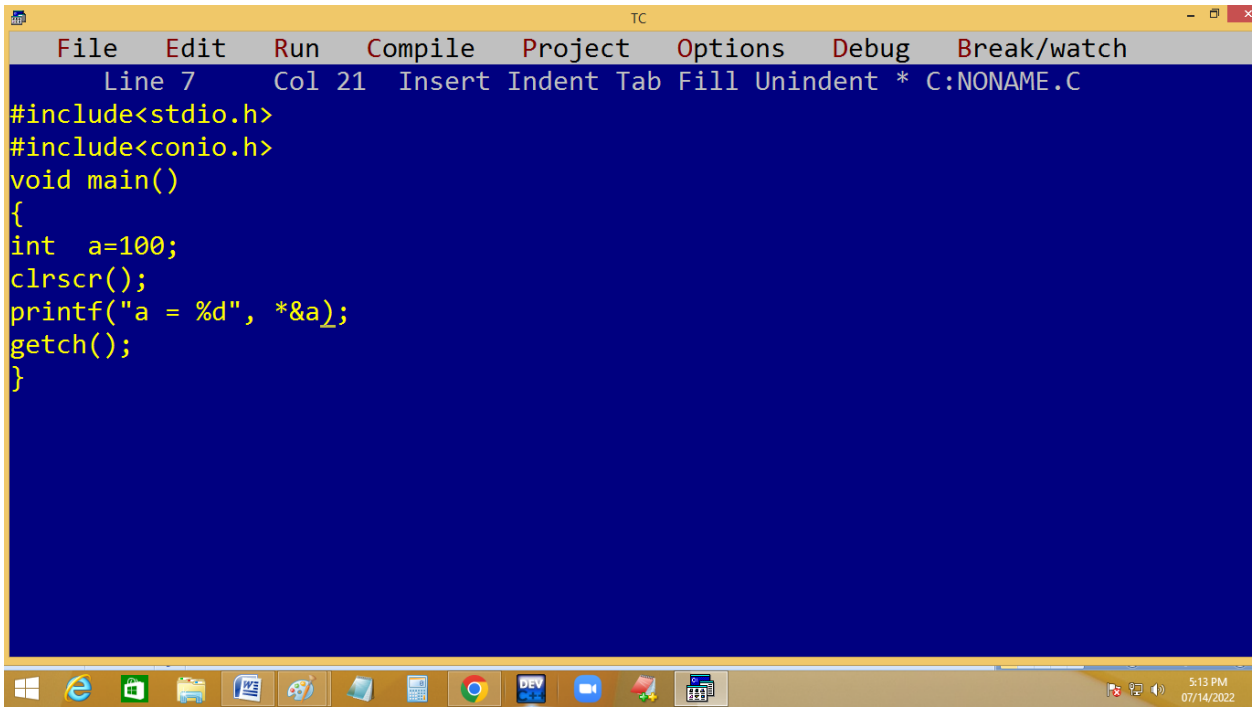


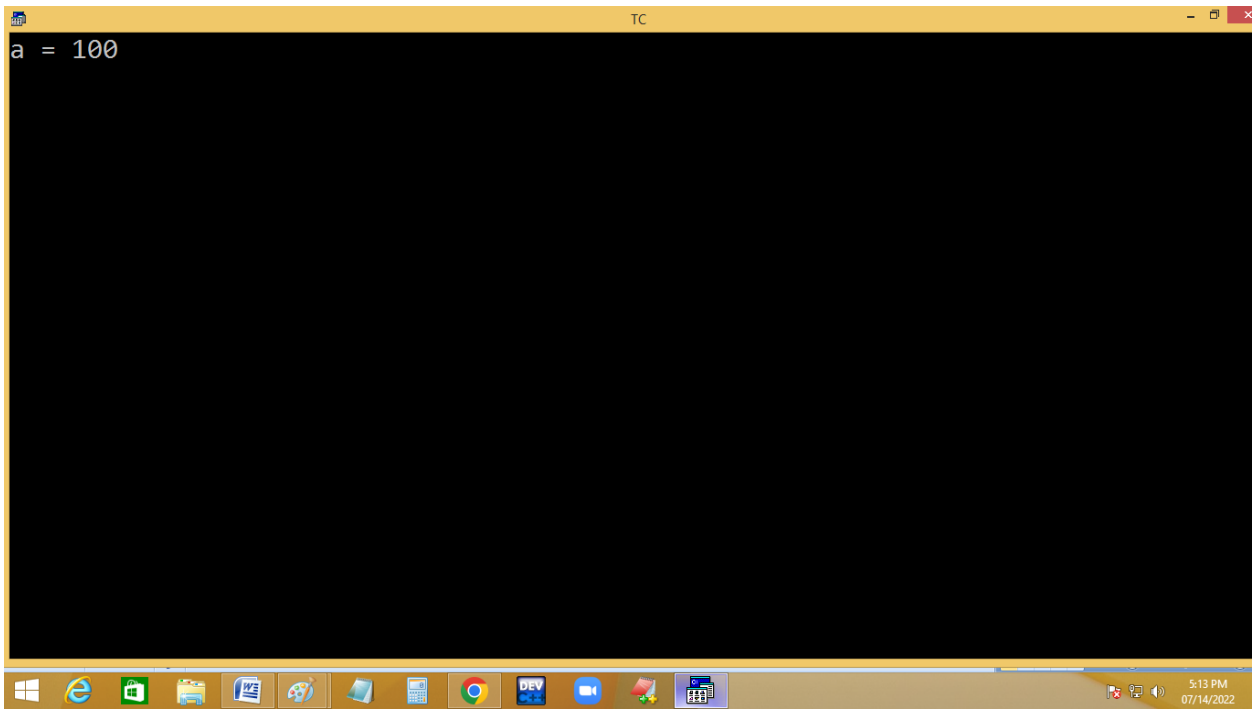

n
f
4 x 1 = 4
3 x 4 = 12
2 x 12 = 24
1
while(n>1)
{
f = f * n;
n--;
}
p(f);
for(f=1;n>1;n--)f=f*n;





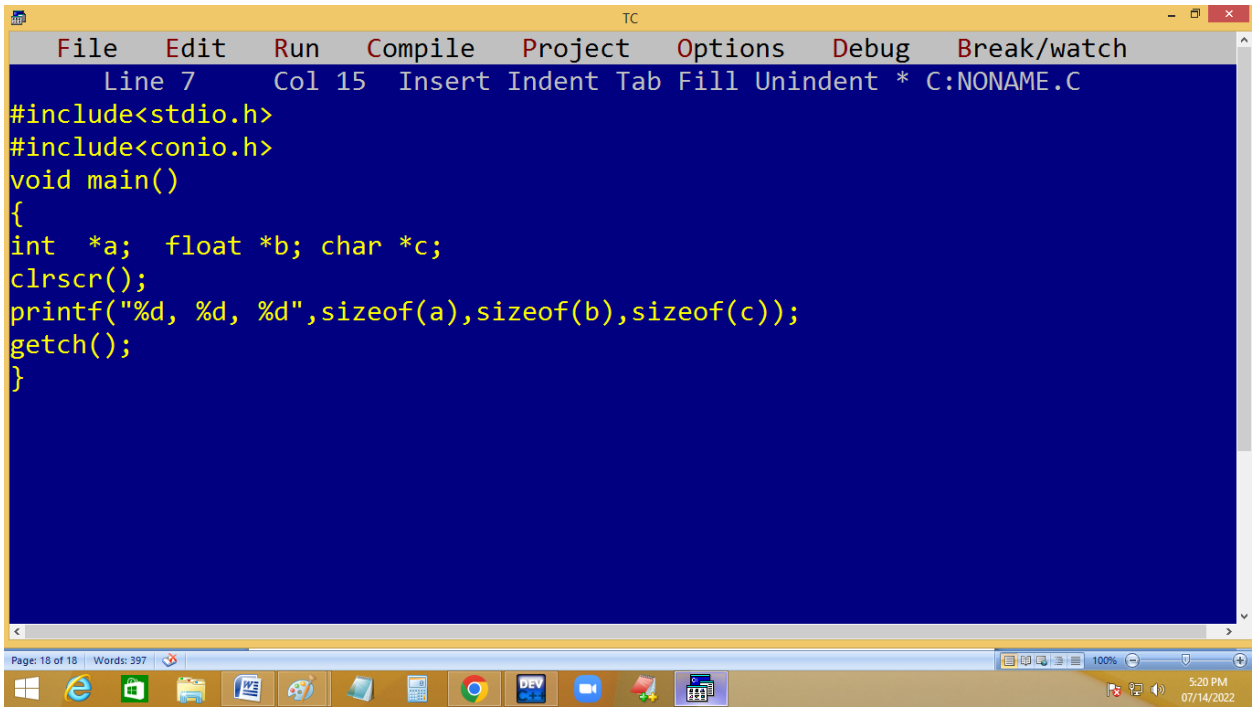
Eg. printing a normal variable value using pointer mechanism.

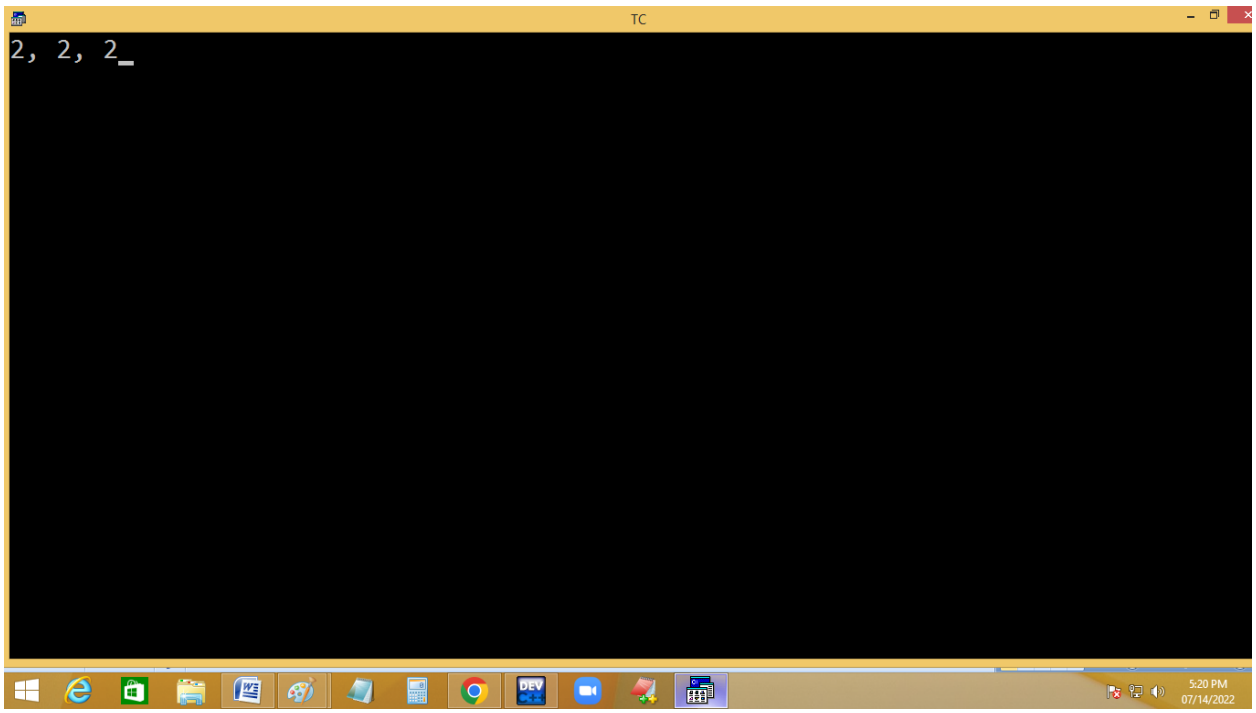

TC
File Edit Run Compile Project Options Debug Break/watch
Line 7 Col 21 Insert Indent Tab Fill Unindent * C:NONAME.C
#include<stdio.h>
#include<conio.h>
void main()
{
int a=100;
clrscr();
printf("a = %d", *&a);
getch();
}


TC
a = 100

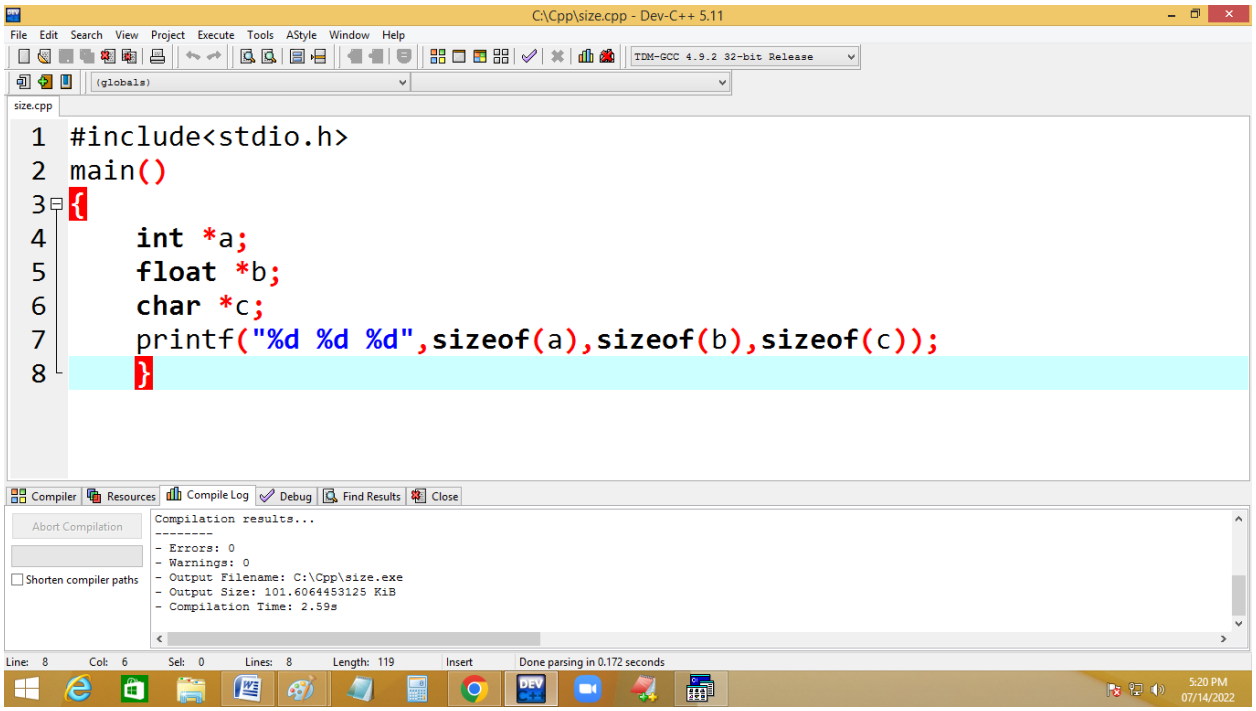
Eg. finding pointer size.

Pointer stores always the address and address is unsigned int. due to this pointer takes 2 / 4 / 8 bytes in 16 / 32 / 64 bit compilers.

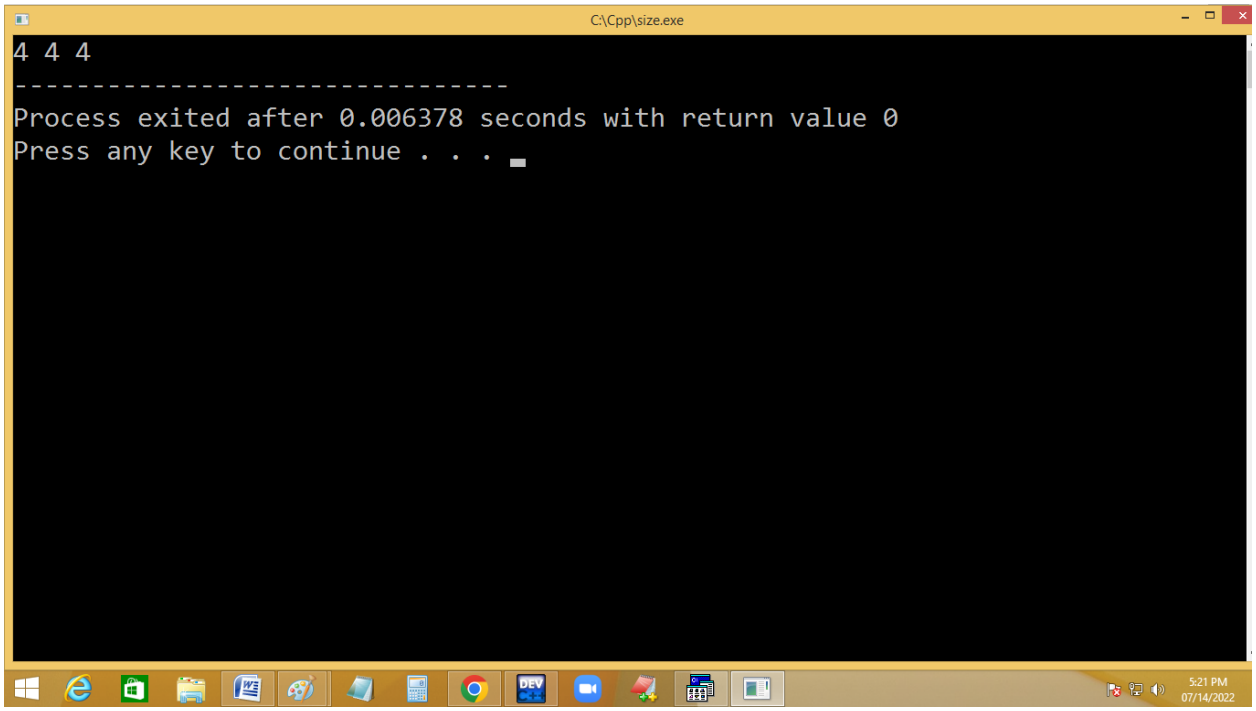

TC
File Edit Run Compile Project Options Debug Break/watch
Line 7 Col 15 Insert Indent Tab Fill Unindent * C:NONAME.C
#include<stdio.h>
#include<conio.h>
void main()
{
int *a; float *b; char *c;
clrscr();
printf("%d, %d, %d",sizeof(a),sizeof(b),sizeof(c));
getch();
}


TC
2, 2, 2

In dev c++:

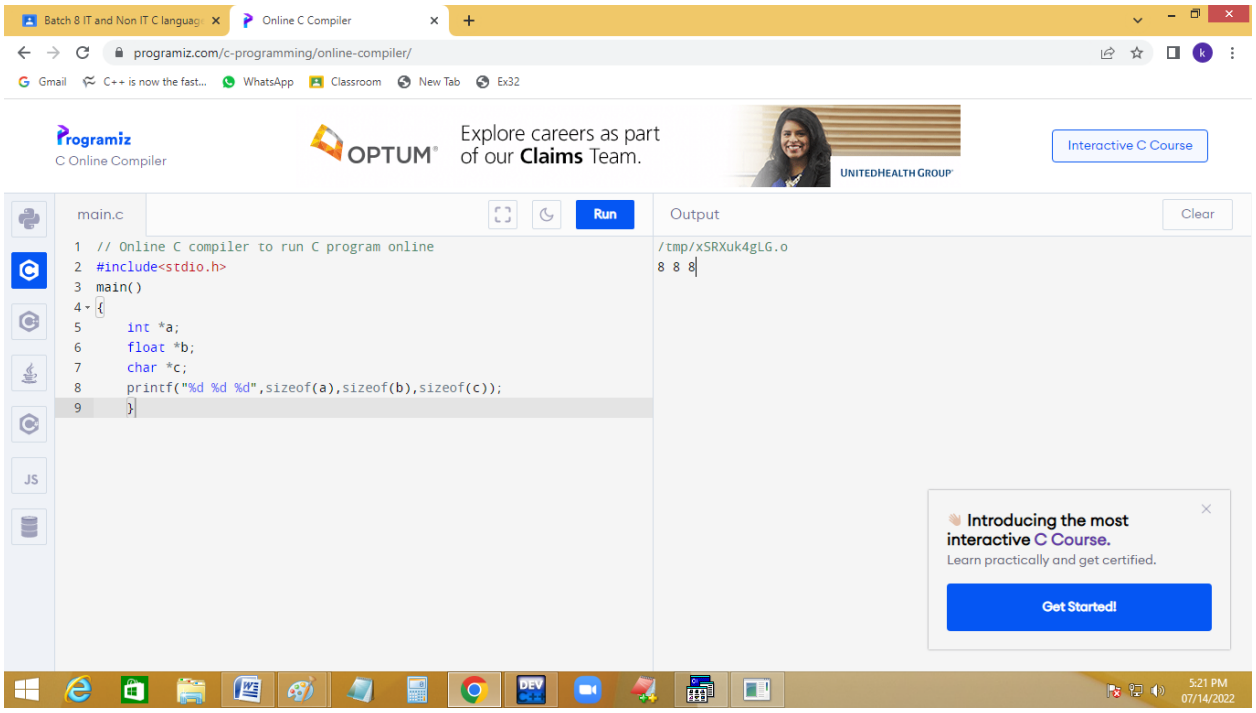

C:\Cpp\size.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 32-bit Release
(globals)
size.cpp
1 #include<stdio.h>
2 main()
3 {
4 int *a;
5 float *b;
6 char *c;
7 printf("%d %d %d",sizeof(a),sizeof(b),sizeof(c));
8 }
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Shorten compiler paths
Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Cpp\size.exe
- Output Size: 101.6064453125 KiB
- Compilation Time: 2.59s
Line: 8 Col: 6 Sel: 0 Lines: 8 Length: 119 Insert Done parsing in 0.172 seconds
5:20 PM
07/14/2022

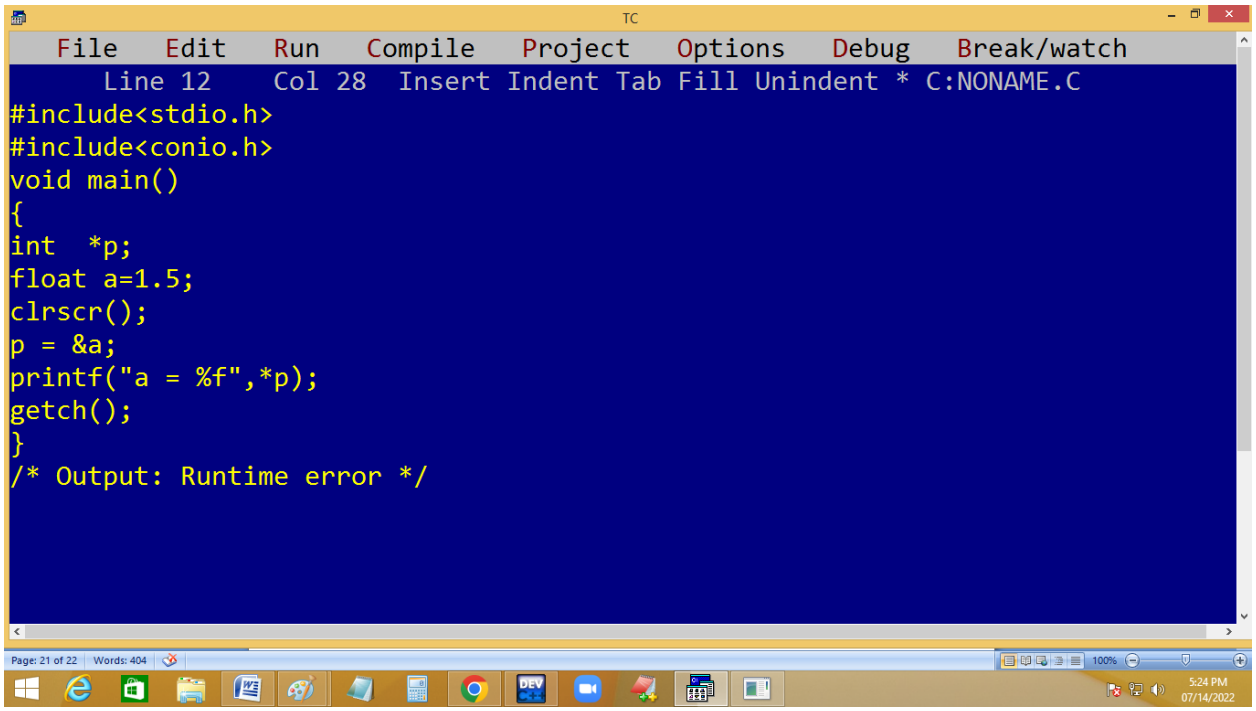

C:\Cpp\size.exe
4 4 4

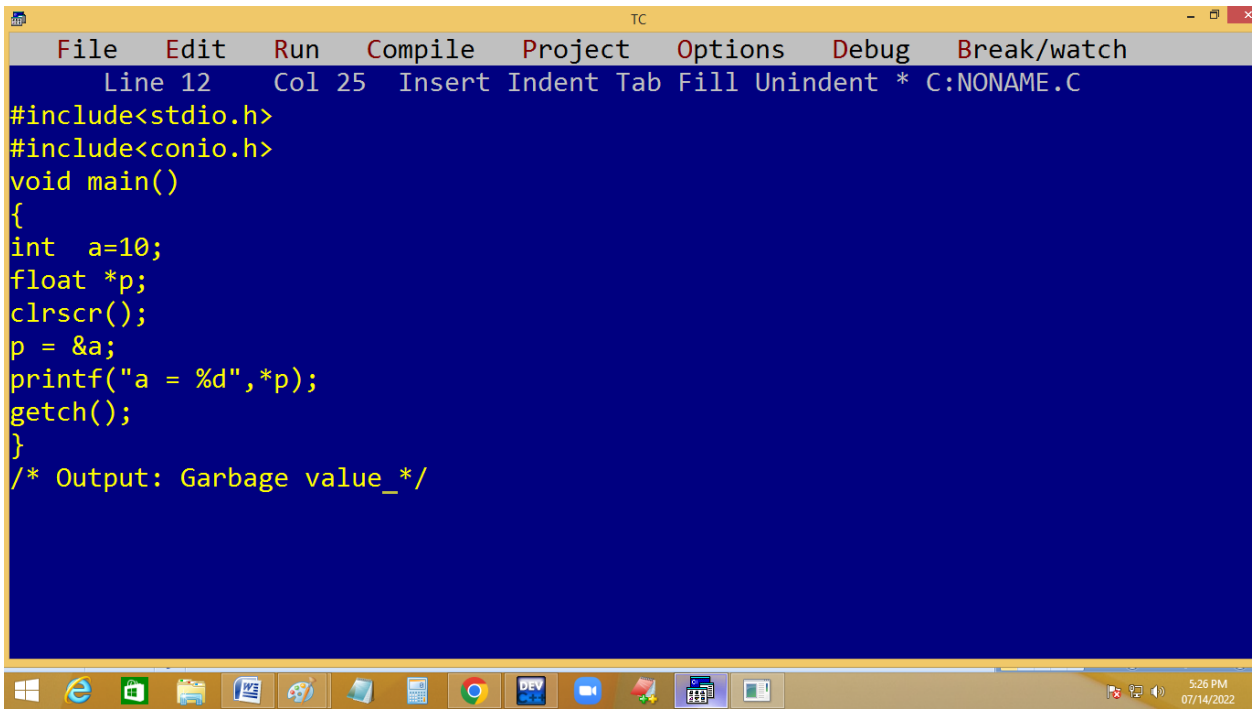
Process exited after 0.006378 seconds with return value 0
Press any key to continue . . .
5:21 PM
07/14/2022

online compiler:

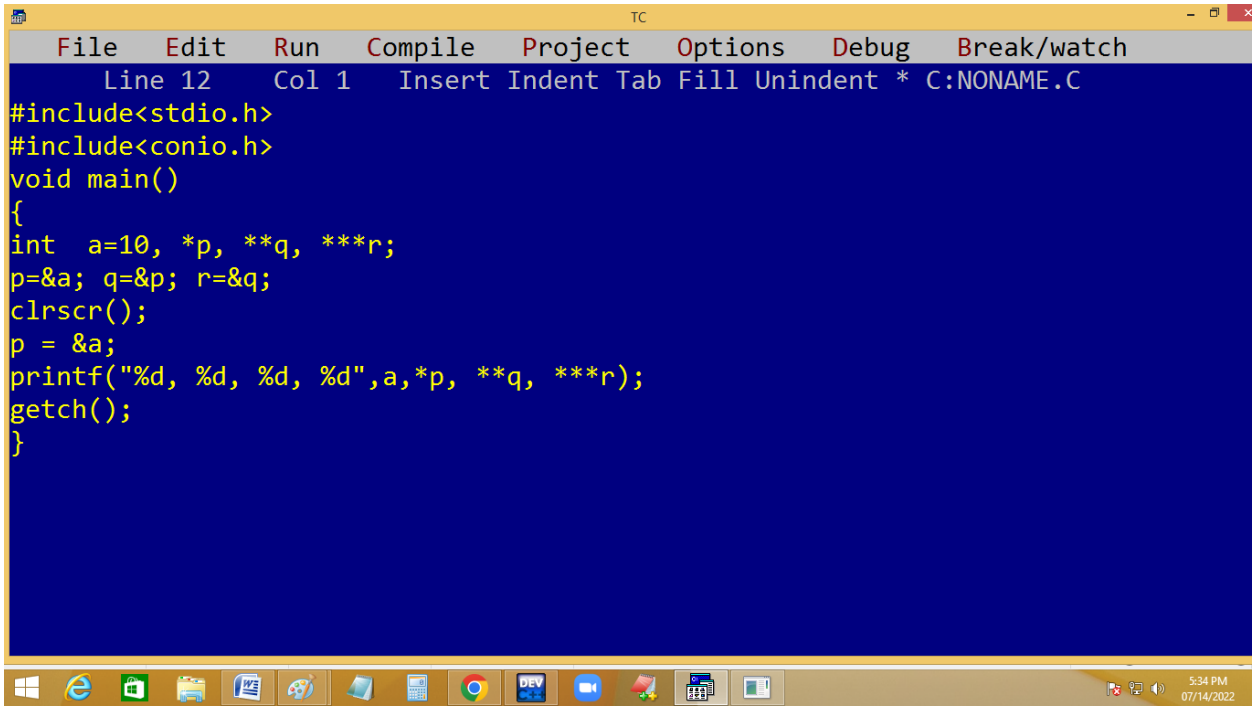


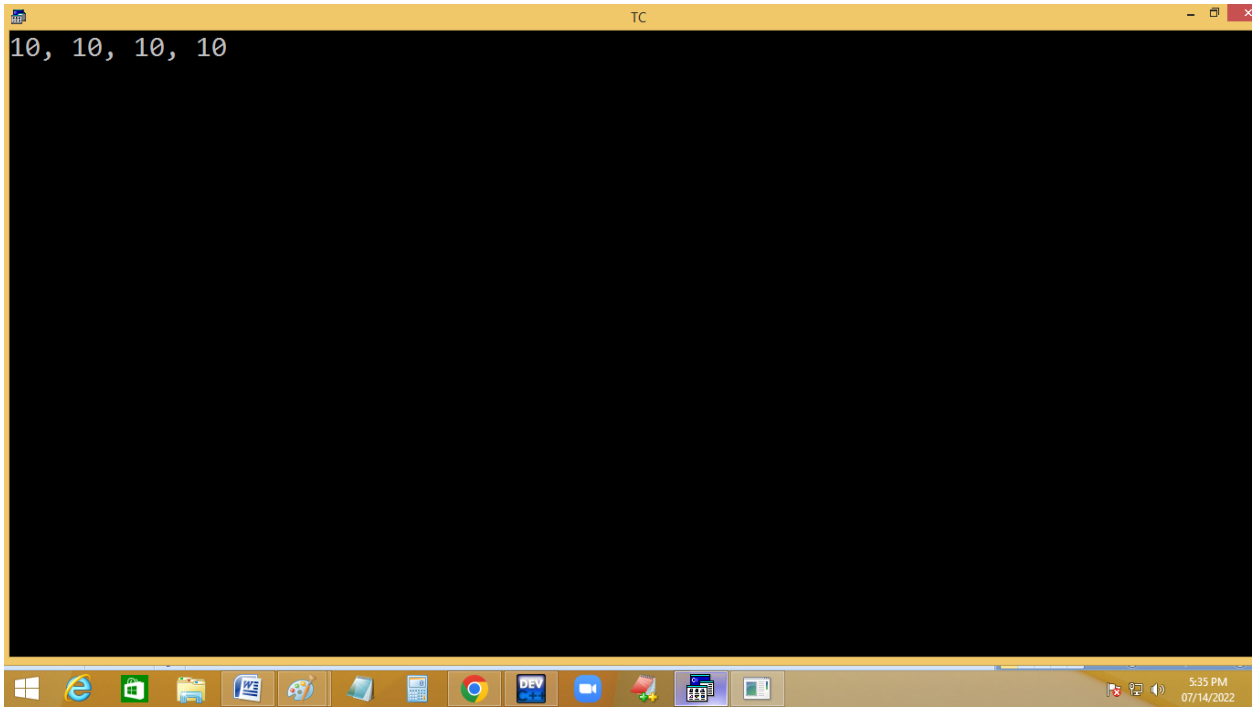
Pointer compatibility: **P**ointer stores address of same type of variable. When different type is given, it gives garbage or runtime error. They are used to handle dynamic multi dimensional array.

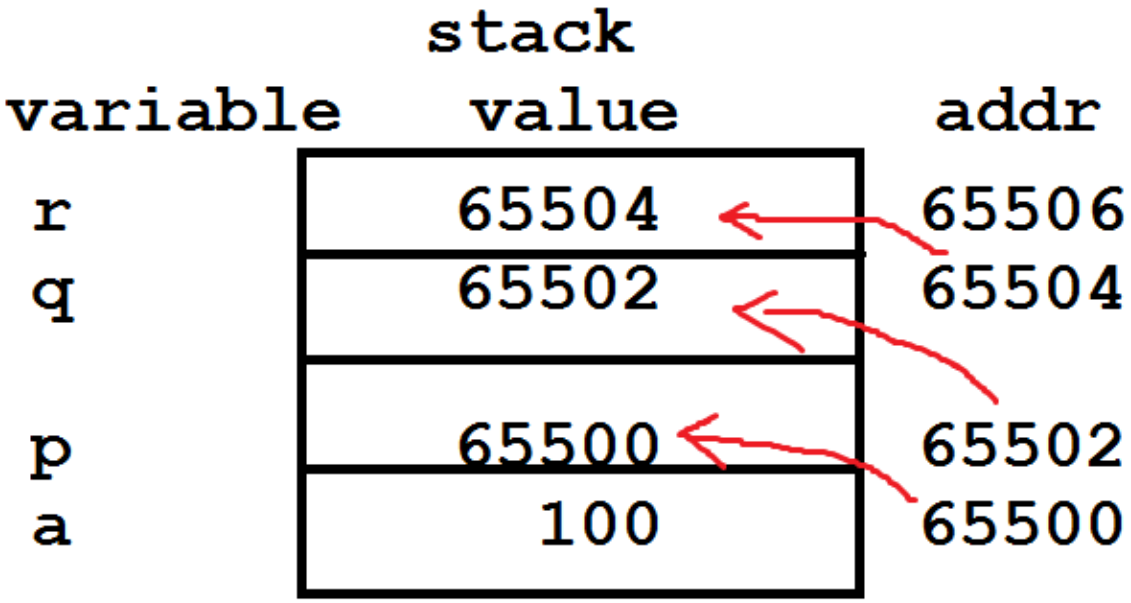

TC
File Edit Run Compile Project Options Debug Break/watch
Line 12 Col 28 Insert Indent Tab Fill Unindent * C:NONAME.C
#include<stdio.h>
#include<conio.h>
void main()
{
int *p;
float a=1.5;
clrscr();
p = &a;
printf("a = %f",*p);
getch();
}
/* Output: Runtime error */

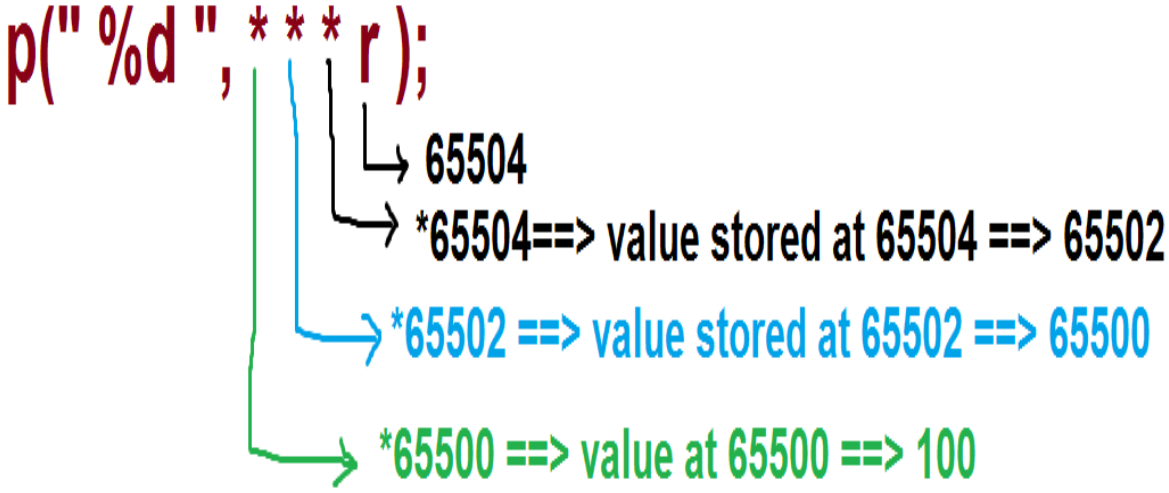

TC
File Edit Run Compile Project Options Debug Break/watch
Line 12 Col 25 Insert Indent Tab Fill Unindent * C:NONAME.C
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10;
float *p;
clrscr();
p = &a;
printf("a = %d",*p);
getch();
}
/* Output: Garbage value_*/

Pointer to pointer / double pointer:





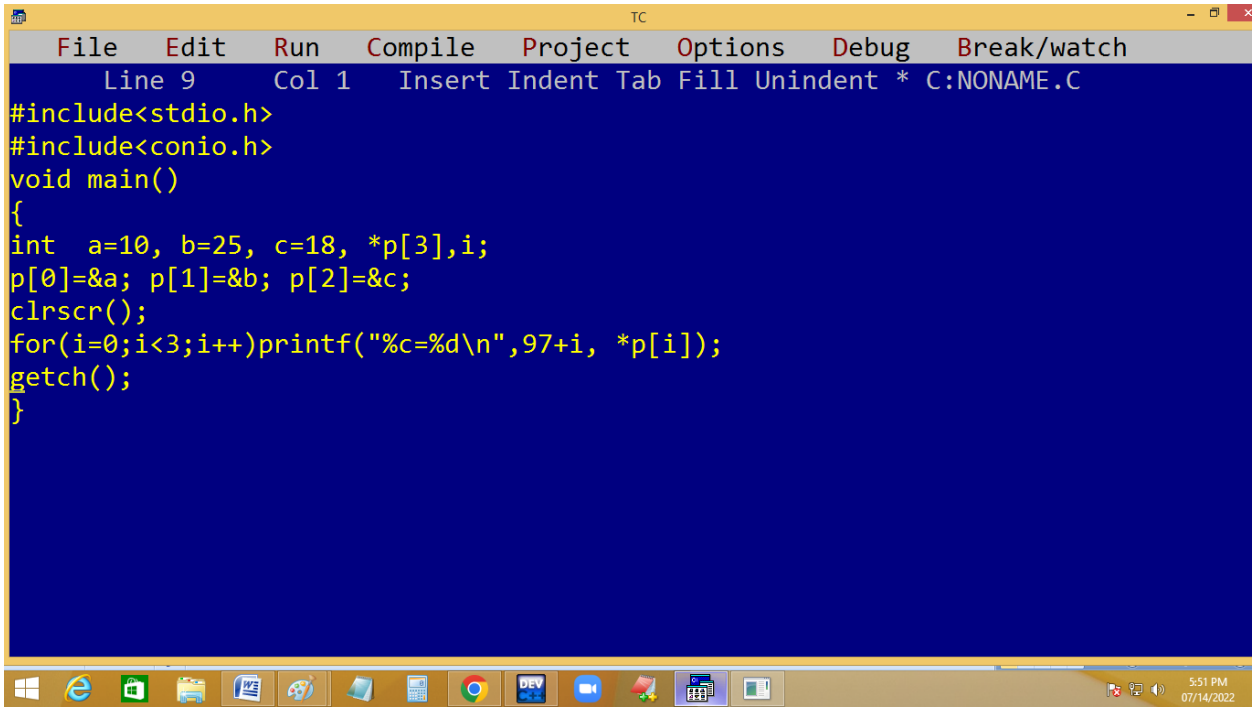


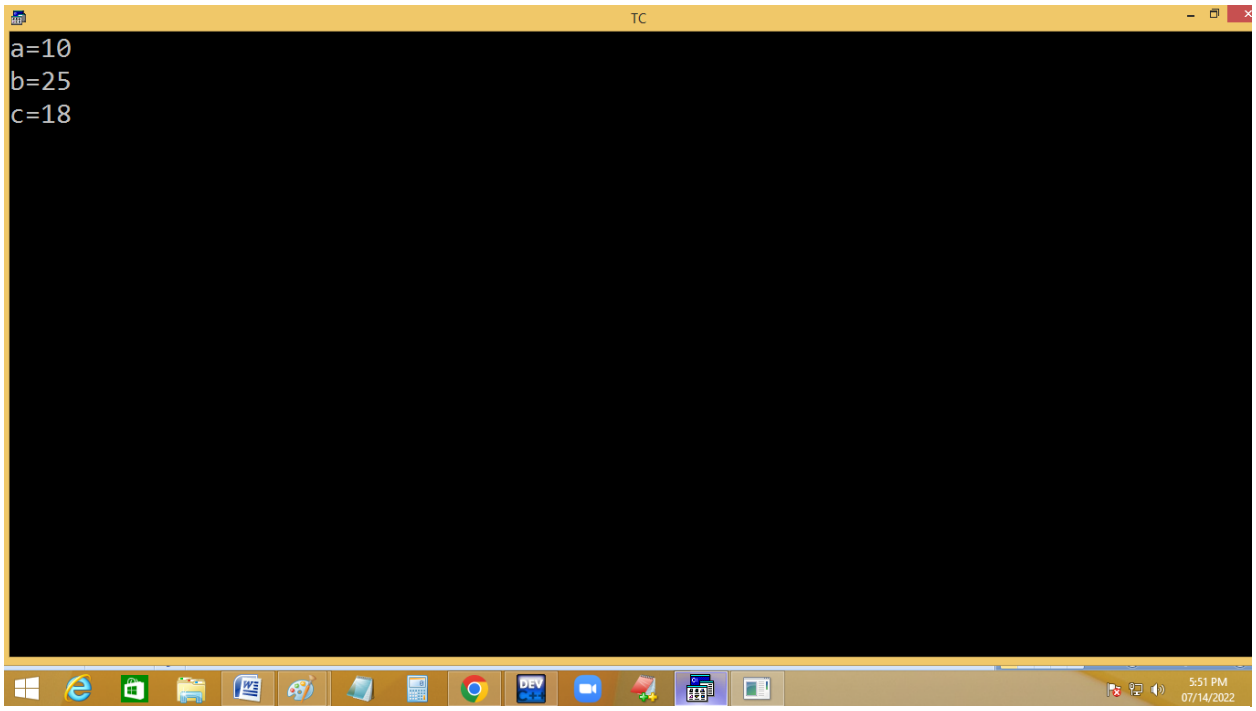


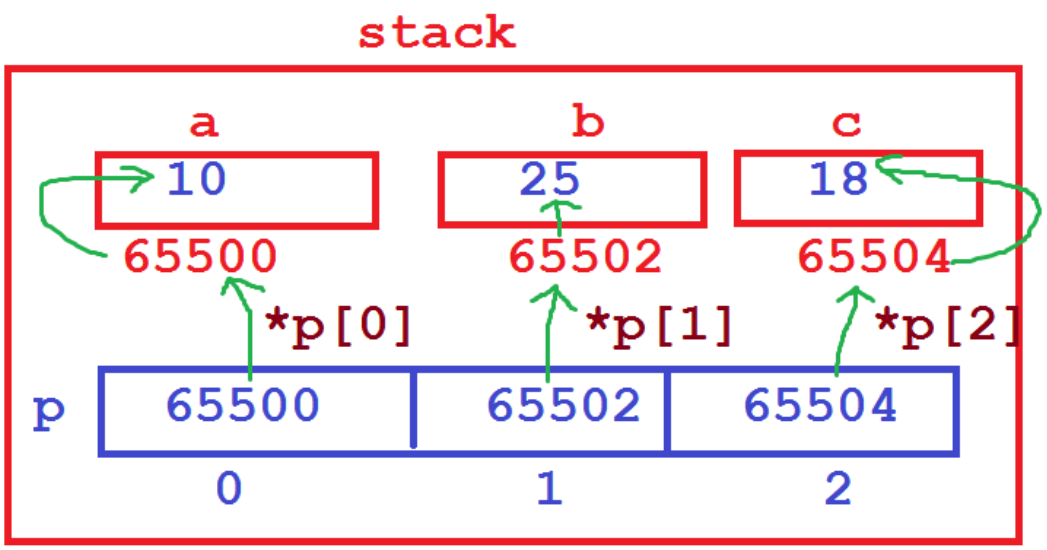
Array of pointer:

Like general variables we can declare the pointer using array. Due to this

we can store several address in one pointer variable. They are used to control dynamic array.




TC
a=10
b=25
c=18


stack
a
b
c
10
25
18
65500
65502
65504
*p[0]
*p[1]
*p[2]
p
65500
65502
65504
0
1
2