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 local variable outside the function i.e. data sharing between functions.** [**call by address/Reference**].
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.

Dis-advantage:

They are not secured.

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.

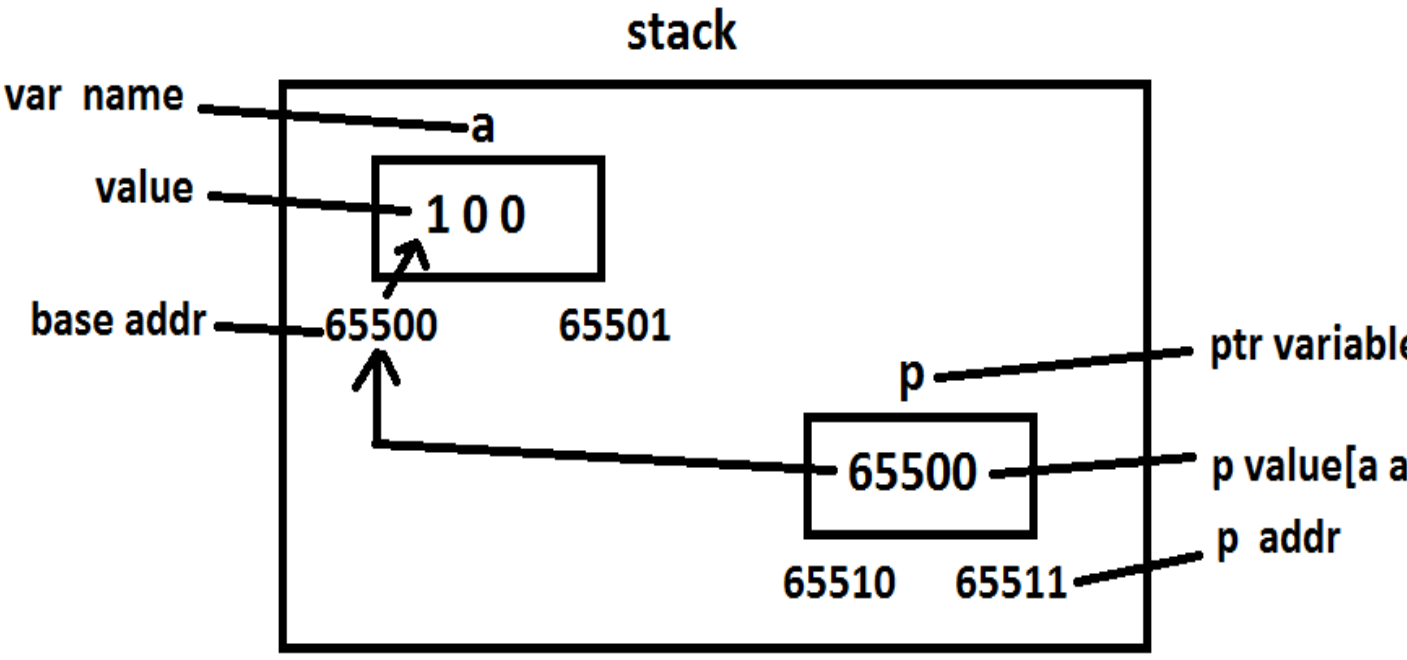
Eg:

```
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.

```
printf( "%d", *p );
```

→ **100**

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.

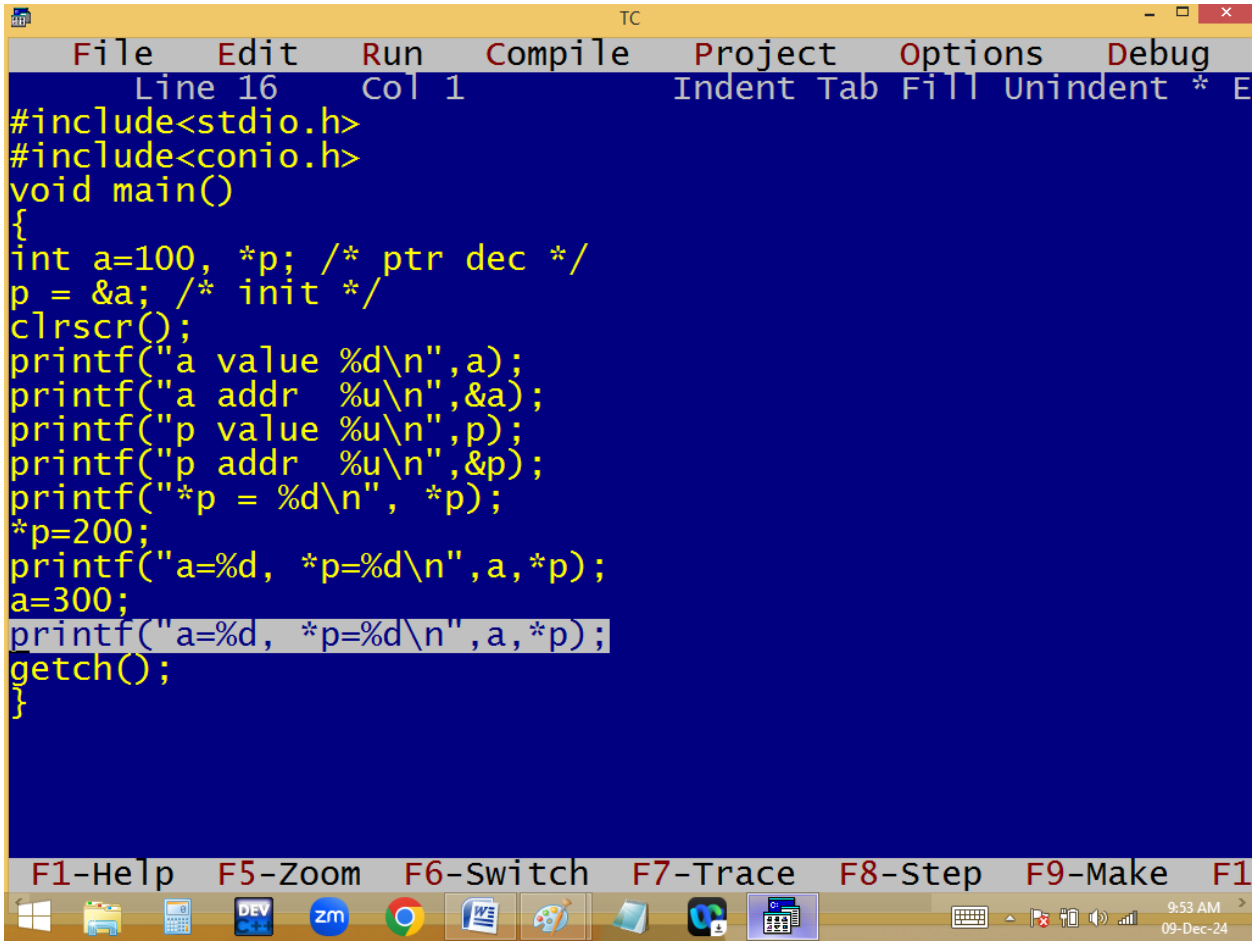
Due to this example any changes conducted in ***p** effects the value of **'a'**. Hence **p** is called **pointer** to **a**.

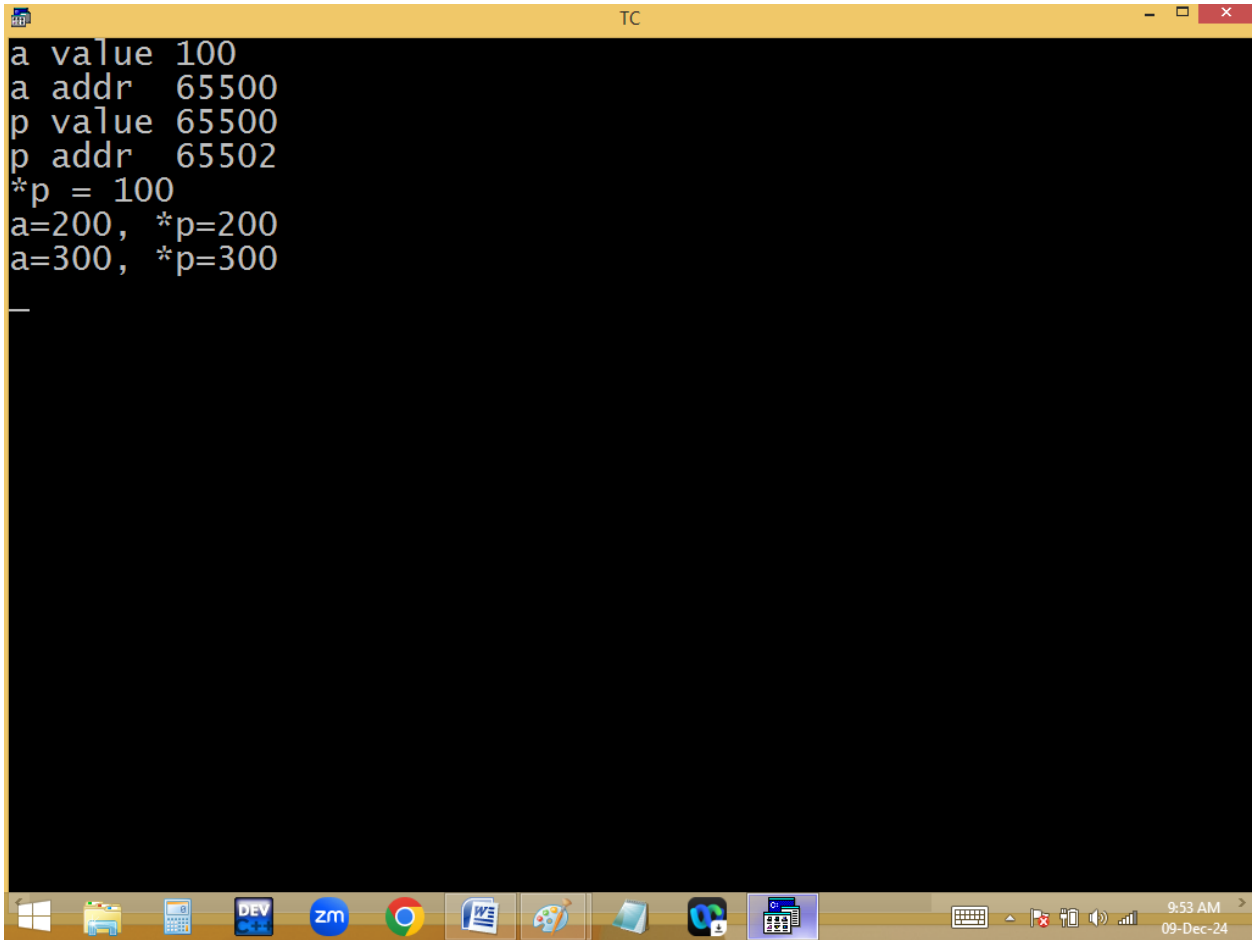
Eg: *p=200;

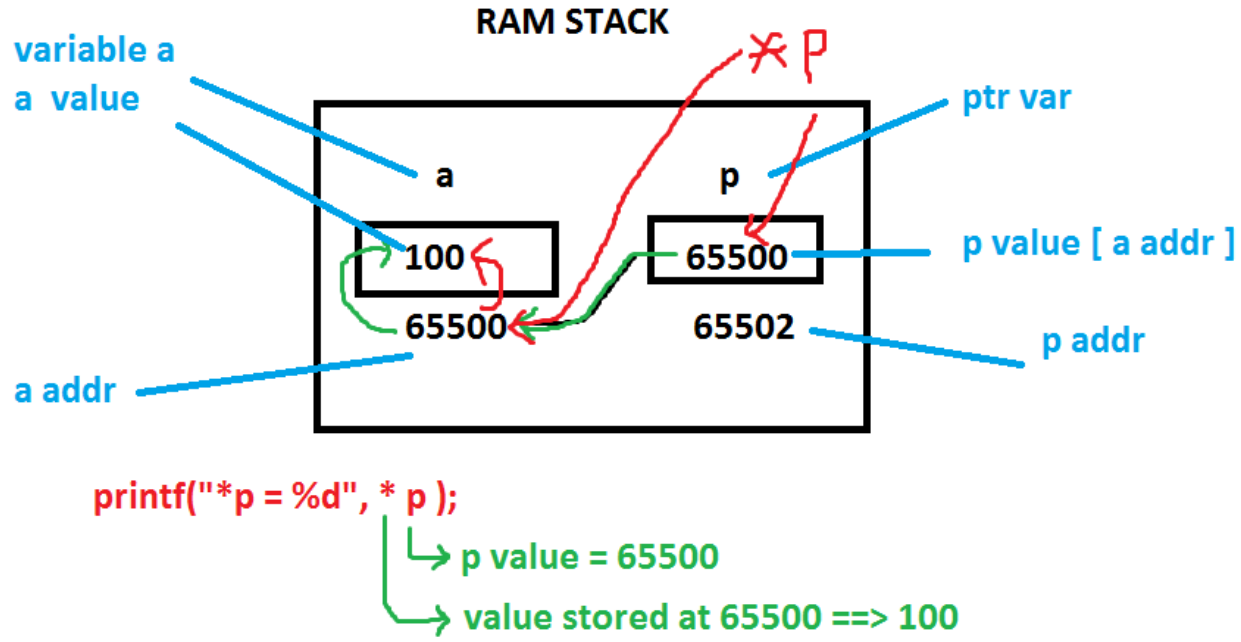
Now **a** value becomes **200**.

Eg:

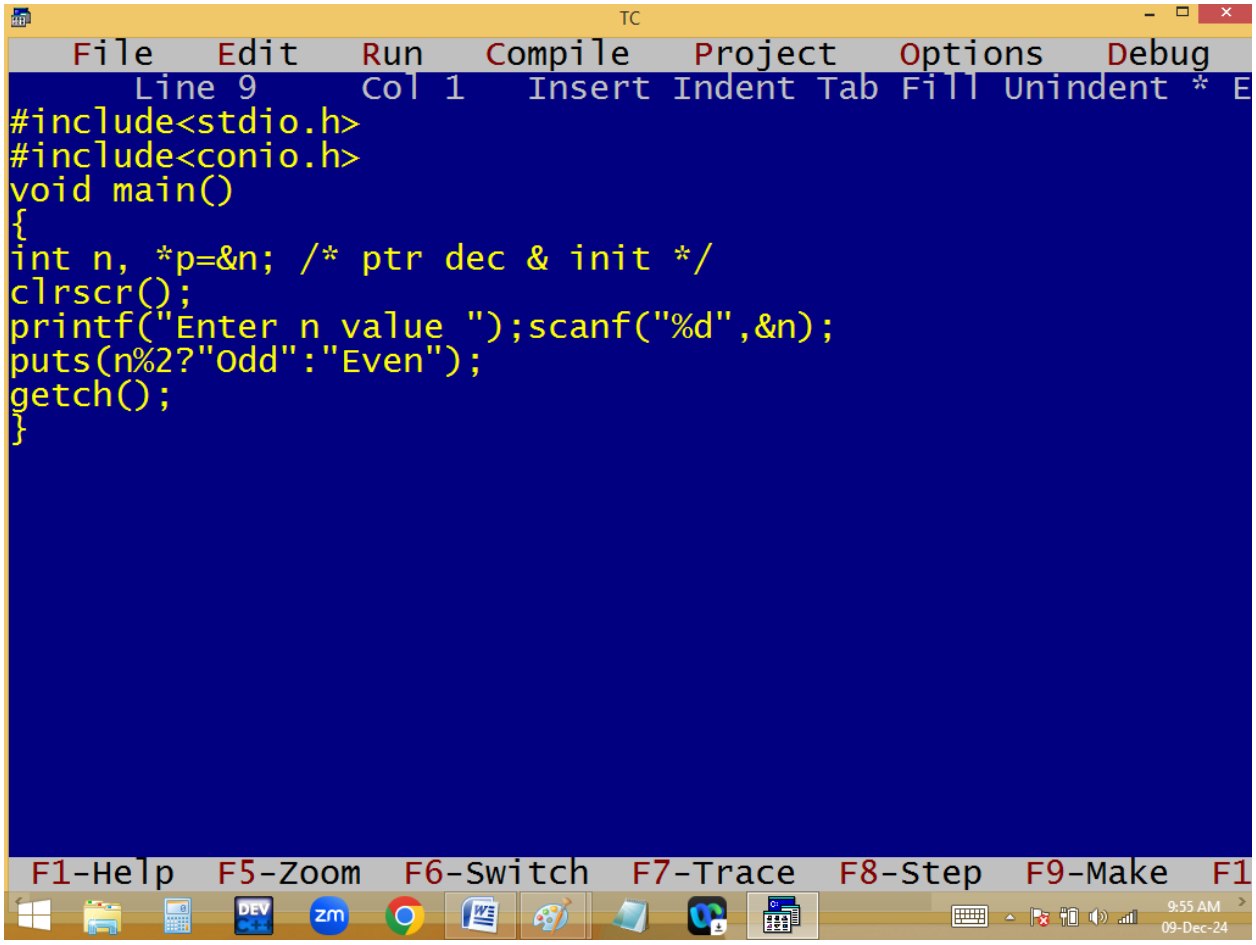
Finding a variable value and address using a pointer:

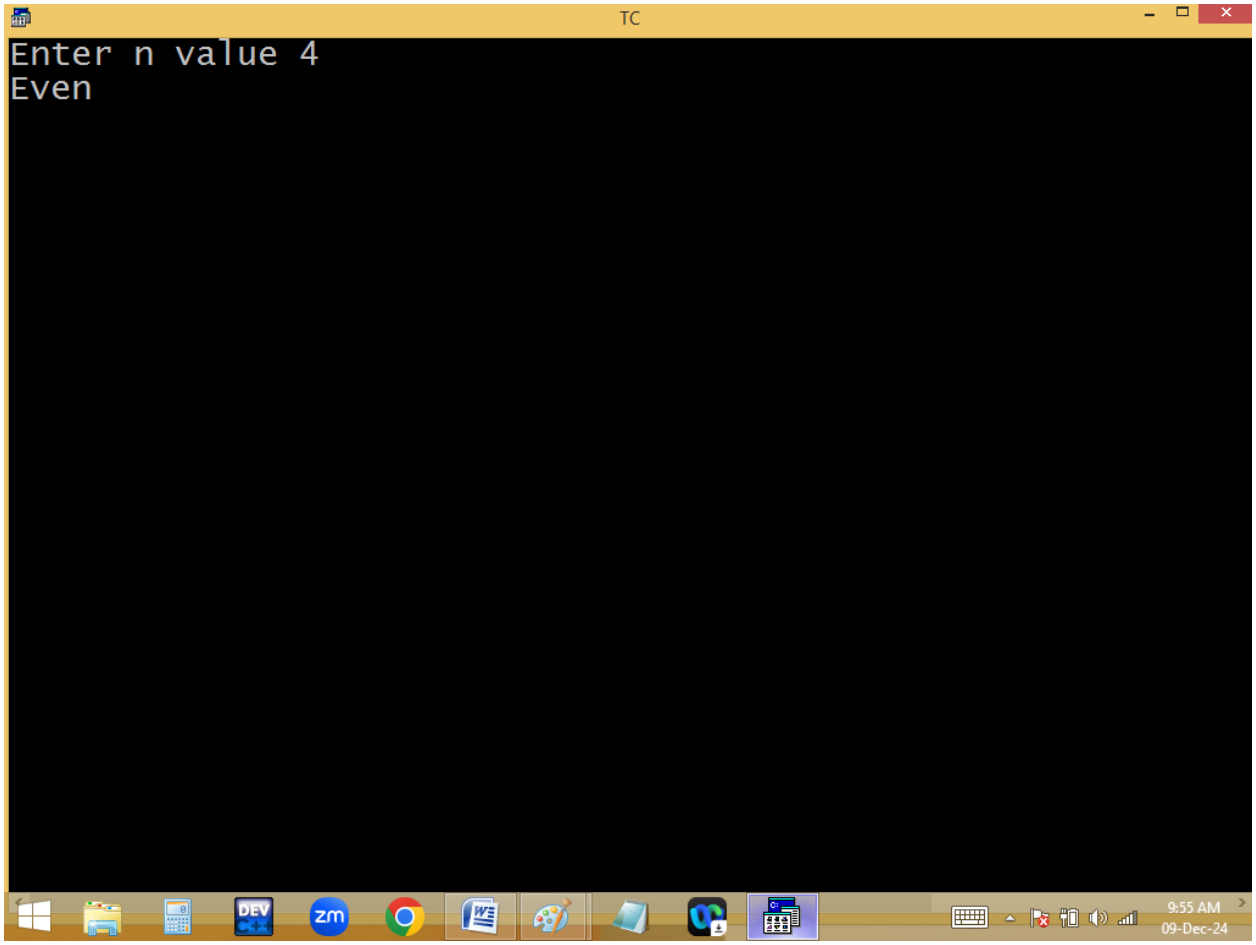

TC
File Edit Run Compile Project Options Debug
Line 16 Col 1 Indent Tab Fill Unindent * E
#include<stdio.h>
#include<conio.h>
void main()
{
int a=100, *p; /* ptr dec */
p = &a; /* init */
clrscr();
printf("a value %d\n",a);
printf("a addr %u\n",&a);
printf("p value %u\n",p);
printf("p addr %u\n",&p);
printf("*p = %d\n", *p);
*p=200;
printf("a=%d, *p=%d\n",a,*p);
a=300;
printf("a=%d, *p=%d\n",a,*p);
getch();
}
F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F1
9:53 AM
09-Dec-24

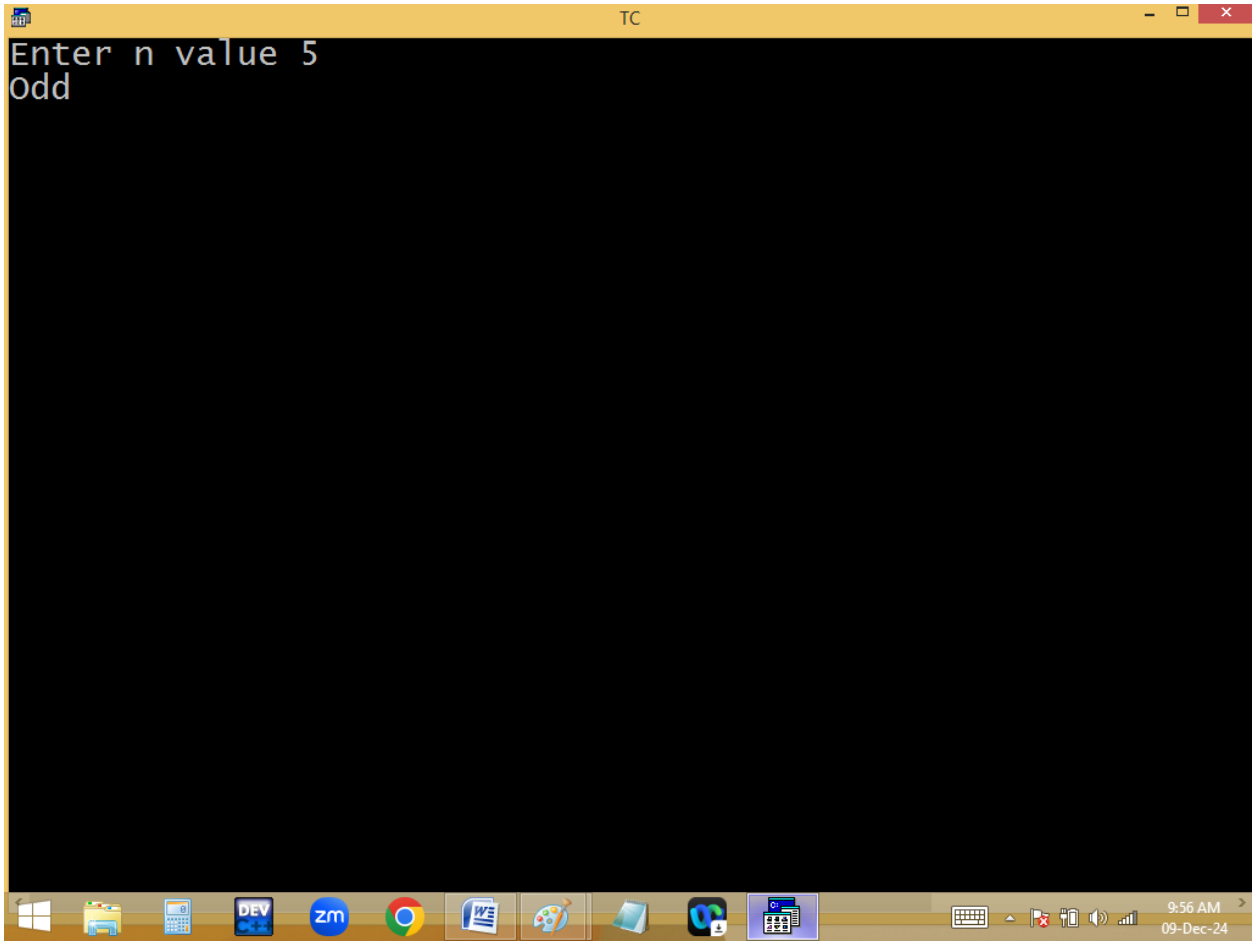

TC
a value 100
a addr 65500
p value 65500
p addr 65502
*p = 100
a=200, *p=200
a=300, *p=300
9:53 AM
09-Dec-24

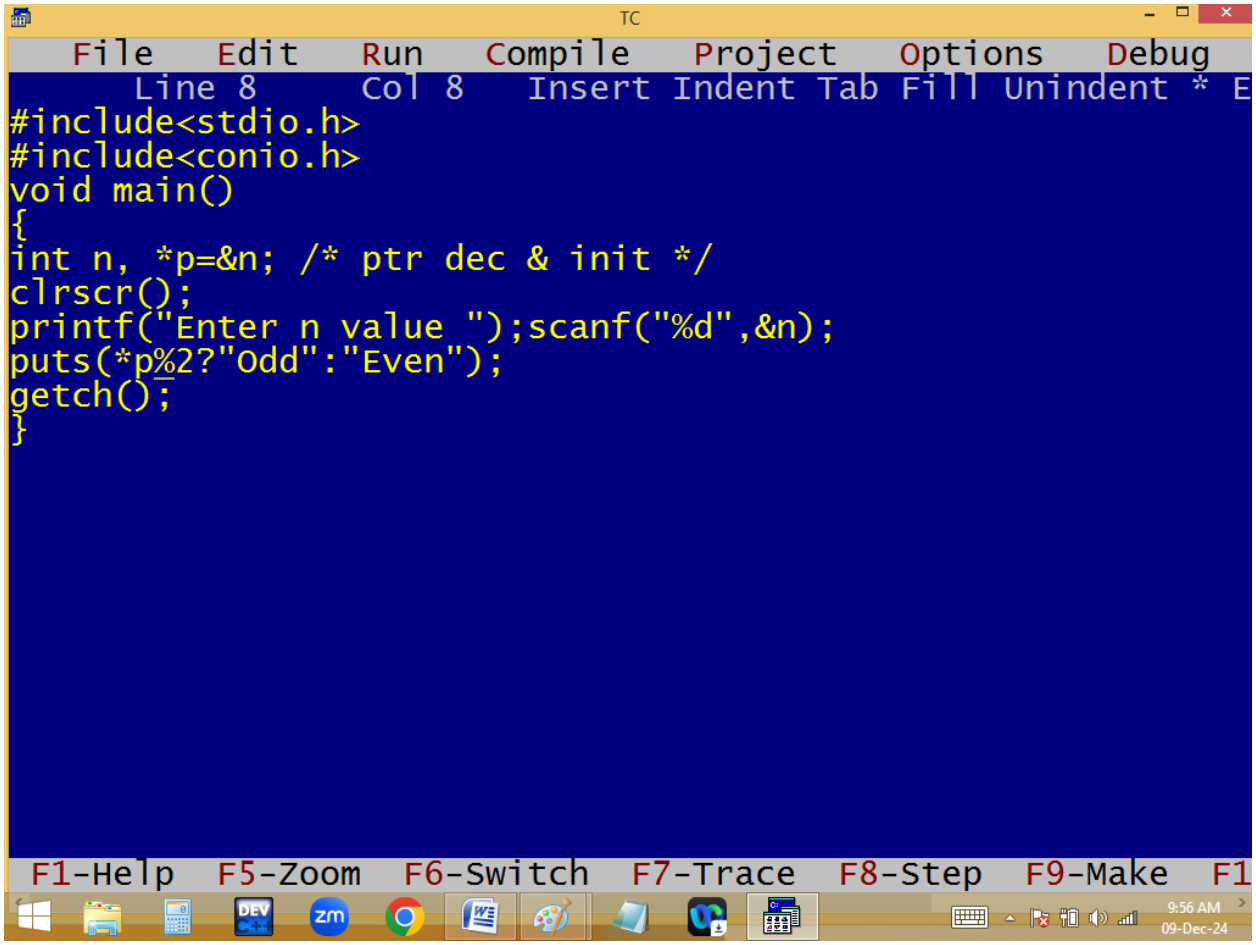

RAM STACK
variable a
a value
a addr
ptr var
p value [a addr]
p addr
a
p
100
65500
65500
65502
* p
printf("*p = %d", * p);
p value = 65500
value stored at 65500 ==> 100

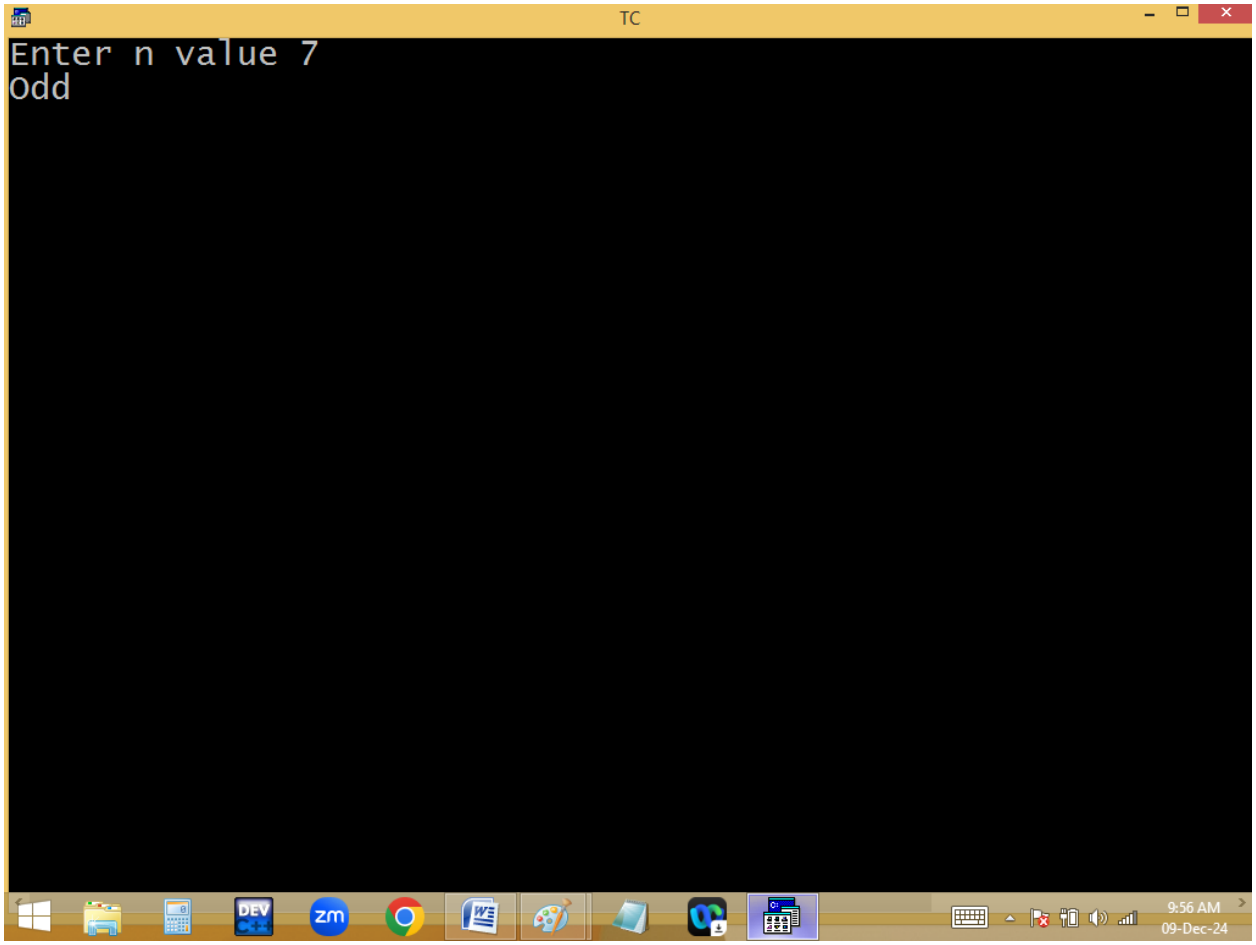
Finding even/odd using pointer:

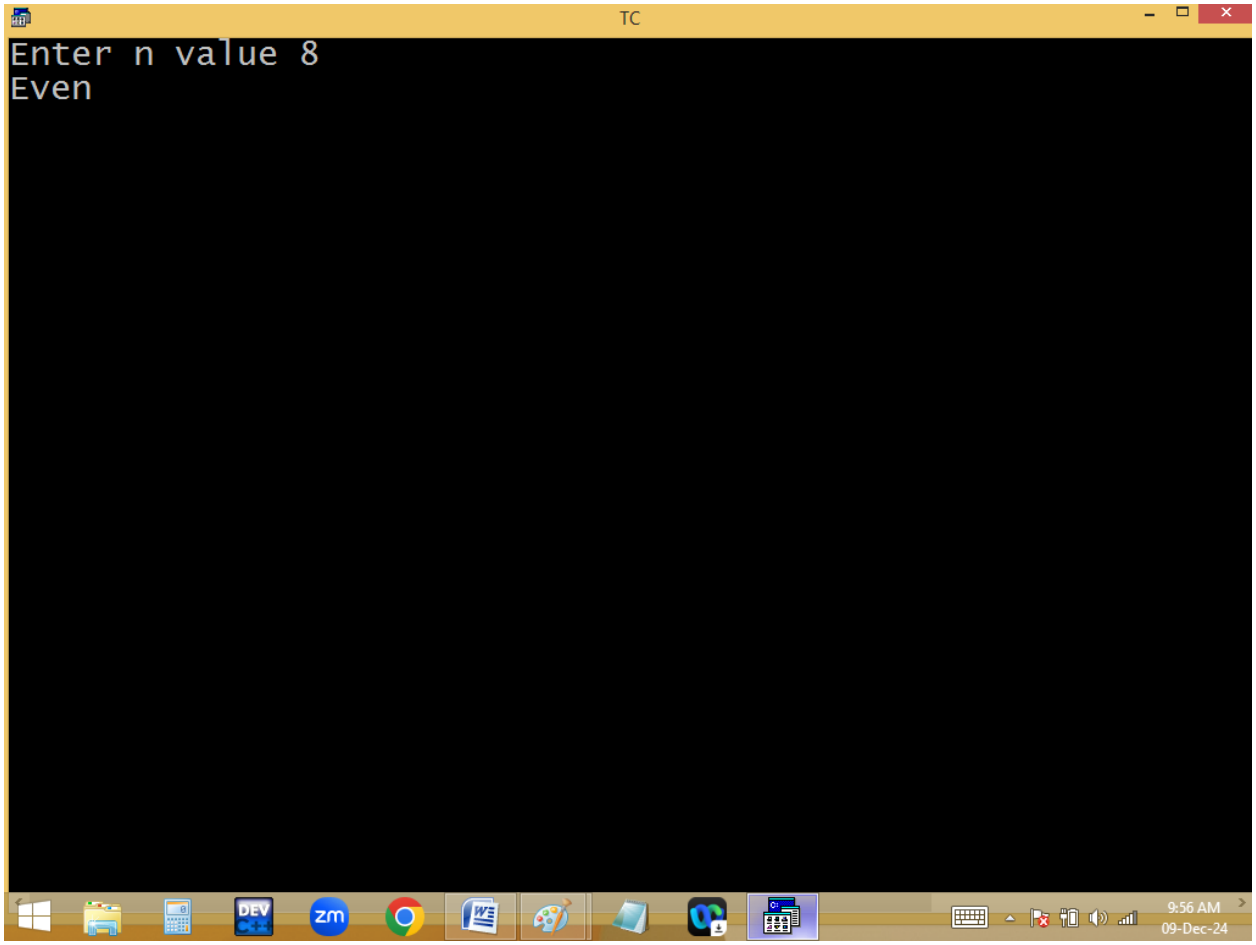



TC
Enter n value 4
Even
9:55 AM
09-Dec-24

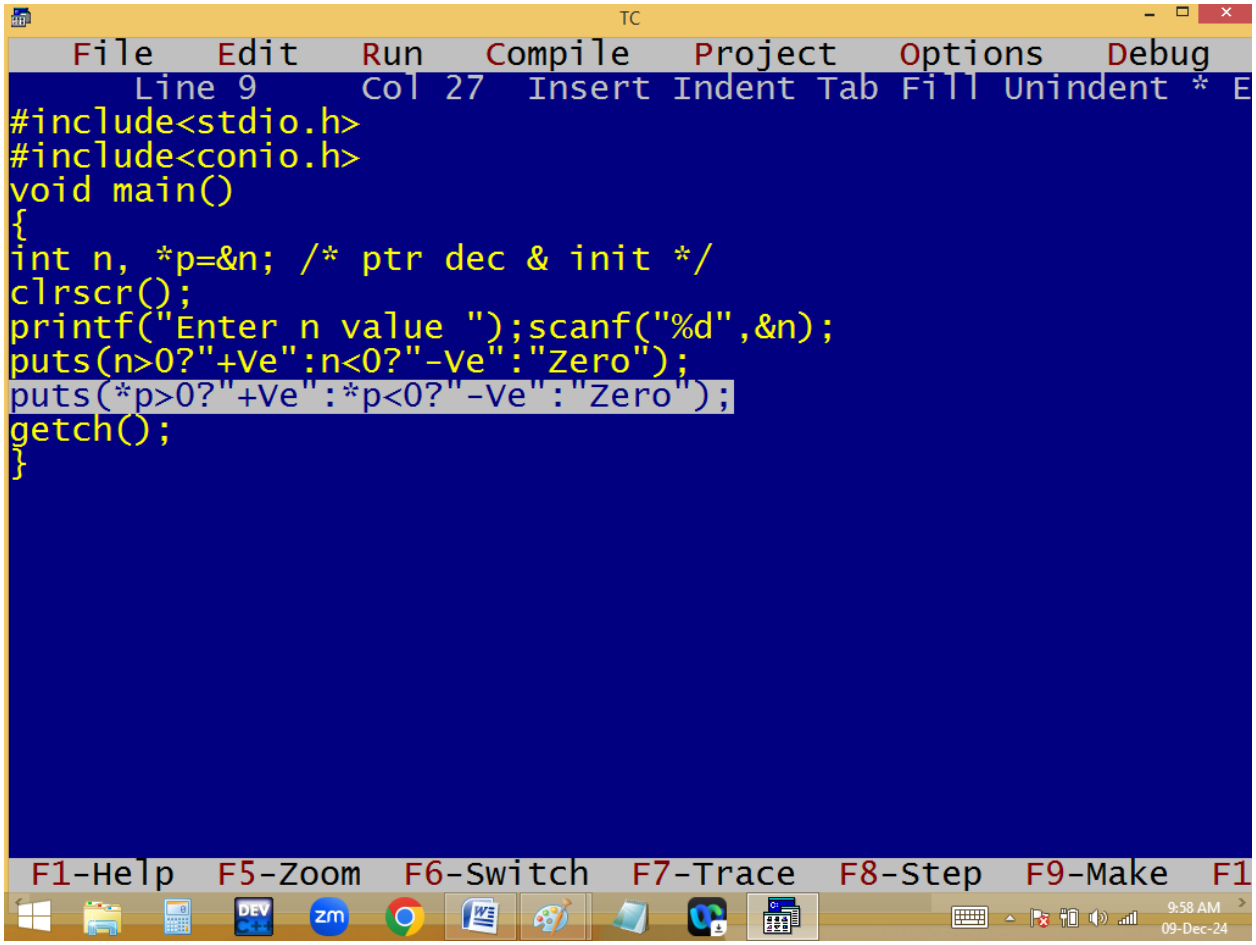

TC
Enter n value 5
Odd
9:56 AM
09-Dec-24

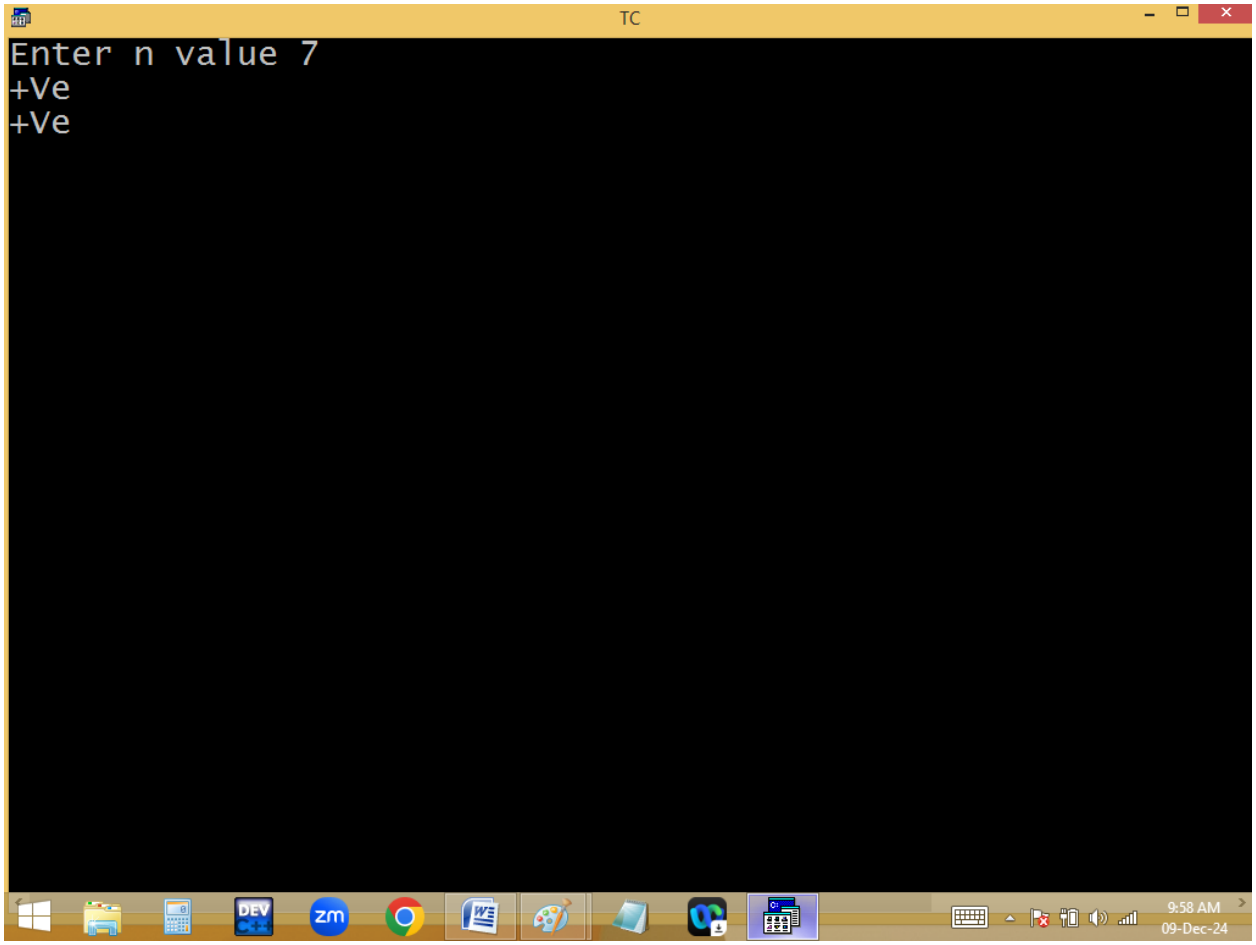

TC
File Edit Run Compile Project Options Debug
Line 8 Col 8 Insert Indent Tab Fill Unindent * E
#include<stdio.h>
#include<conio.h>
void main()
{
int n, *p=&n; /* ptr dec & init */
clrscr();
printf("Enter n value ");scanf("%d",&n);
puts(*p%2?"Odd":"Even");
getch();
}
F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F1
9:56 AM
09-Dec-24

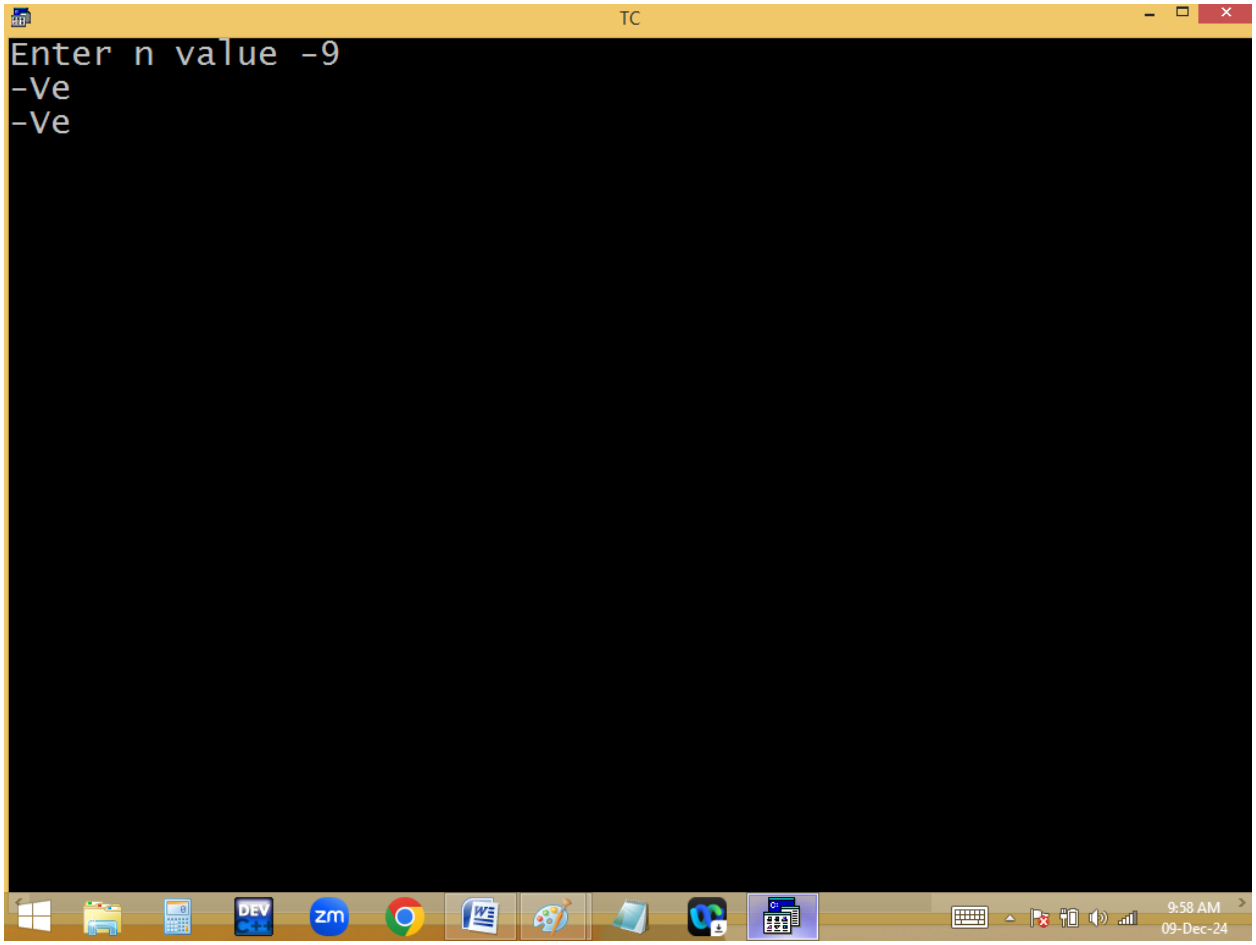

TC
Enter n value 7
Odd
9:56 AM
09-Dec-24

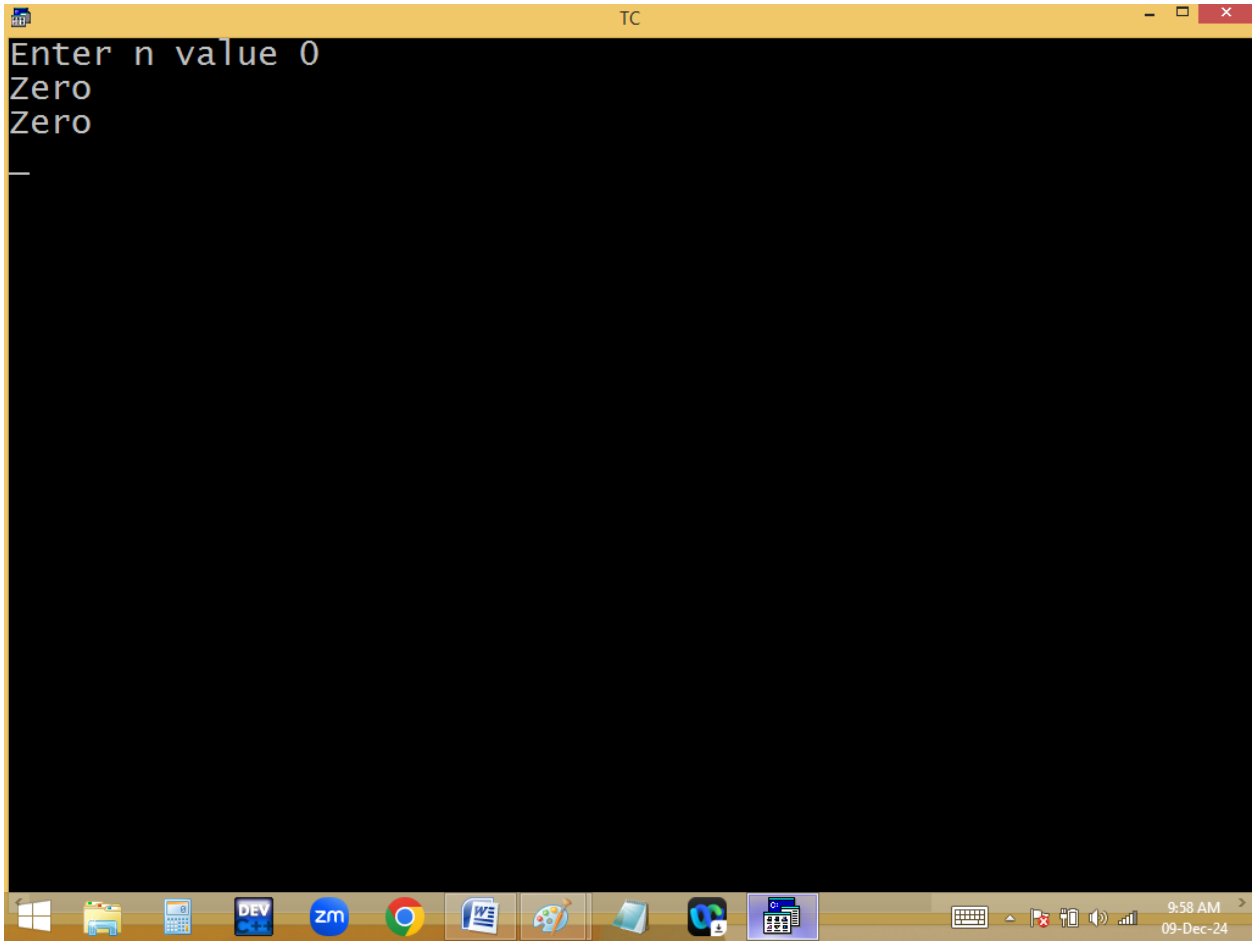


Finding +Ve/-Ve/0 using pointer:

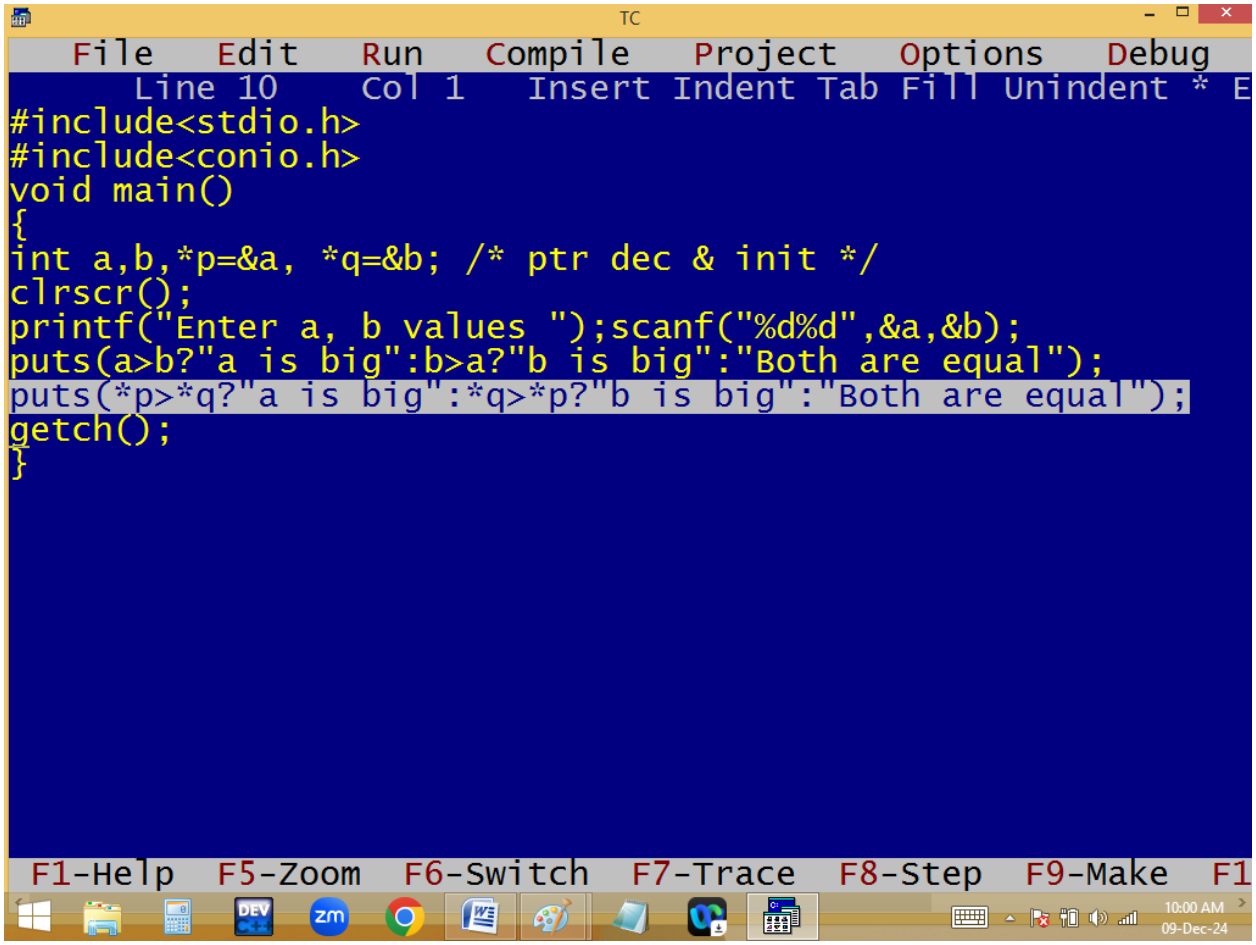

TC
File Edit Run Compile Project Options Debug
Line 9 Col 27 Insert Indent Tab Fill Unindent * E
#include<stdio.h>
#include<conio.h>
void main()
{
int n, *p=&n; /* ptr dec & init */
clrscr();
printf("Enter n value ");scanf("%d",&n);
puts(n>0?"+Ve":n<0?"-Ve":"Zero");
puts(*p>0?"+Ve":*p<0?"-Ve":"Zero");
getch();
}
F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F1
9:58 AM
09-Dec-24

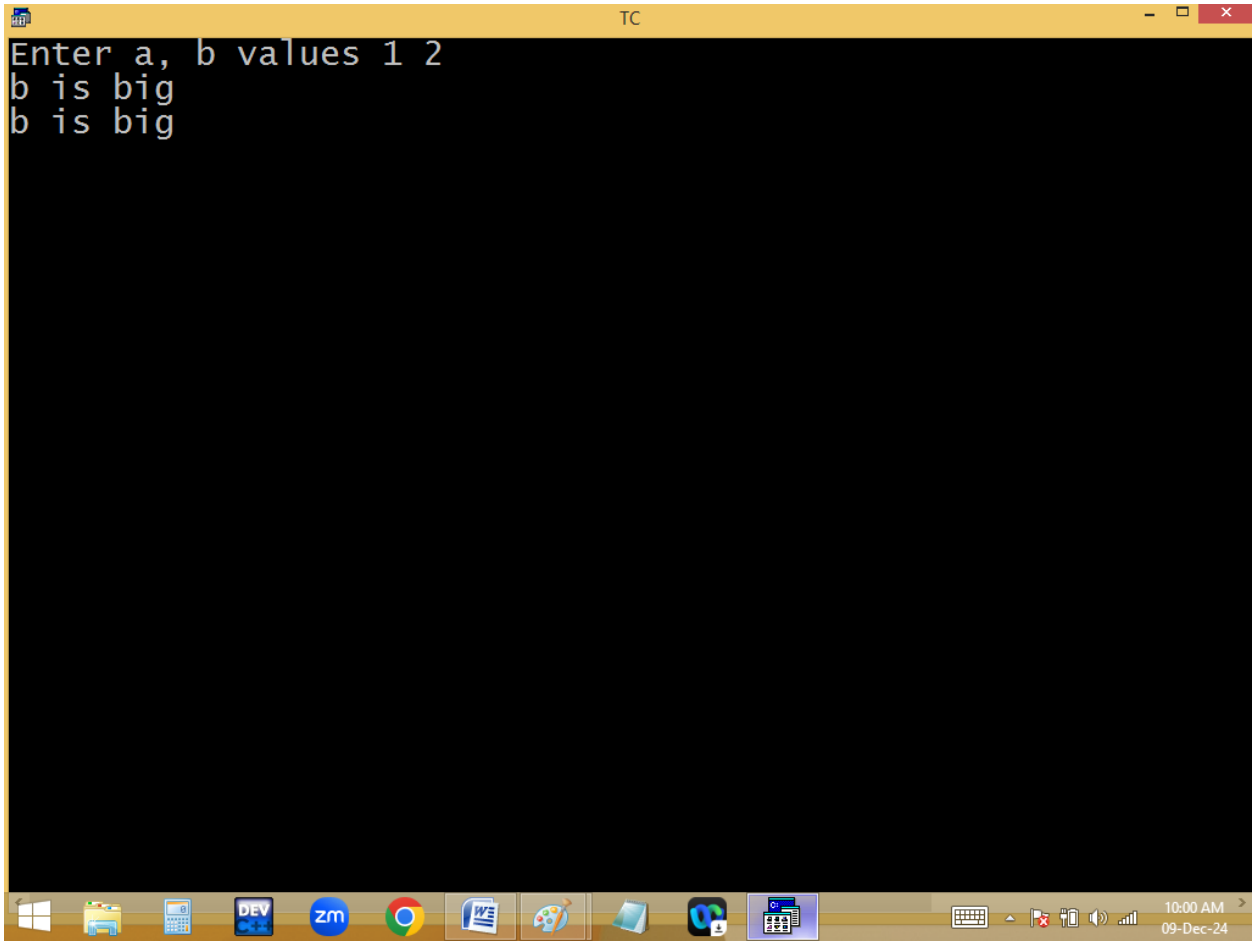

TC
Enter n value 7
+Ve
+Ve
9:58 AM
09-Dec-24

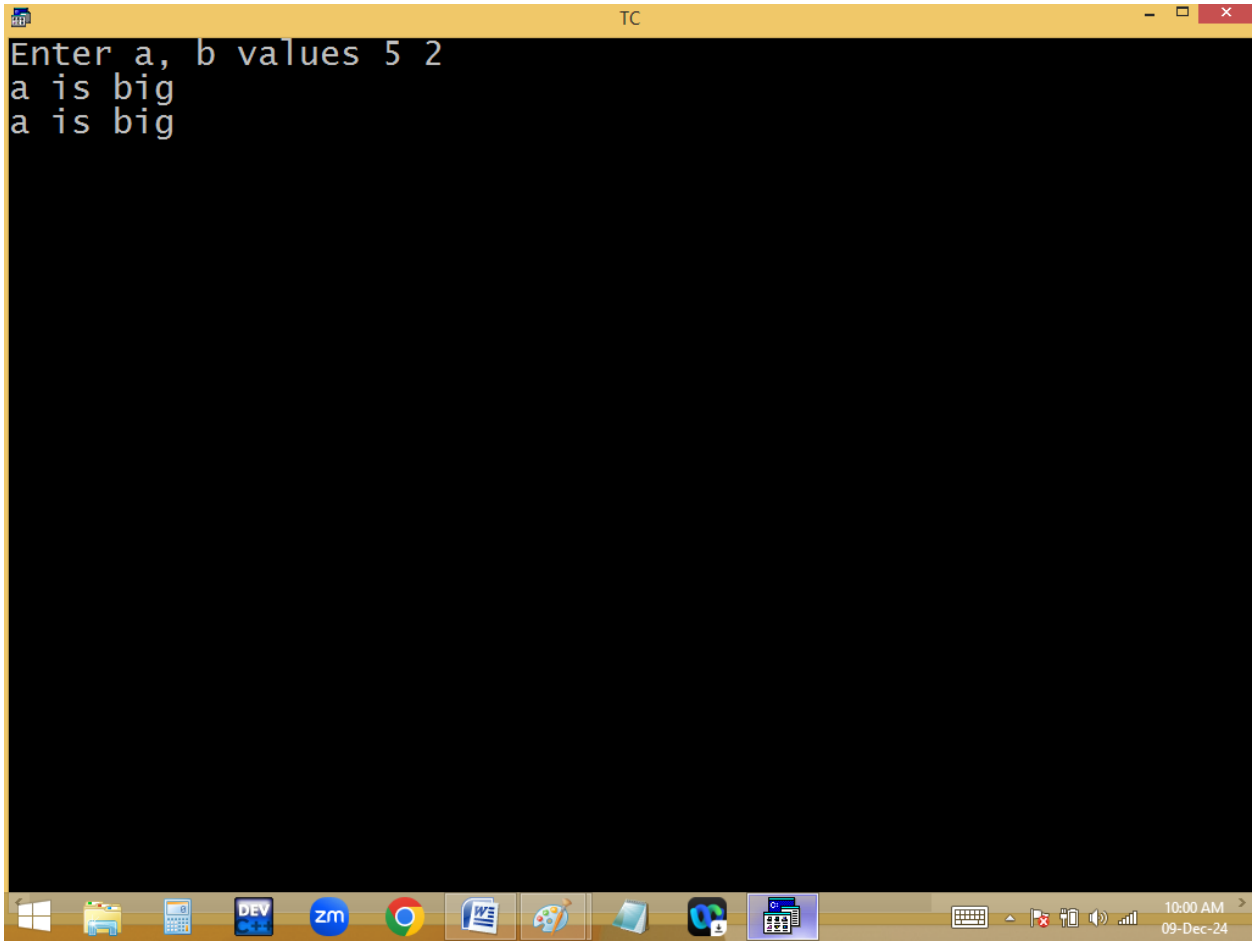

TC
Enter n value -9
-Ve
-Ve
9:58 AM
09-Dec-24

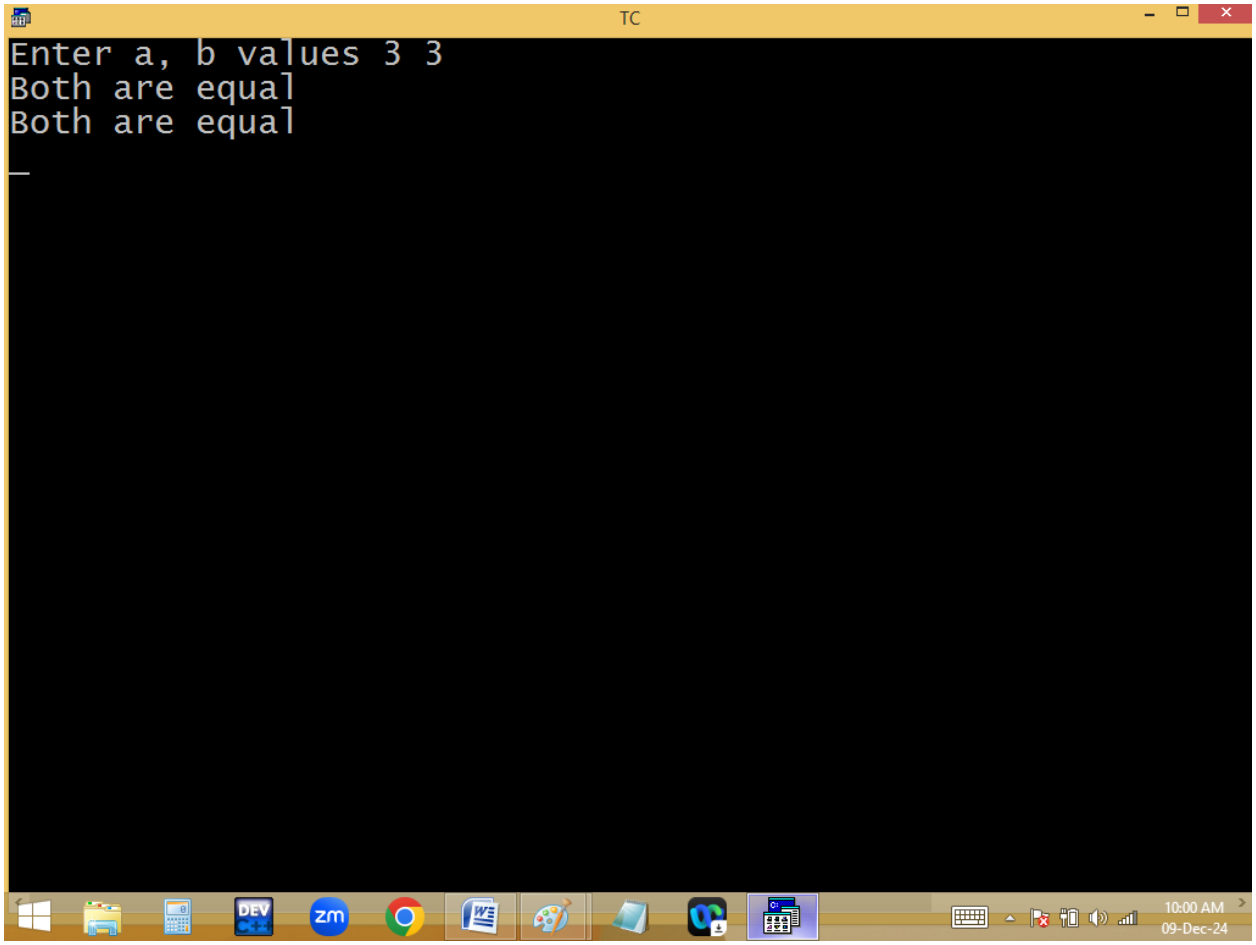


Finding max in 2 no's using pointer:

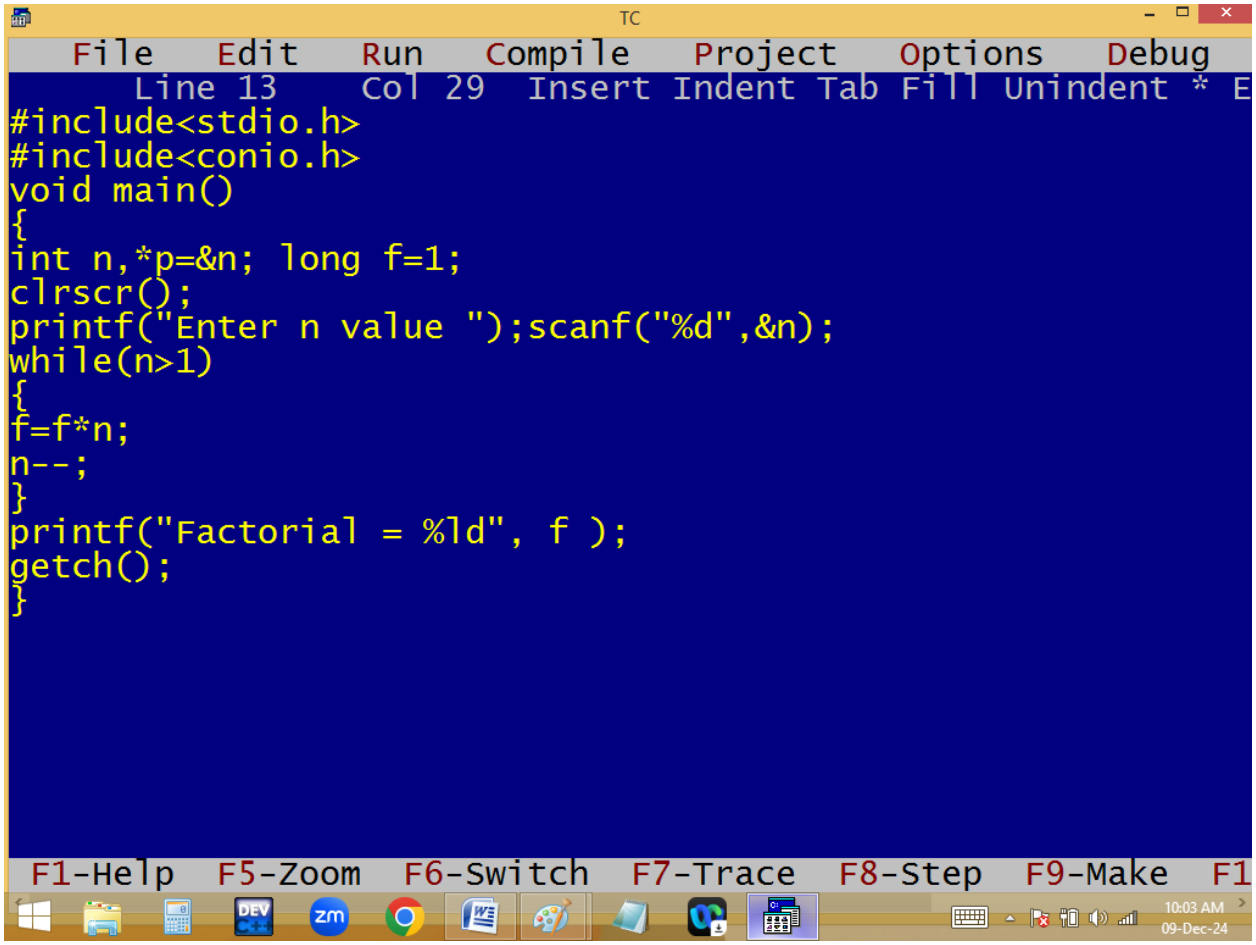

TC
File Edit Run Compile Project Options Debug
Line 10 Col 1 Insert Indent Tab Fill Unindent * E
#include<stdio.h>
#include<conio.h>
void main()
{
int a,b,*p=&a, *q=&b; /* ptr dec & init */
clrscr();
printf("Enter a, b values ");scanf("%d%d",&a,&b);
puts(a>b?"a is big":b>a?"b is big":"Both are equal");
puts(*p>*q?"a is big":*q>*p?"b is big":"Both are equal");
getch();
}
F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F1
10:00 AM
09-Dec-24

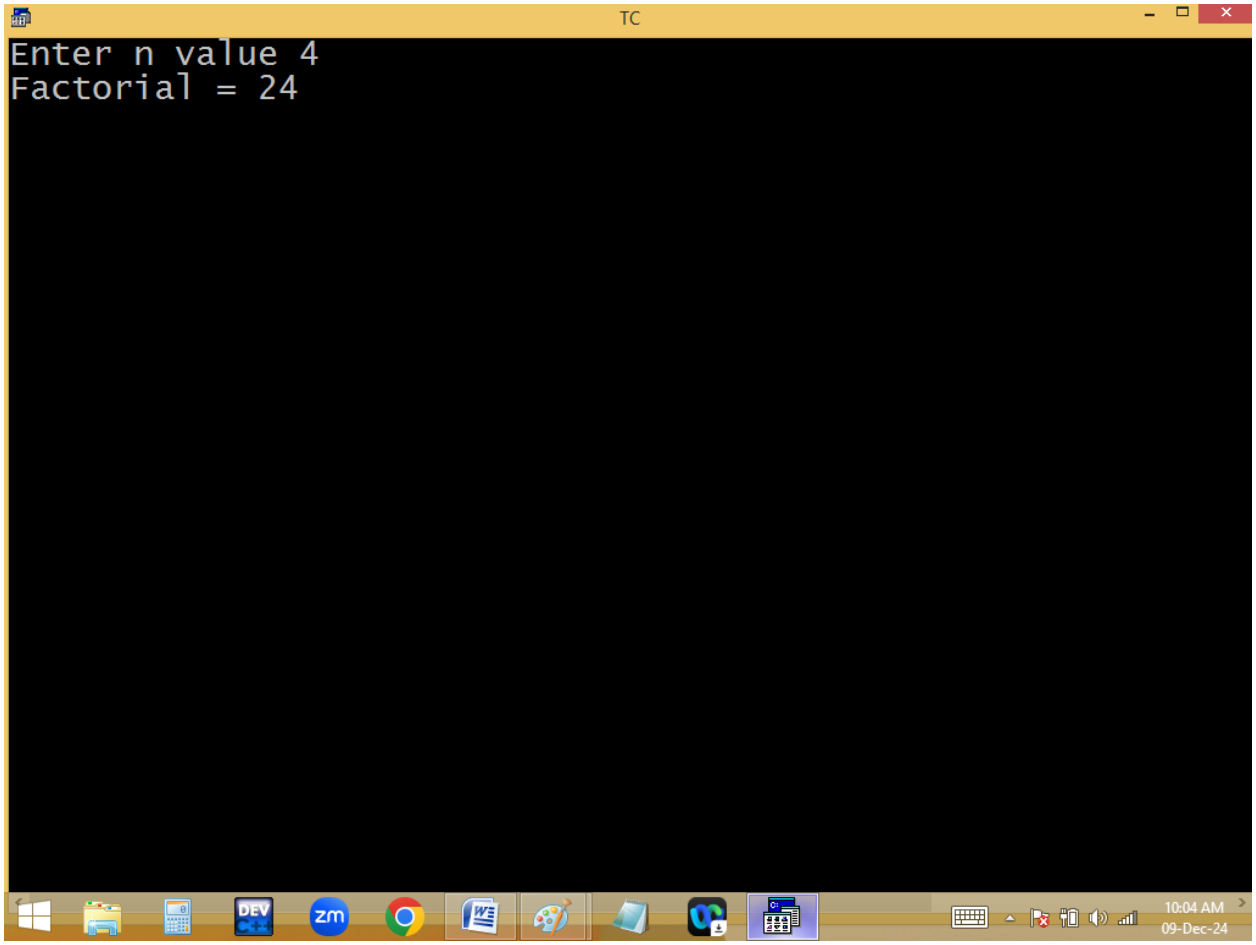

TC
Enter a, b values 1 2
b is big
b is big
10:00 AM
09-Dec-24

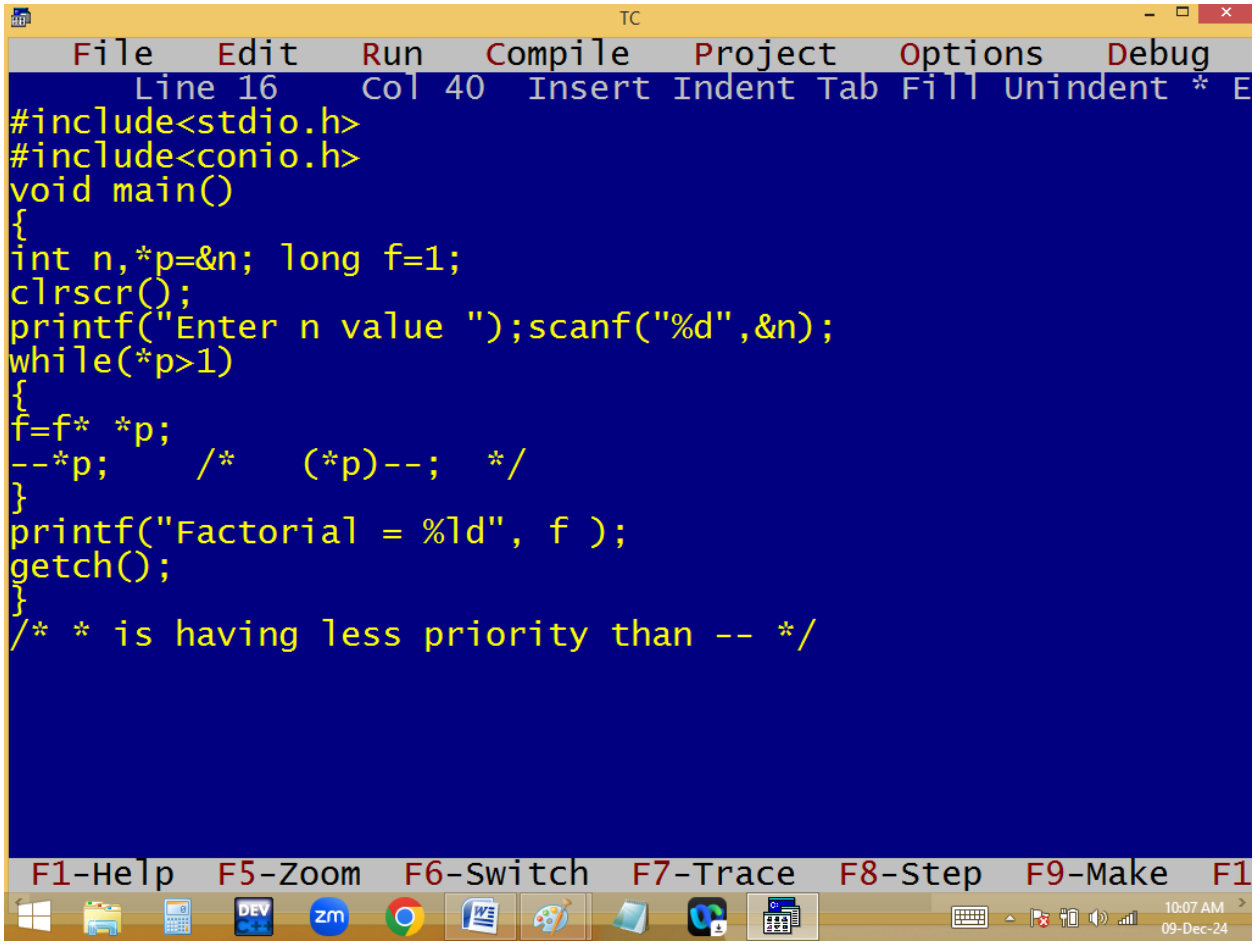

TC
Enter a, b values 5 2
a is big
a is big
10:00 AM
09-Dec-24

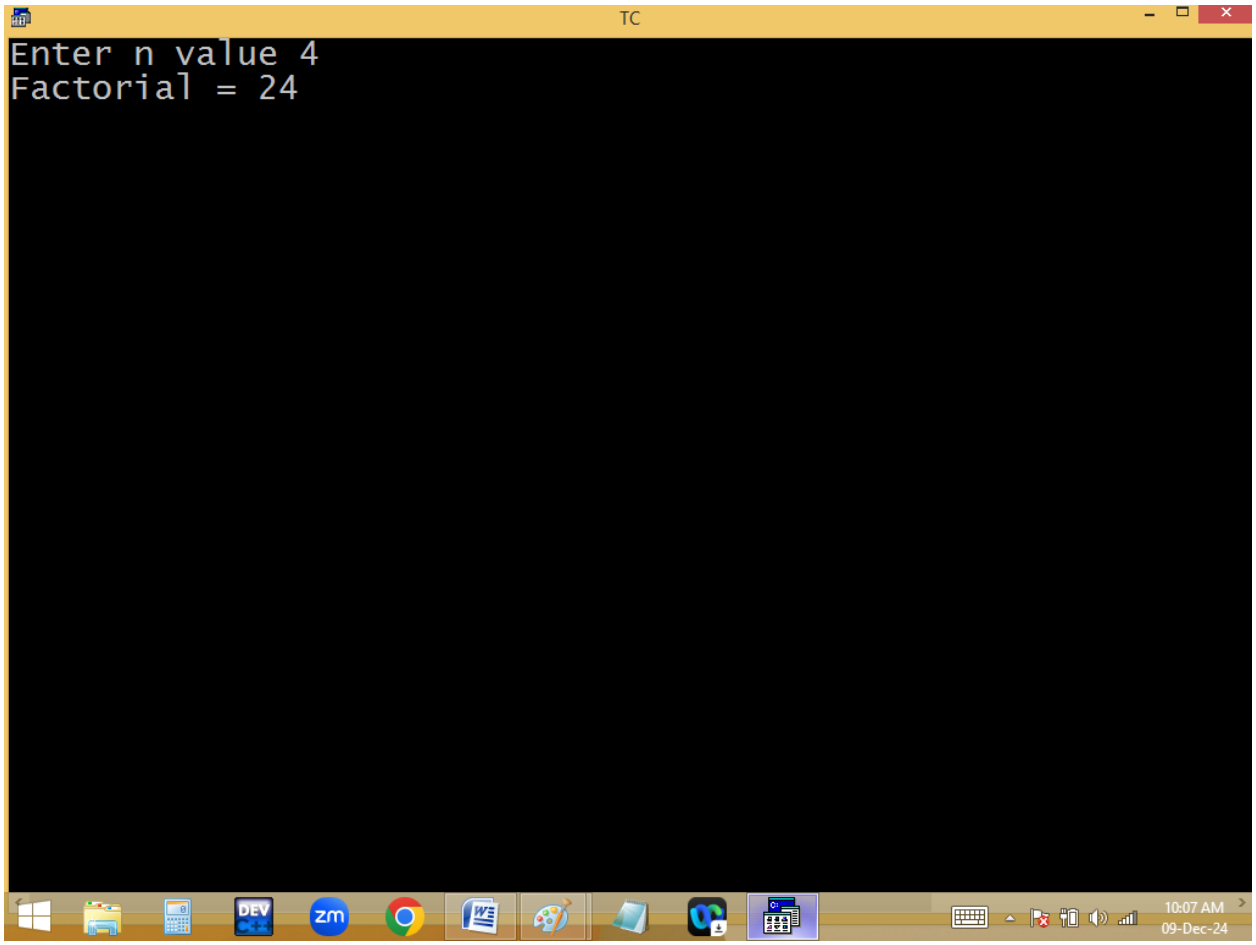


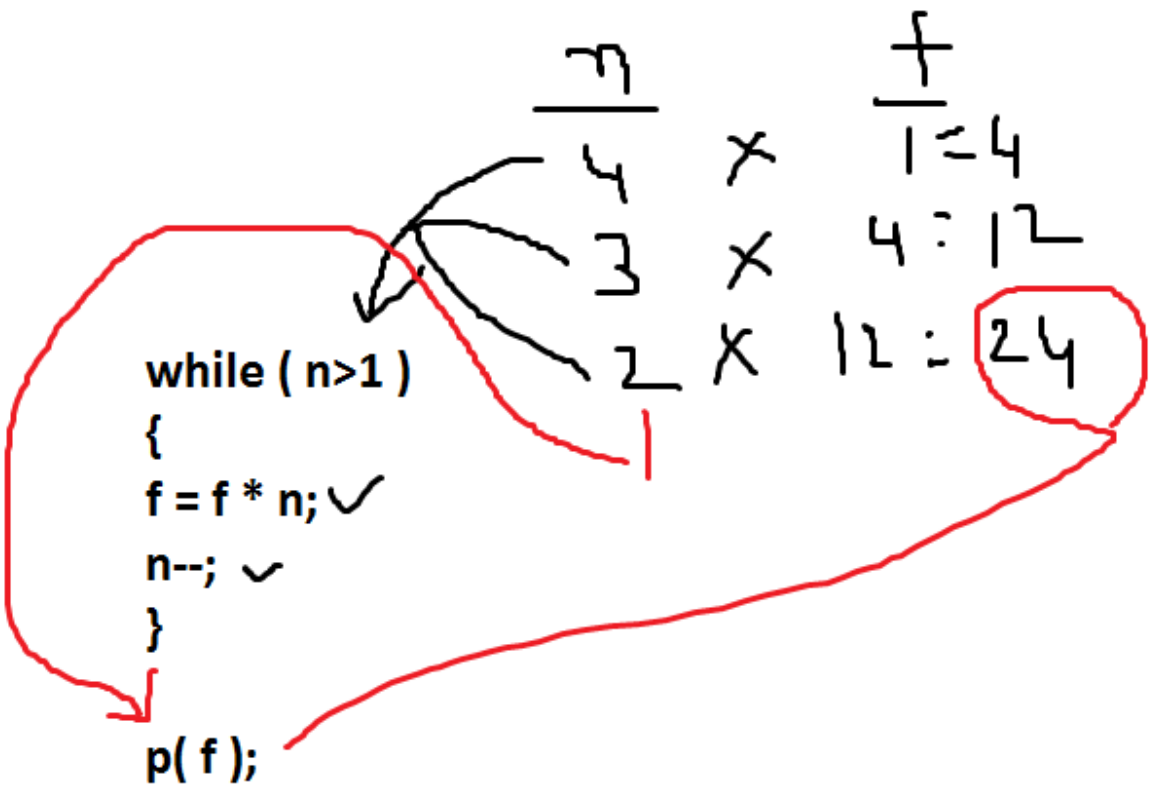
Finding factorial using pointer:


TC
File Edit Run Compile Project Options Debug
Line 13 Col 29 Insert Indent Tab Fill Unindent * E
#include<stdio.h>
#include<conio.h>
void main()
{
int n,*p=&n; long f=1;
clrscr();
printf("Enter n value ");scanf("%d",&n);
while(n>1)
{
f=f*n;
n--;
}
printf("Factorial = %ld", f);
getch();
}
F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F1
10:03 AM
09-Dec-24

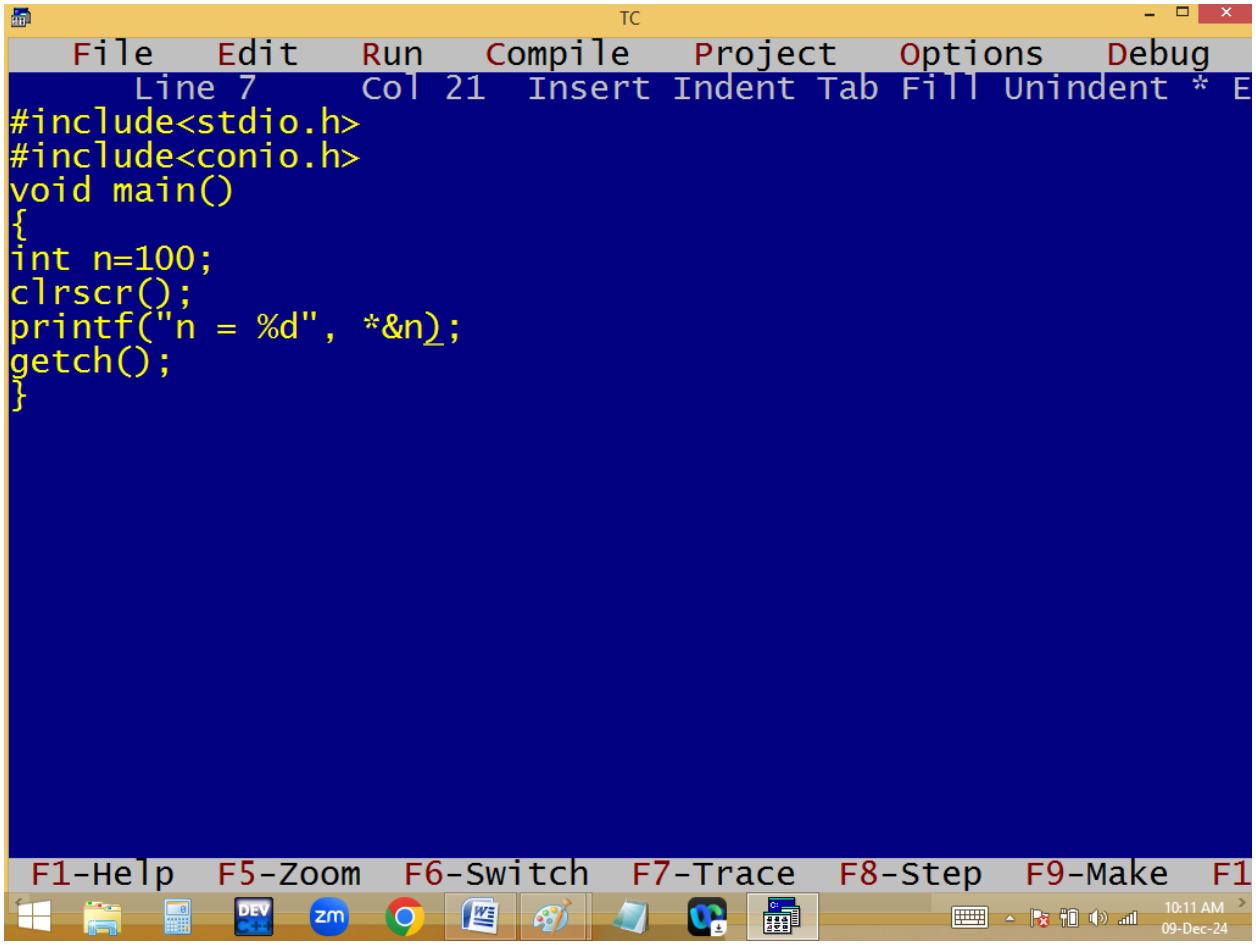

TC
Enter n value 4
Factorial = 24
10:04 AM
09-Dec-24

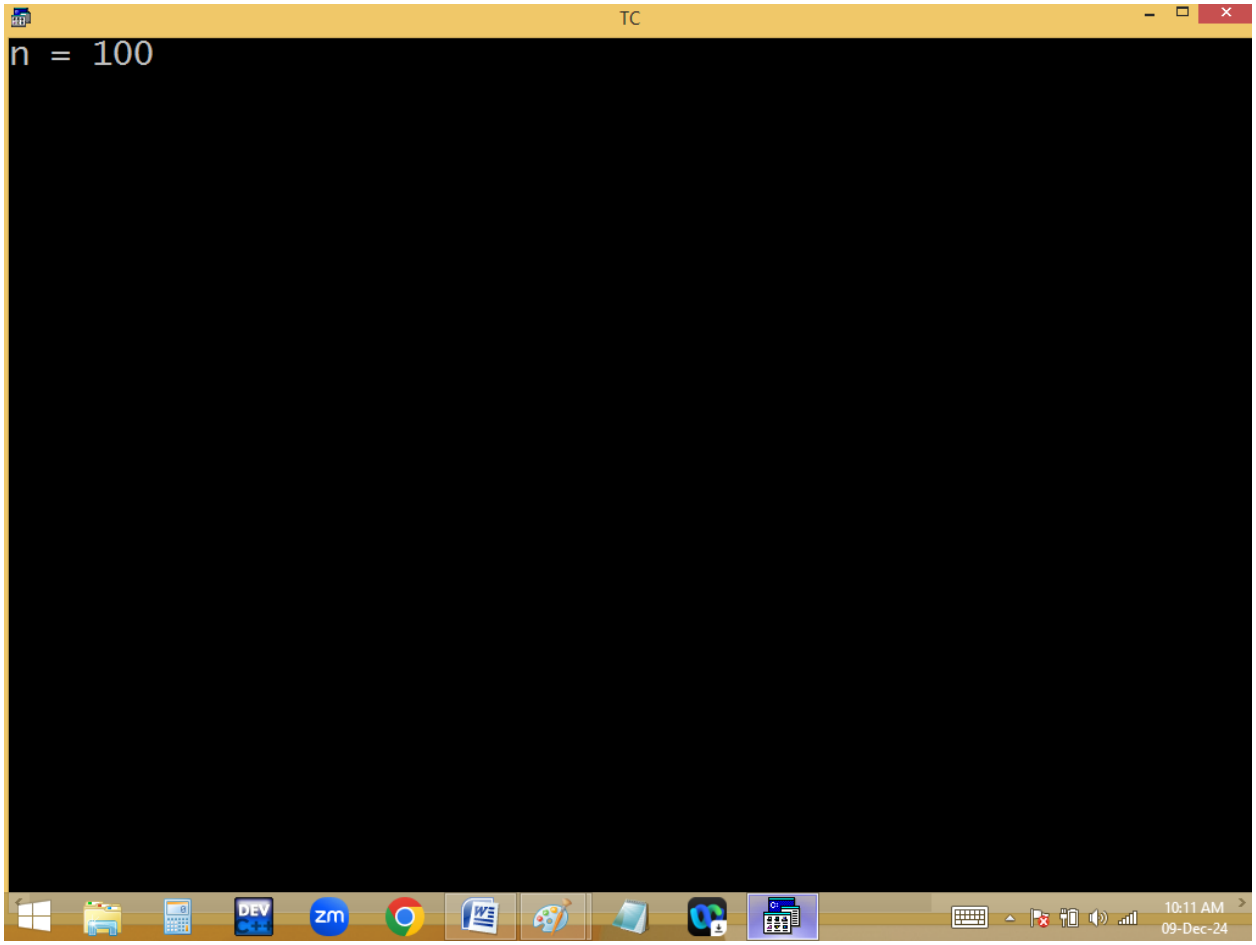

TC
File Edit Run Compile Project Options Debug
Line 16 Col 40 Insert Indent Tab Fill Unindent * E
#include<stdio.h>
#include<conio.h>
void main()
{
int n,*p=&n; long f=1;
clrscr();
printf("Enter n value ");scanf("%d",&n);
while(*p>1)
{
f=f* *p;
--*p; /* (*p)--; */
}
printf("Factorial = %ld", f);
getch();
}
/* * is having less priority than -- */
F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F1
10:07 AM
09-Dec-24


TC
Enter n value 4
Factorial = 24
10:07 AM
09-Dec-24



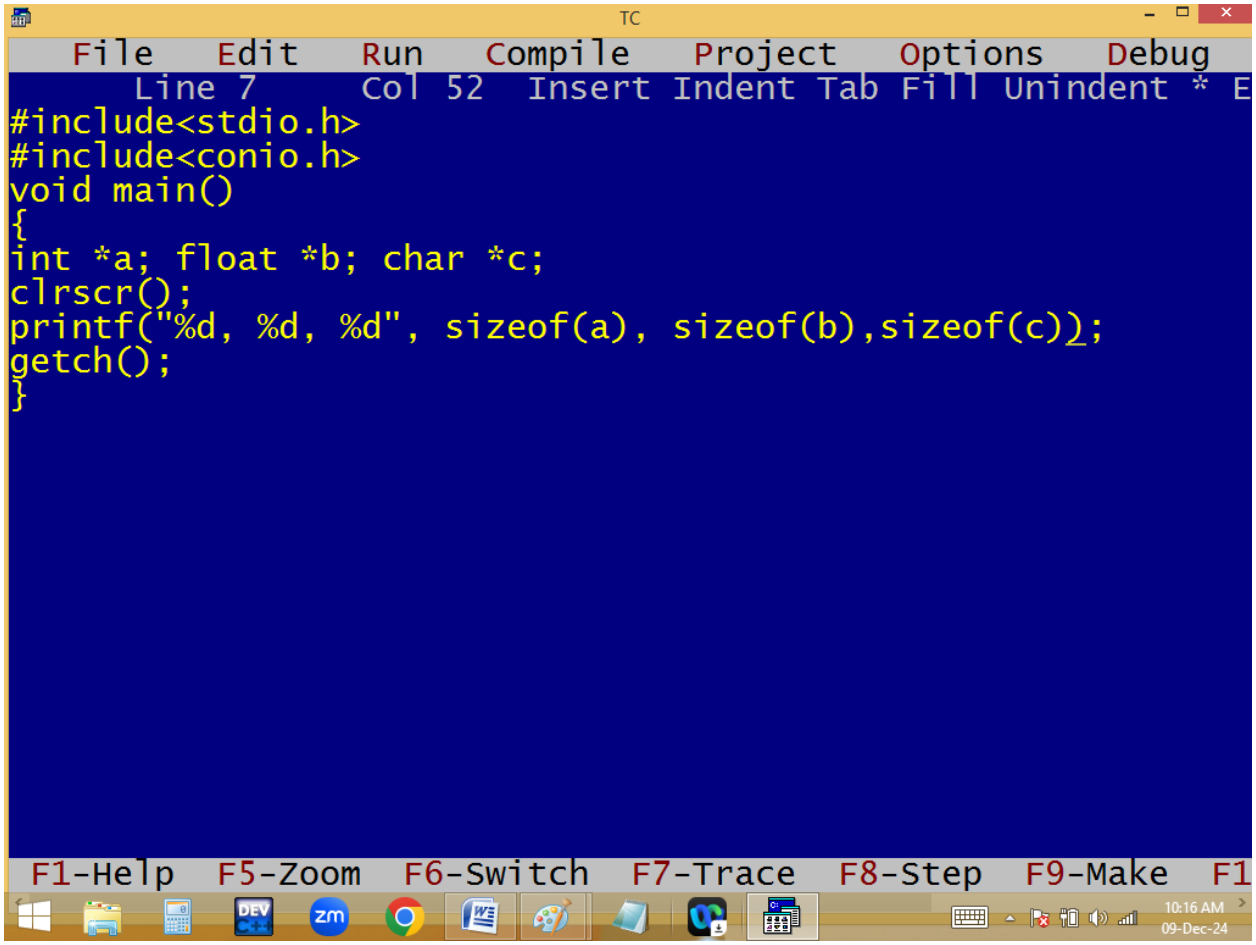
Finding a normal variable value using pointer technique:

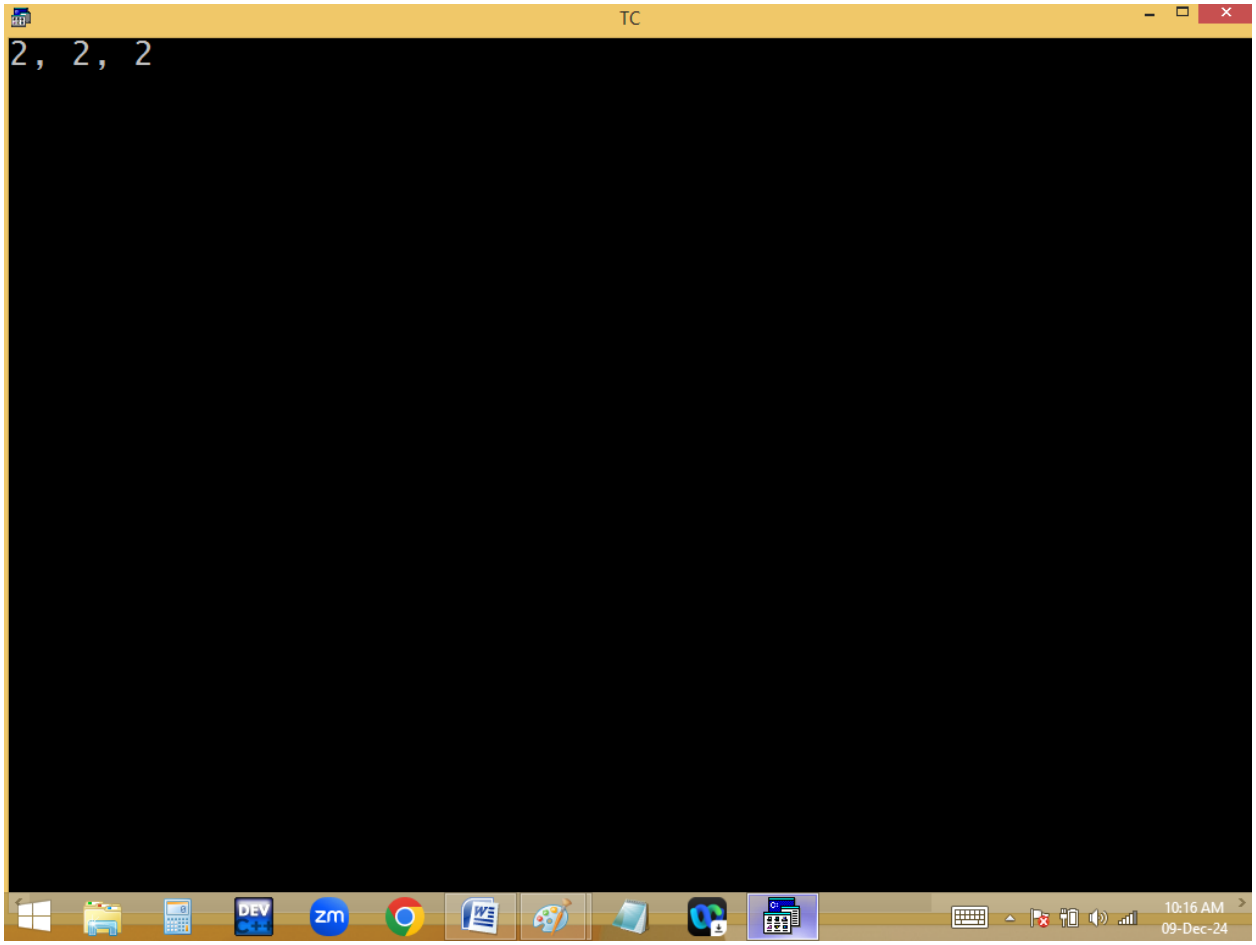

TC
File Edit Run Compile Project Options Debug
Line 7 Col 21 Insert Indent Tab Fill Unindent * E
#include<stdio.h>
#include<conio.h>
void main()
{
int n=100;
clrscr();
printf("n = %d", *&n);
getch();
}
F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F1
10:11 AM
09-Dec-24



Finding pointer size:

Pointer stores always the variable / memory address and it is an unsigned int. due to this any type of pointer it takes 2 / 4 / 8 bytes in 16 / 32 / 64 bit compilers.

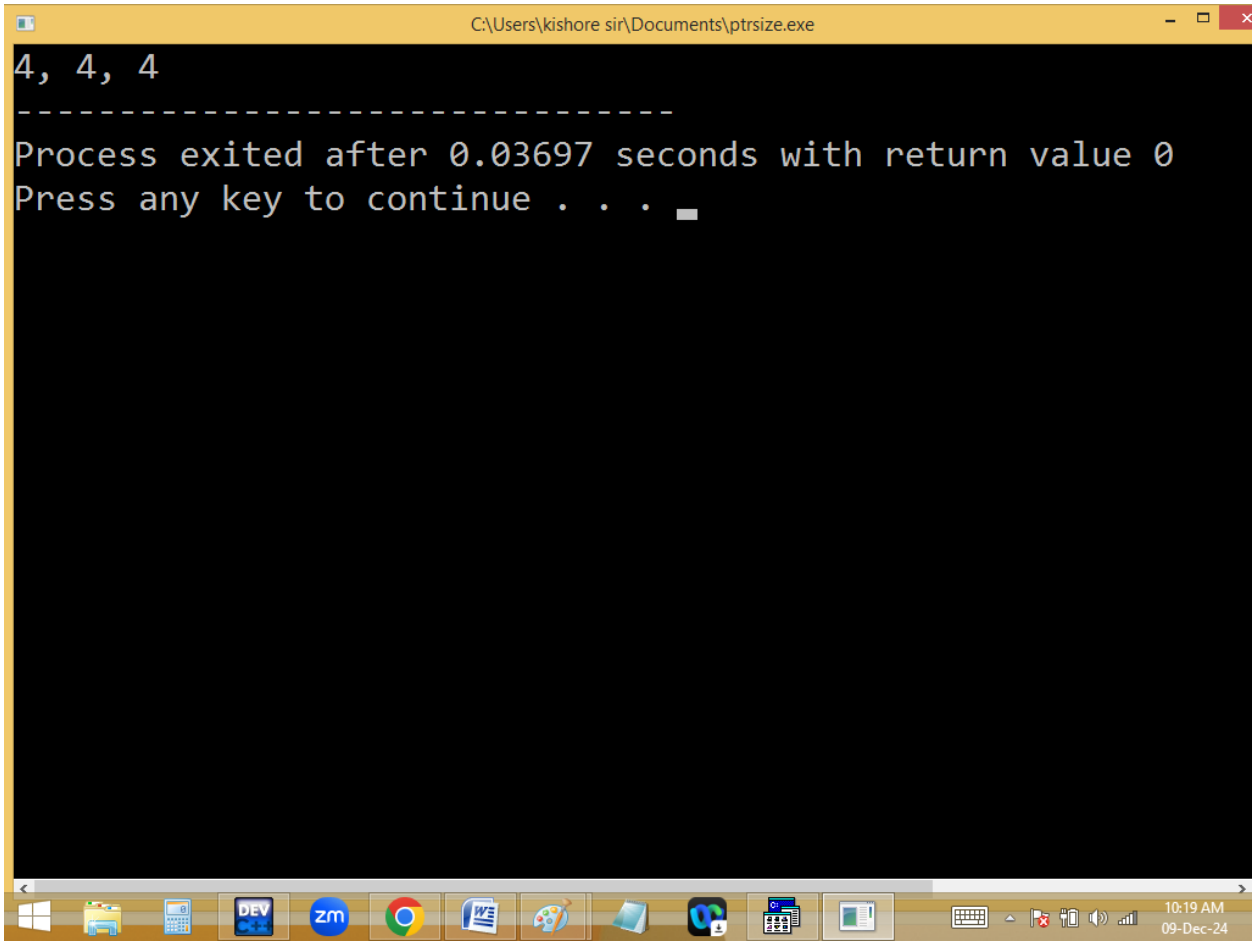

TC
File Edit Run Compile Project Options Debug
Line 7 Col 52 Insert Indent Tab Fill Unindent * E
#include<stdio.h>
#include<conio.h>
void main()
{
int *a; float *b; char *c;
clrscr();
printf("%d, %d, %d", sizeof(a), sizeof(b),sizeof(c));
getch();
}
F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F1
10:16 AM
09-Dec-24



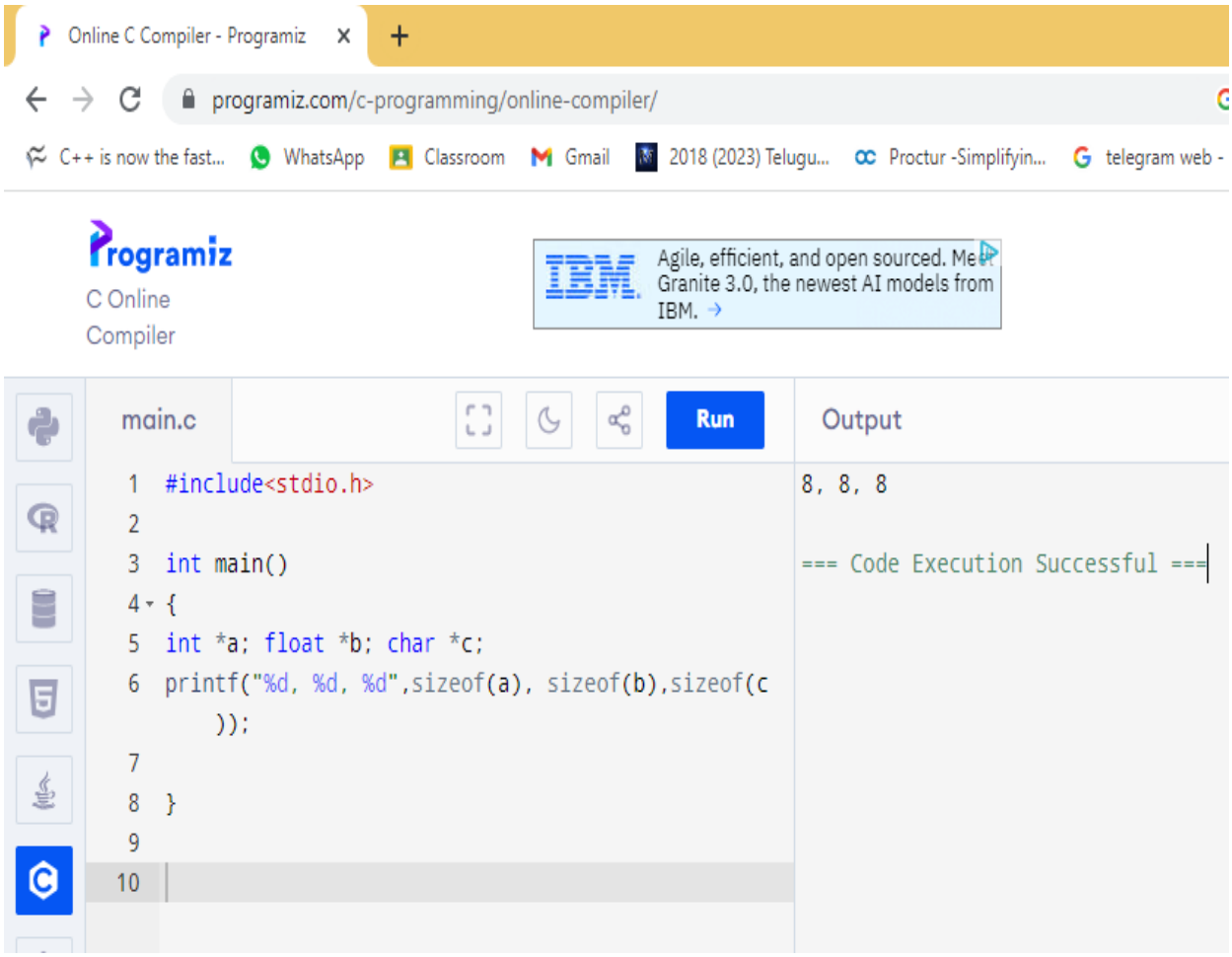
In Dev C++:

```
#include<stdio.h>

int main()

{

int *a; float *b; char *c;

printf("%d, %d, %d",sizeof(a), sizeof(b),sizeof(c));
```

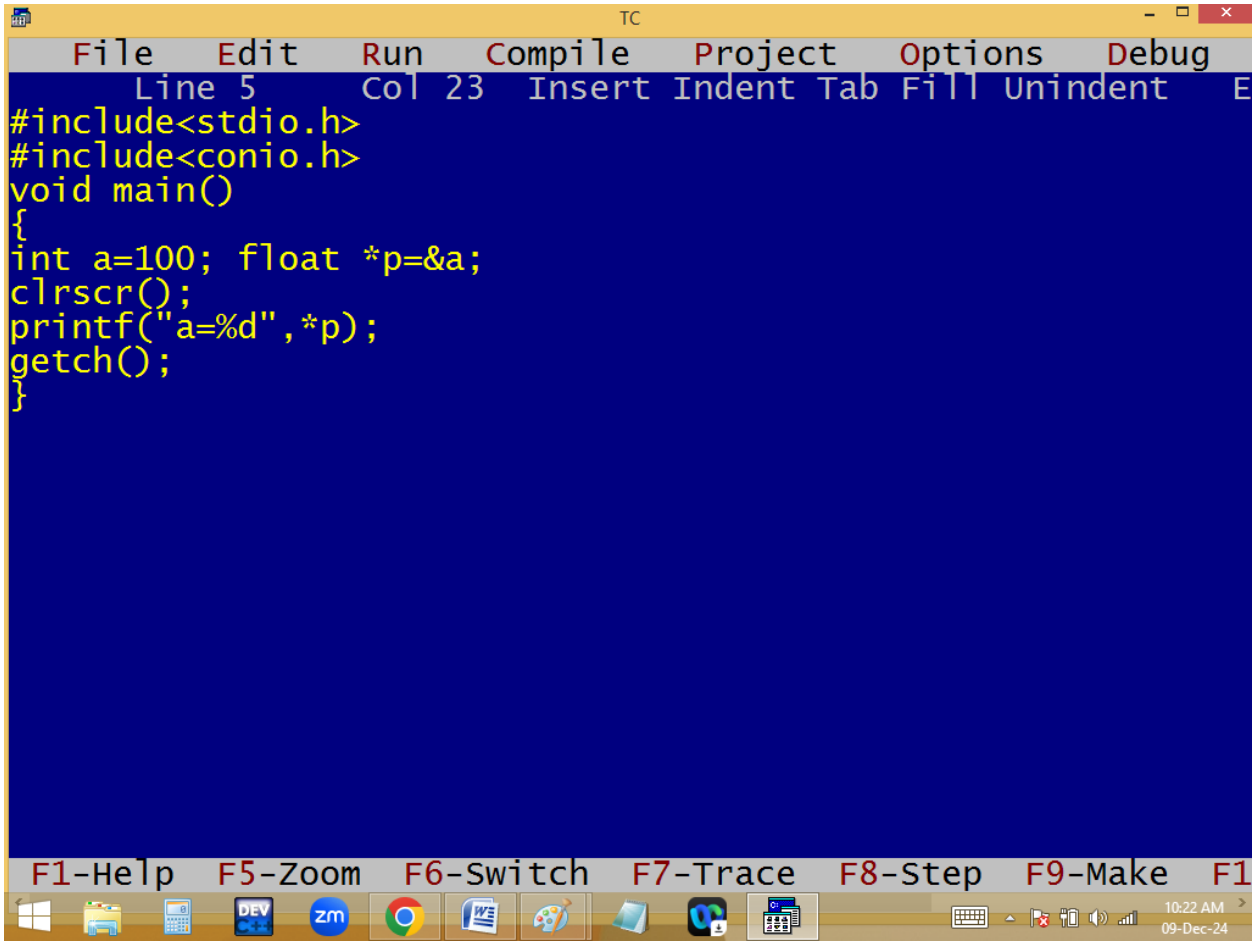
```
}
```

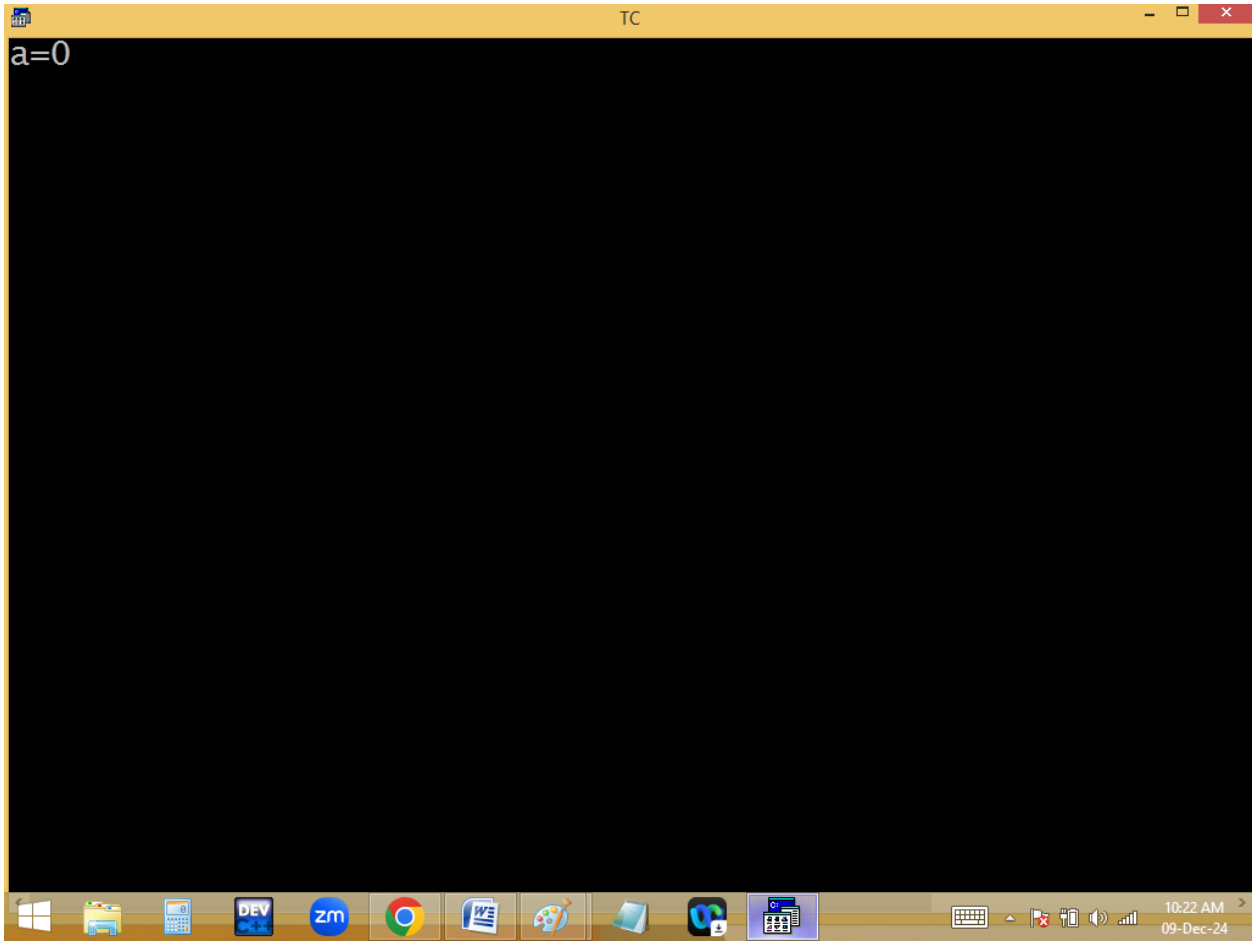


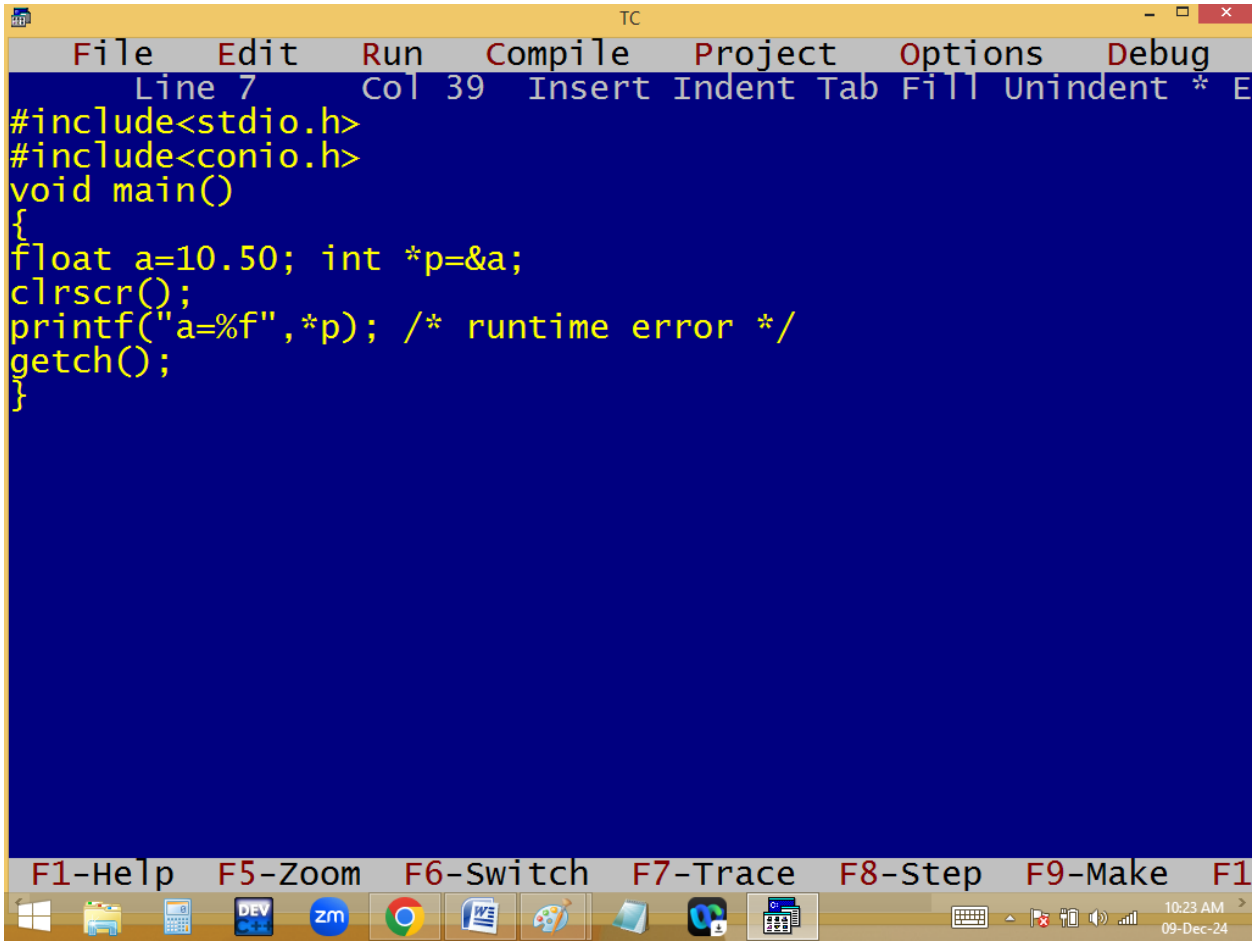
Online compiler:



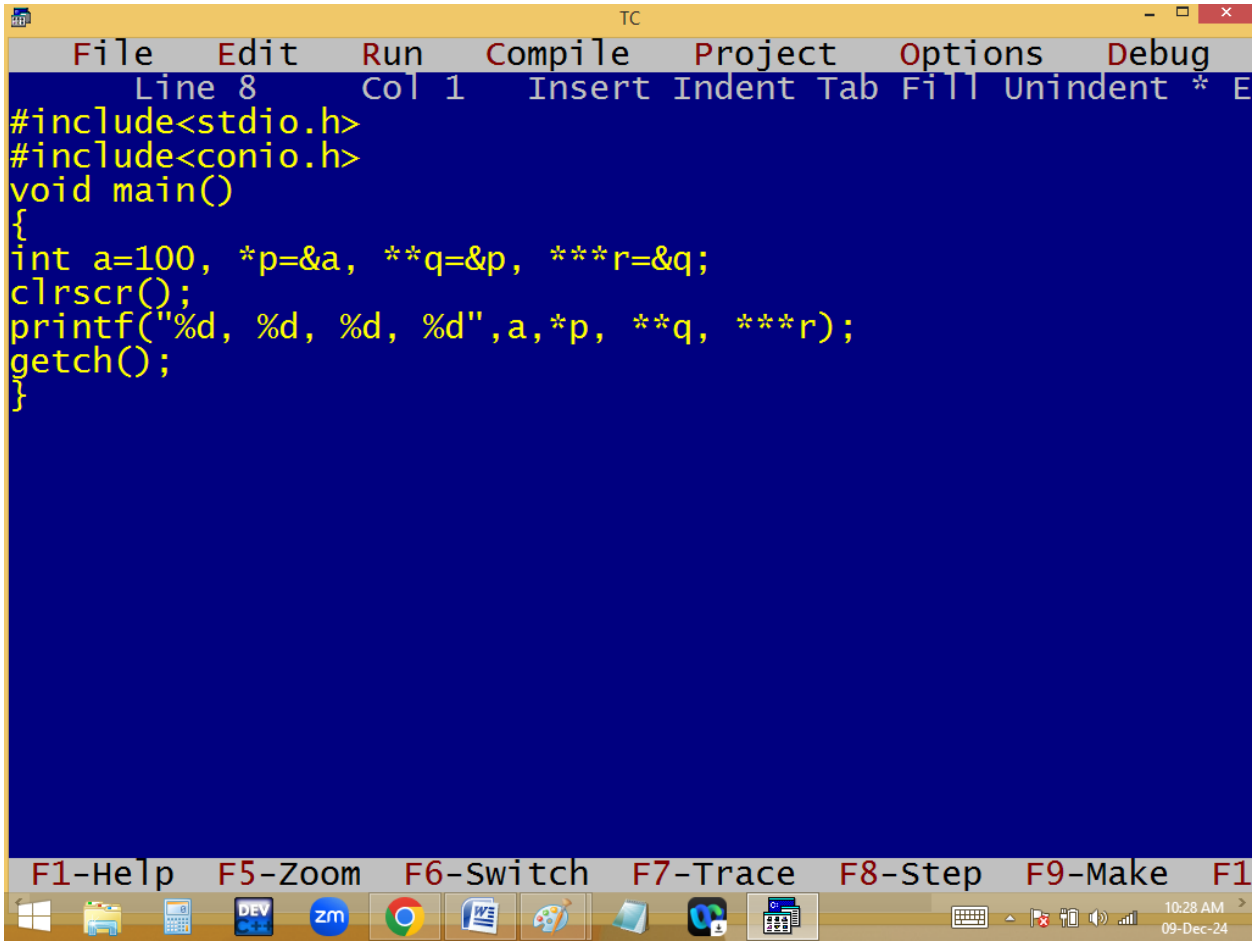
Pointer compatibility: Pointer can store only the same type of variable address. When we are providing different type address, it gives either garbage or runtime error.

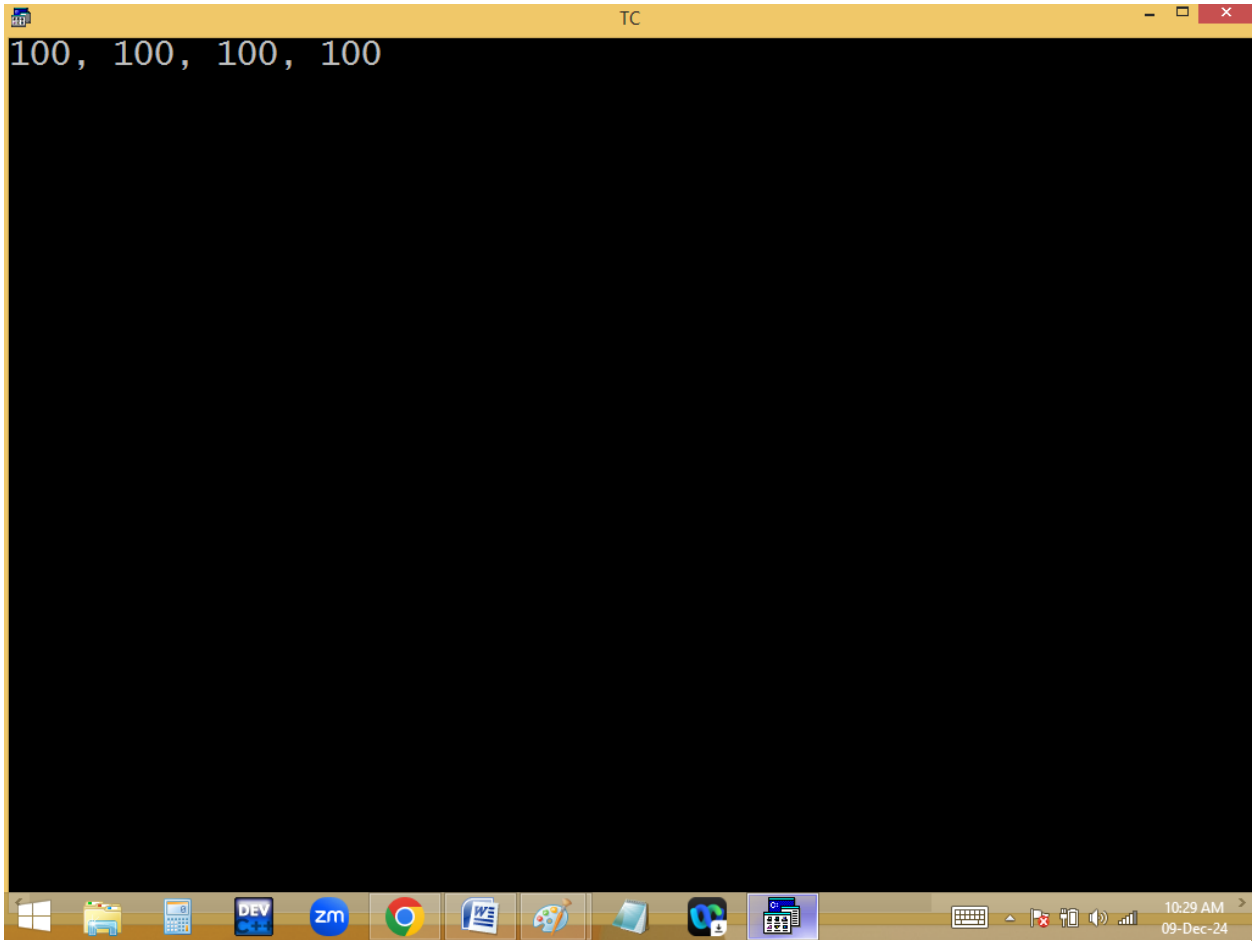

TC
File Edit Run Compile Project Options Debug
Line 5 Col 23 Insert Indent Tab Fill Unindent E
#include<stdio.h>
#include<conio.h>
void main()
{
int a=100; float *p=&a;
clrscr();
printf("a=%d",*p);
getch();
}
F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F1
10:22 AM
09-Dec-24

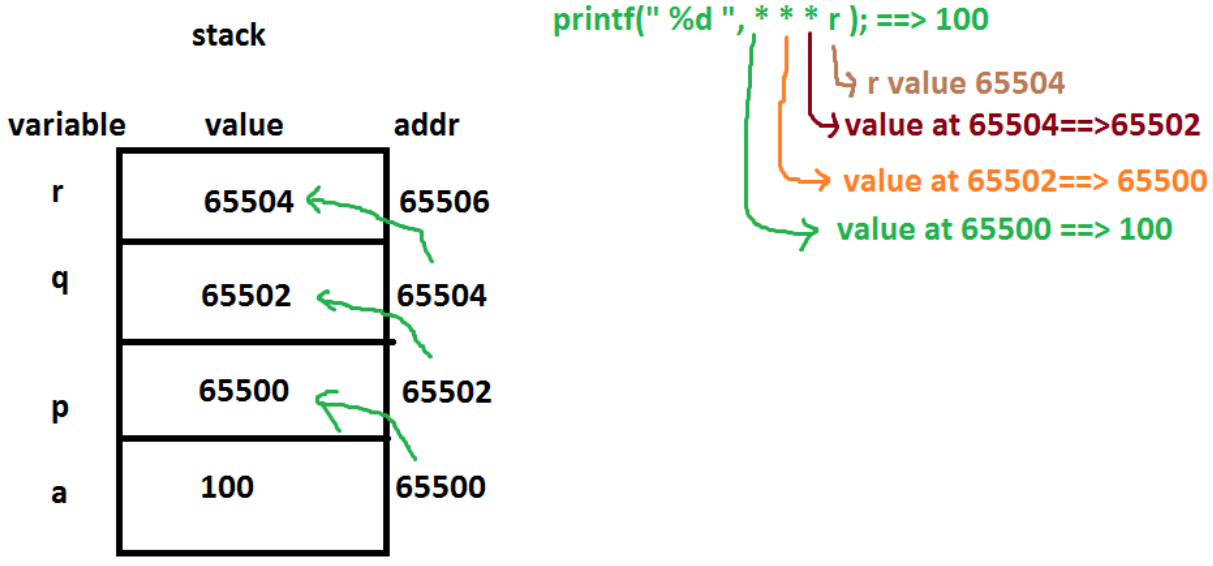

TC
a=0
10:22 AM
09-Dec-24



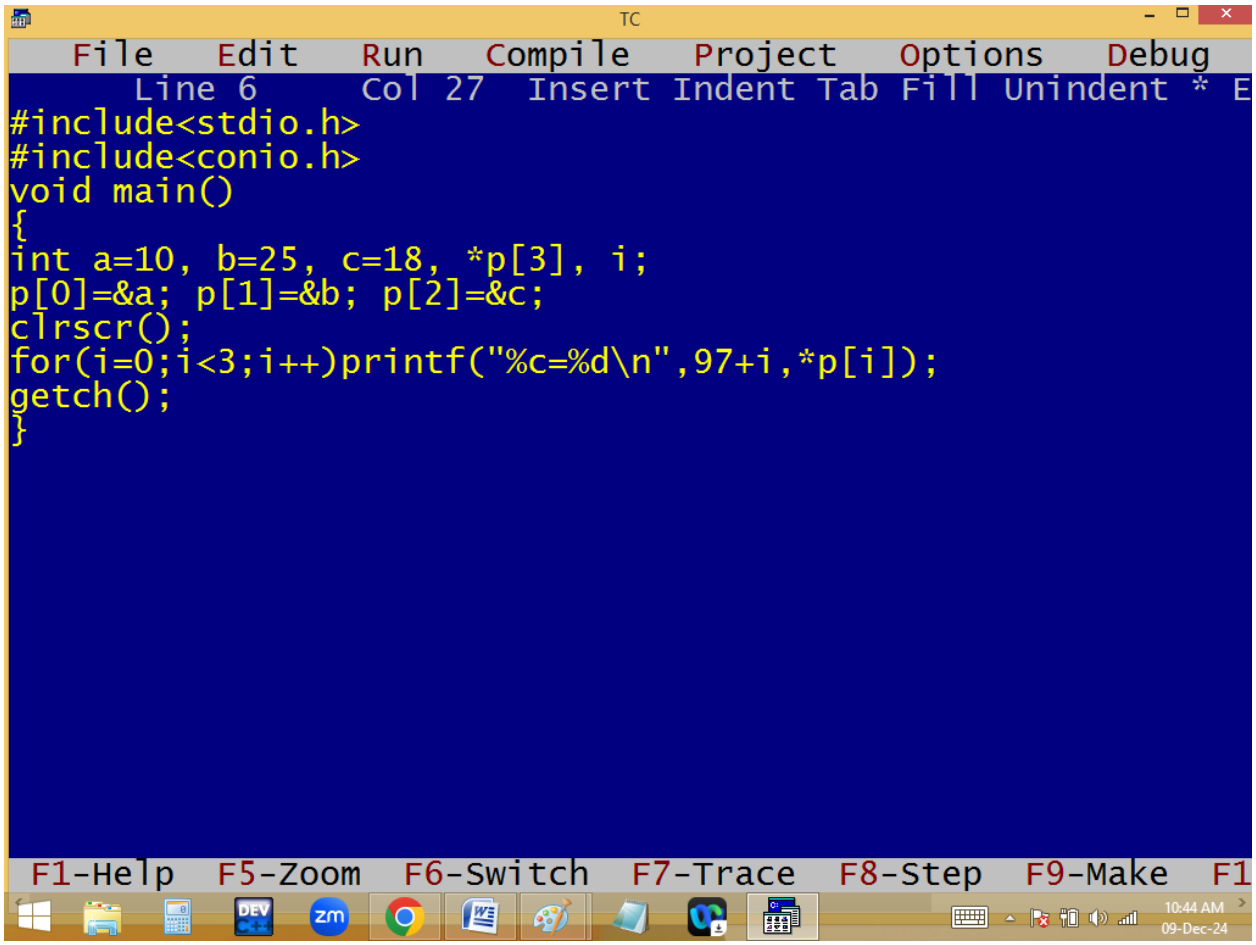
Double pointer / pointer to pointer: the pointer which stores the address of another pointer is called double pointer. It is used to handle dynamic multi dimensional arrays.

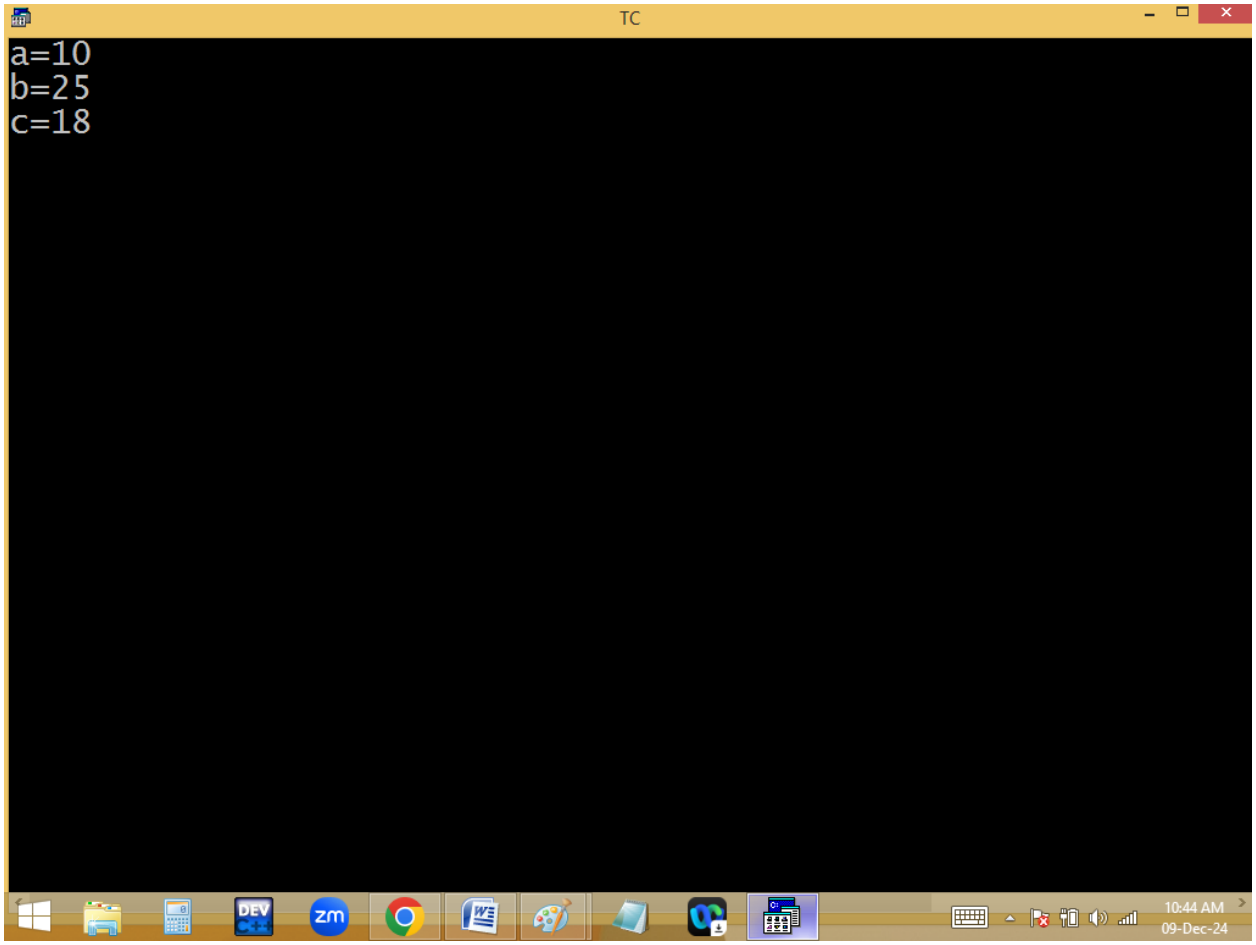

TC
File Edit Run Compile Project Options Debug
Line 8 Col 1 Insert Indent Tab Fill Unindent * E
#include<stdio.h>
#include<conio.h>
void main()
{
int a=100, *p=&a, **q=&p, ***r=&q;
clrscr();
printf("%d, %d, %d, %d",a,*p, **q, ***r);
getch();
}
F1-Help F5-Zoom F6-Switch F7-Trace F8-Step F9-Make F1
10:28 AM
09-Dec-24

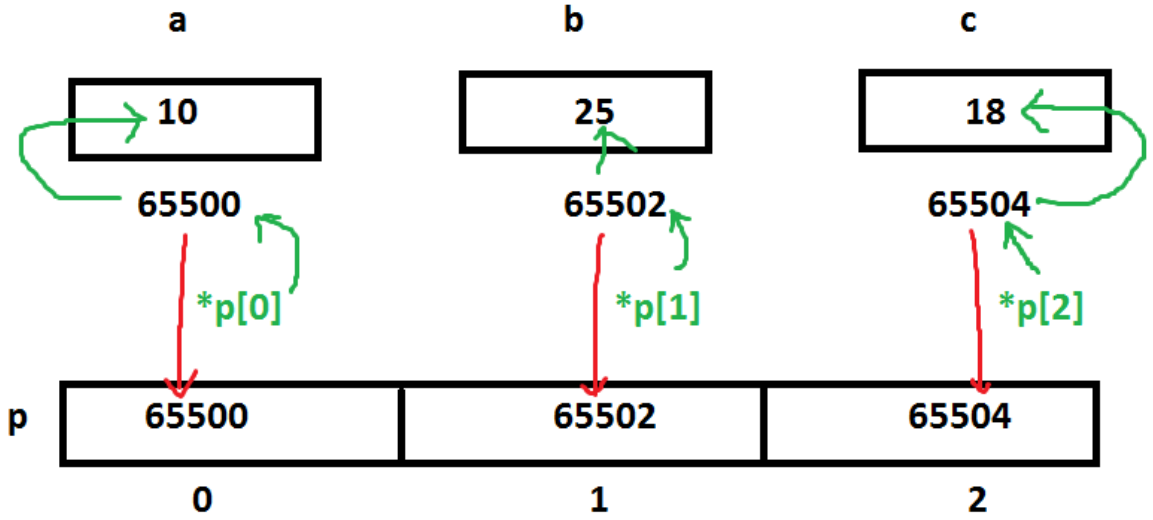

TC
100, 100, 100, 100
10:29 AM
09-Dec-24


stack
variable
value
addr
r
65504
65506
q
65502
65504
p
65500
65502
a
100
65500
printf(" %d ", * * * r); ==> 100
r value 65504
value at 65504==>65502
value at 65502==> 65500
value at 65500 ==> 100

Array of pointer: Like general variables, we can also declare pointer using array. Due to this we can store multiple variable address in a single pointer. It is used to handle dynamic multi dimensional arrays.




TC
a=10
b=25
c=18
10:44 AM
09-Dec-24


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