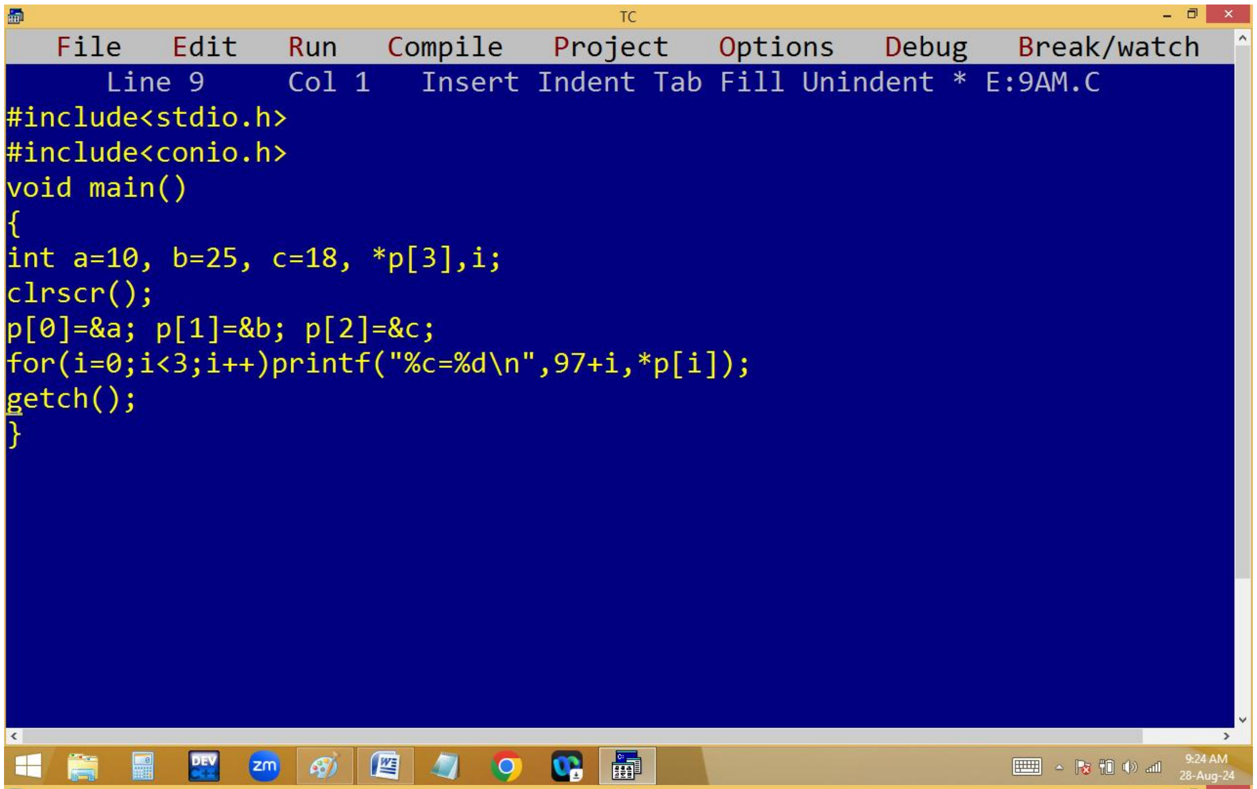
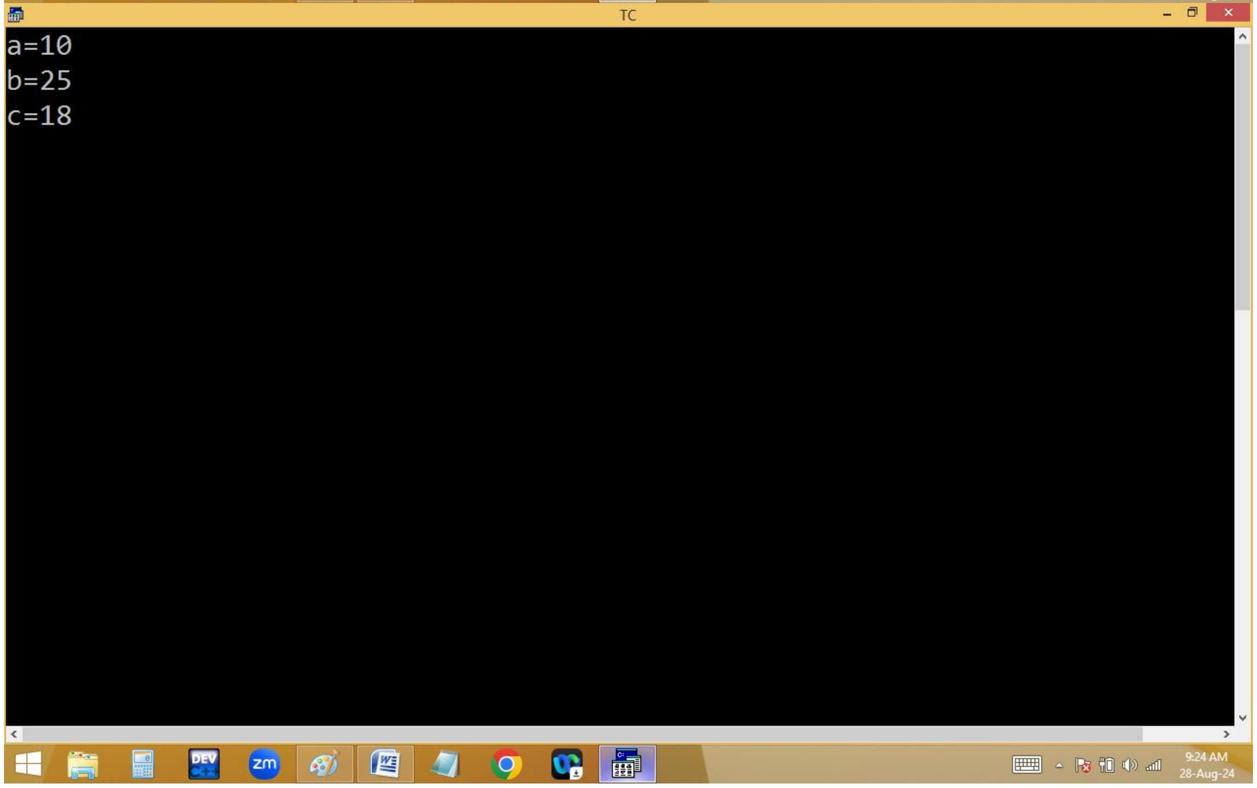
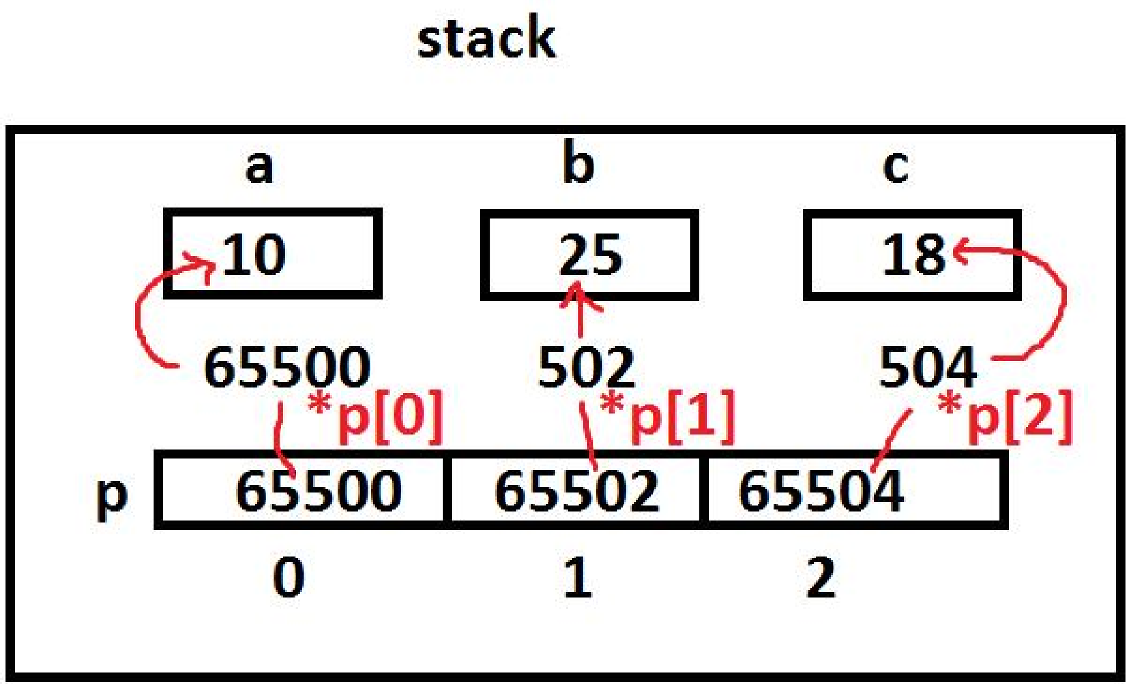
Array of pointers: pointer stores address of a variable. when several variables are there we have to define several pointers. By using array of pointer we can store several address in one pointer variable. it is used to handle **dynamic multi dimensional** arrays.


TC
File Edit Run Compile Project Options Debug Break/watch
Line 9 Col 1 Insert Indent Tab Fill Unindent * E:9AM.C
9:24 AM
28-Aug-24

```c
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10, b=25, c=18, *p[3],i;
clrscr();
p[0]=&a; p[1]=&b; p[2]=&c;
for(i=0;i<3;i++)printf("%c=%d\n",97+i,*p[i]);
getch();
}
```



```
a=10
b=25
c=18
```



Pointer to array:

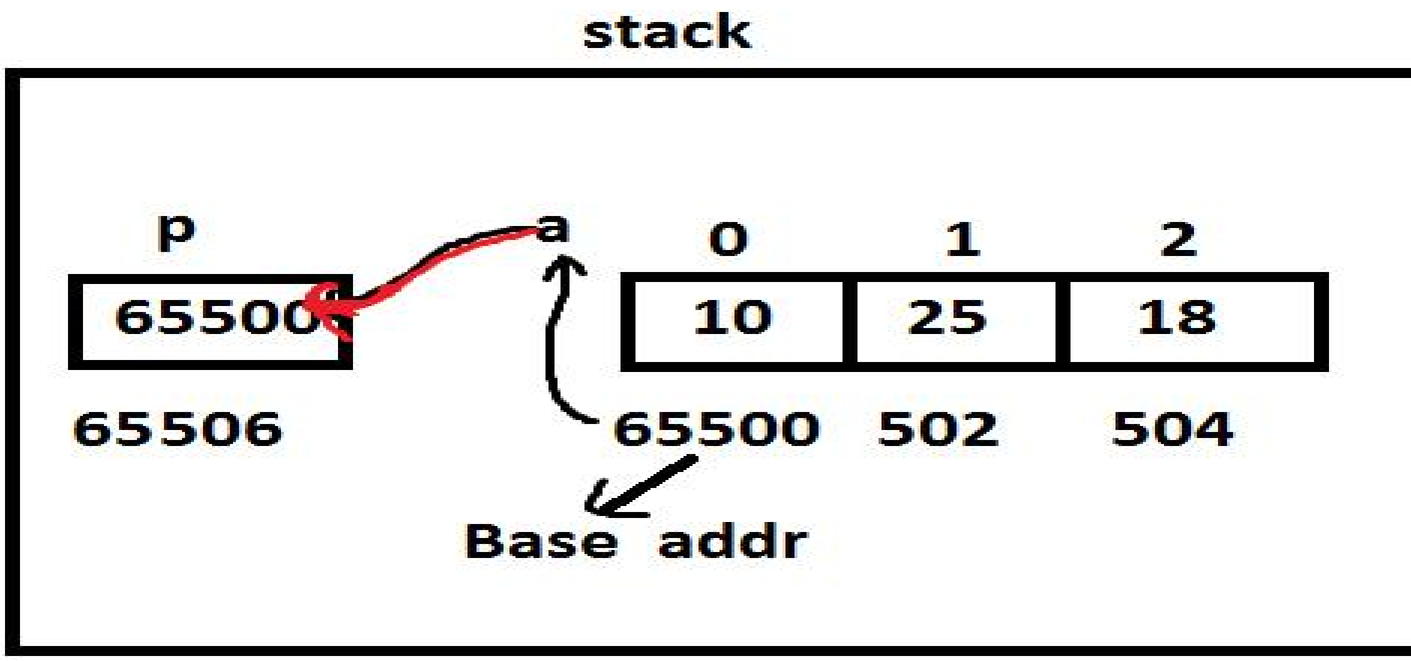
Array is implicit pointer. Due to this it holds the base cell addr [0 cell addr] implicitly. By assigning the array name or 0 cell addr to the pointer, we can handle array elements using the following syntax.

***(ptrvariable + offset/index * sizeof(variable));**

Eg:

int a[3]={10, 25, 18}, *p, i;

p = a ; or p = &a[0]; or p = &a;



for(i=0;i<3;i++)

printf(“%4d”, *(p+i));

Here *(p+i) meaning is:

p is 65500

1. *(p+0***2**)➔*65500➔value at 65500➔**10**
2. *(p+1***2**)➔*65502➔value at 65502➔**25**
3. *(p+2***2**)➔*65504➔value at 65504➔**18**

Note: Here 2 is int size.

Eg:

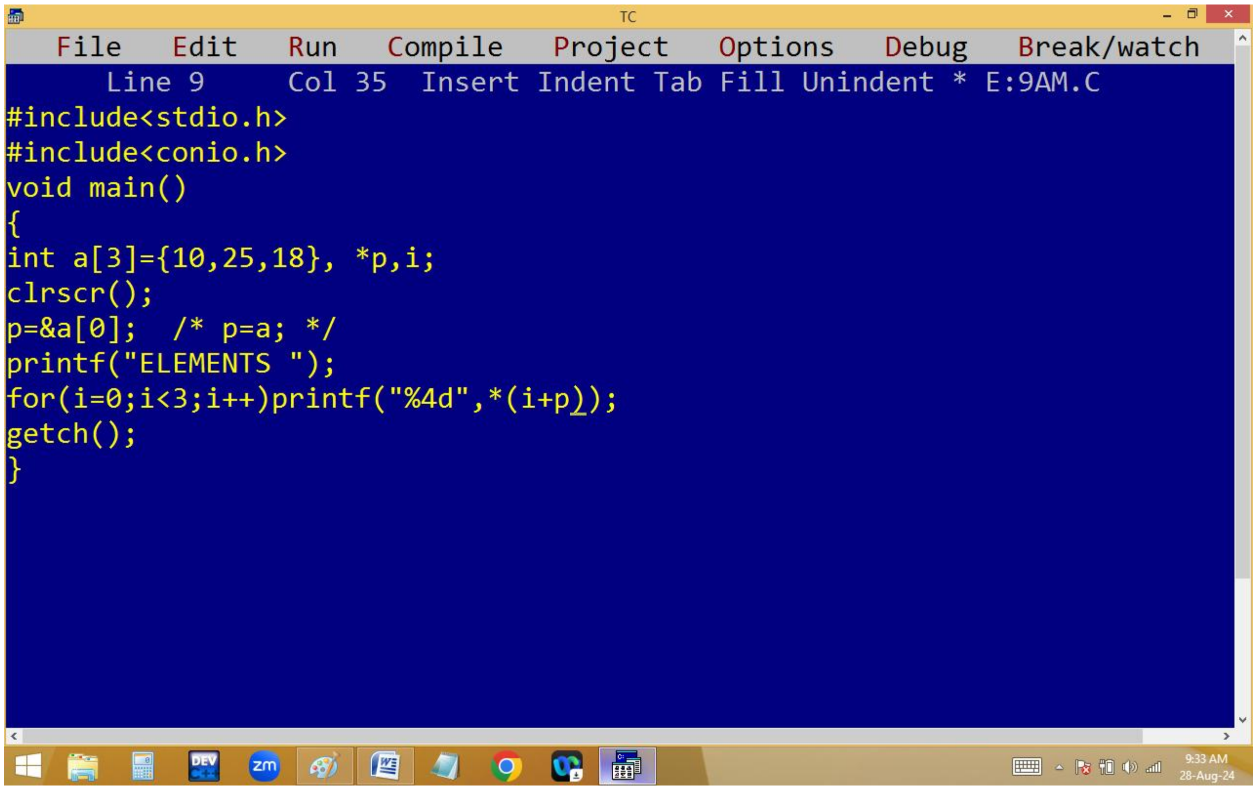
```
#include<stdio.h>
#include<conio.h>
void  main()
{
int  *p,  a[3]={10,25,18}, i;
clrscr();
p = a;   /* p=&a[0]; or p=&a; */
```

printf("Elements are: ");
for(i=0;i<3;i++)
printf("%4d",*(p+i));
getch();
}

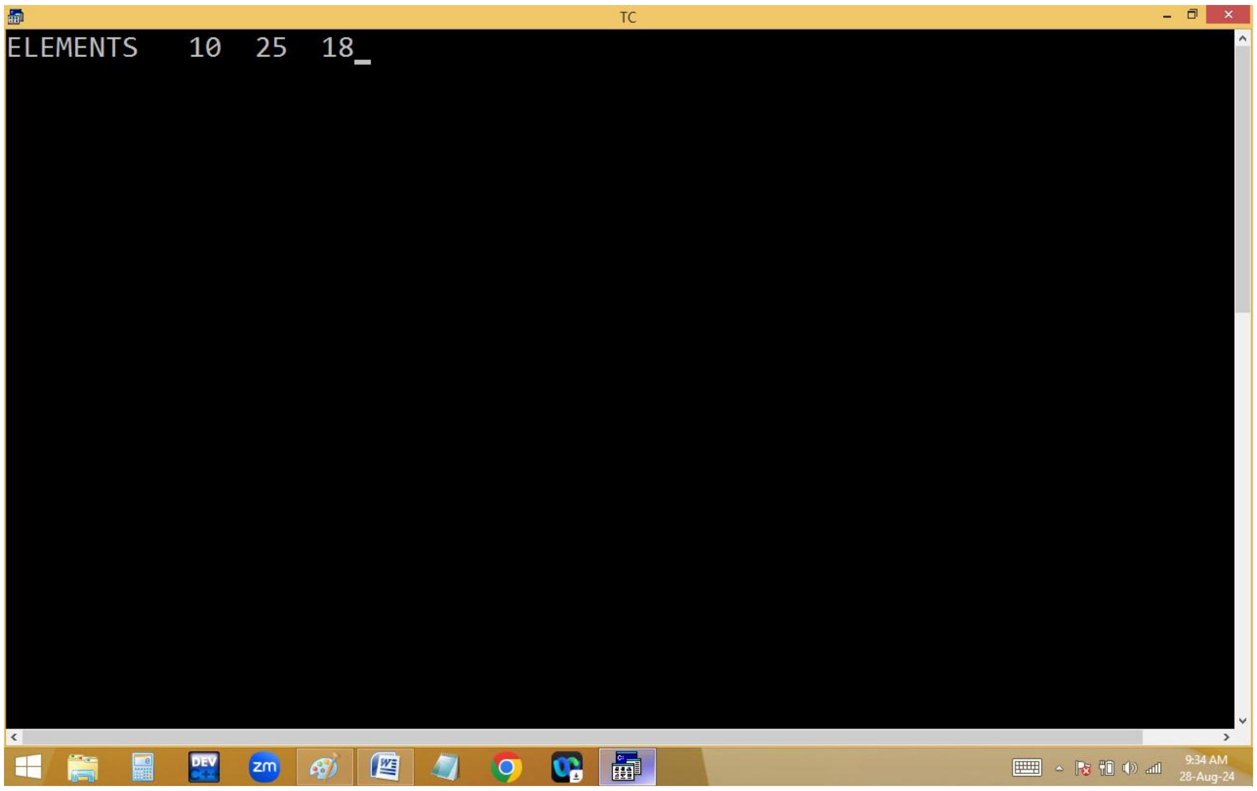
Output: Elements are: 10   25   18

Note: We can access array elements using array / pointer in following ways.

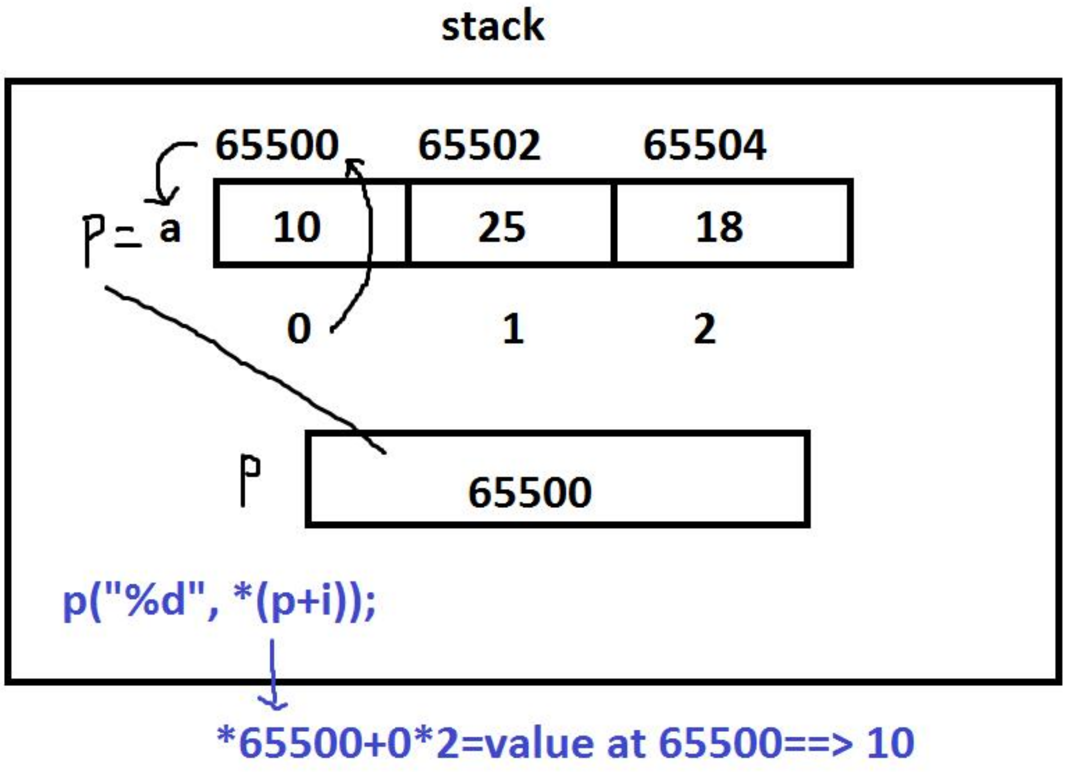
a[i] / i[a] / p[i] / i[p] / *(p+i) / *(a+i) / *(i+p) / *(i+a)


TC
File Edit Run Compile Project Options Debug Break/watch
Line 9 Col 35 Insert Indent Tab Fill Unindent * E:9AM.C

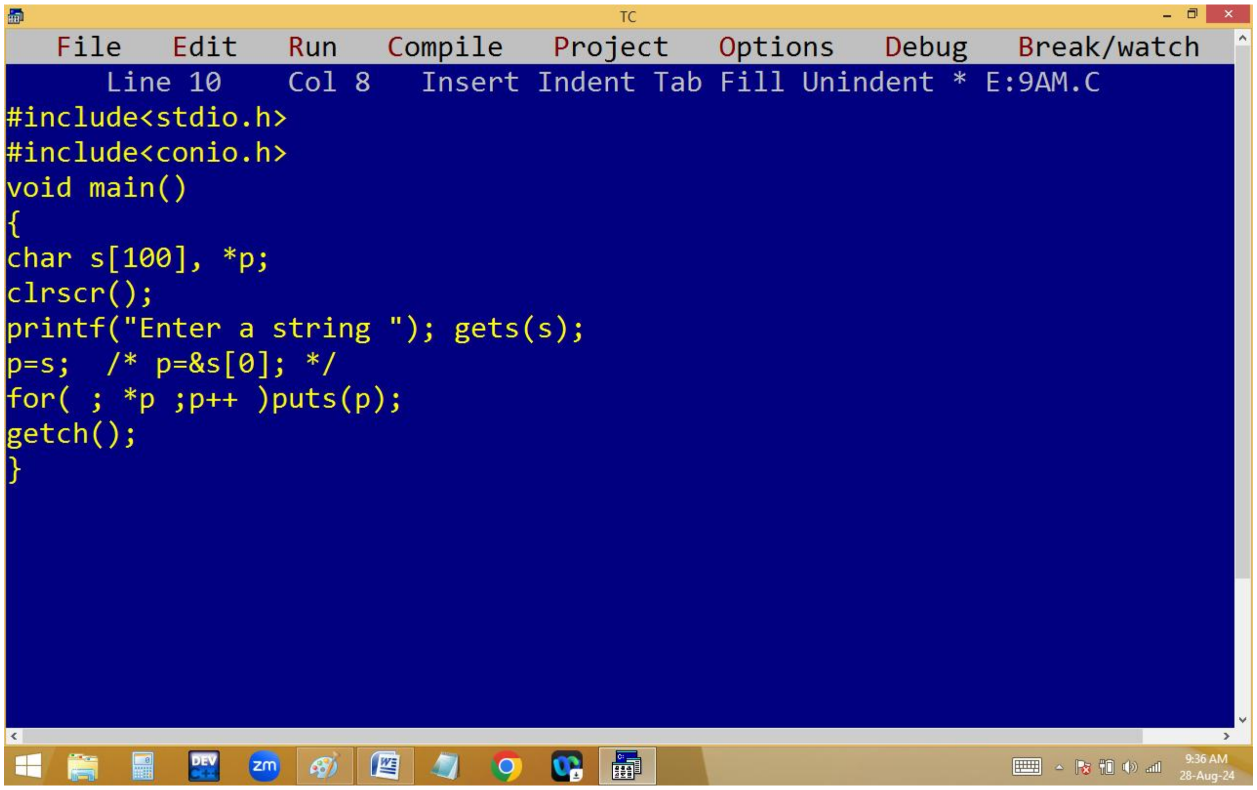
```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[3]={10,25,18}, *p,i;
clrscr();
p=&a[0];  /* p=a; */
printf("ELEMENTS ");
for(i=0;i<3;i++)printf("%4d",*(i+p));
getch();
}
```



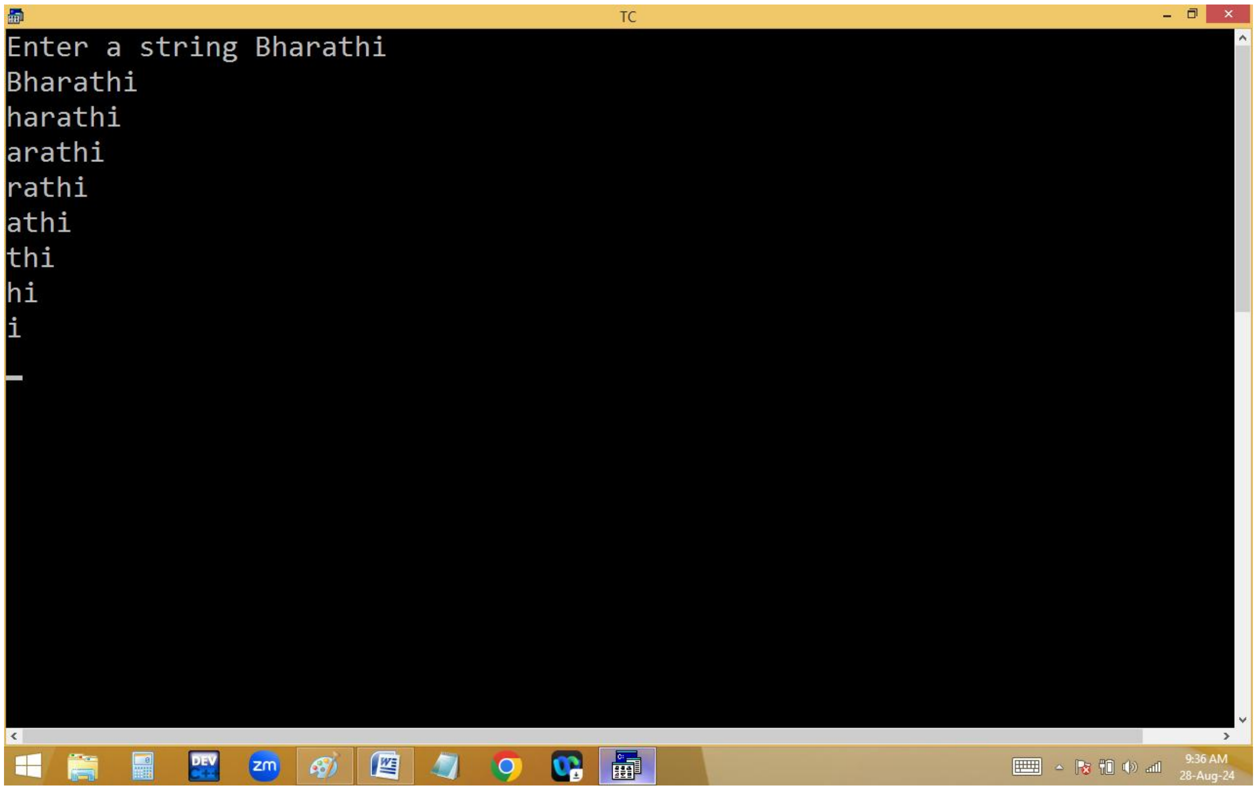
```
ELEMENTS    10   25   18_
```



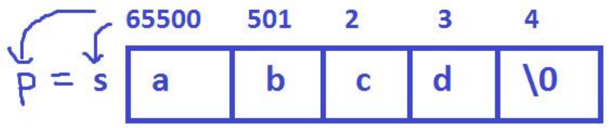
Pointer to string:


TC
File Edit Run Compile Project Options Debug Break/watch
Line 10 Col 8 Insert Indent Tab Fill Unindent * E:9AM.C
9:36 AM
28-Aug-24

```
#include<stdio.h>
#include<conio.h>
void main()
{
char s[100], *p;
clrscr();
printf("Enter a string "); gets(s);
p=s;  /* p=&s[0]; */
for( ; *p ;p++ )puts(p);
getch();
}
```

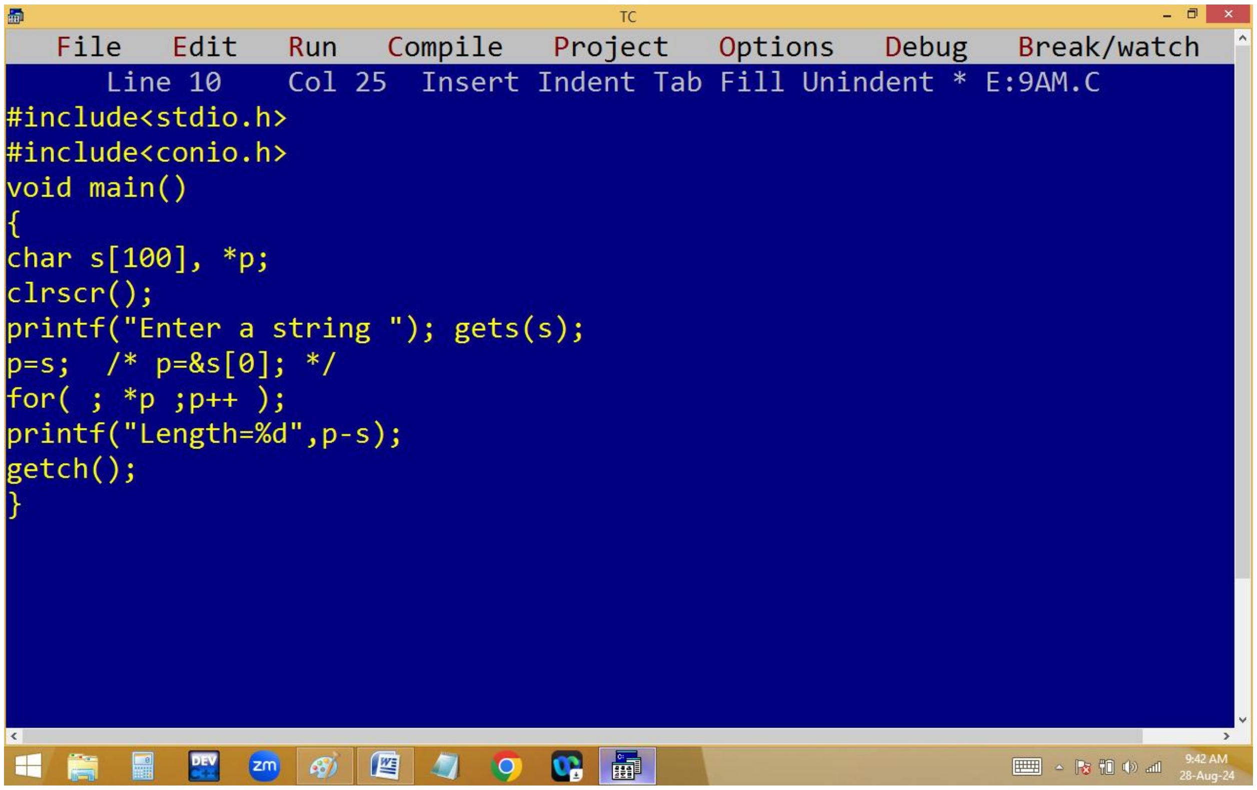


```
Enter a string Bharathi
Bharathi
harathi
arathi
rathi
athi
thi
hi
i
```

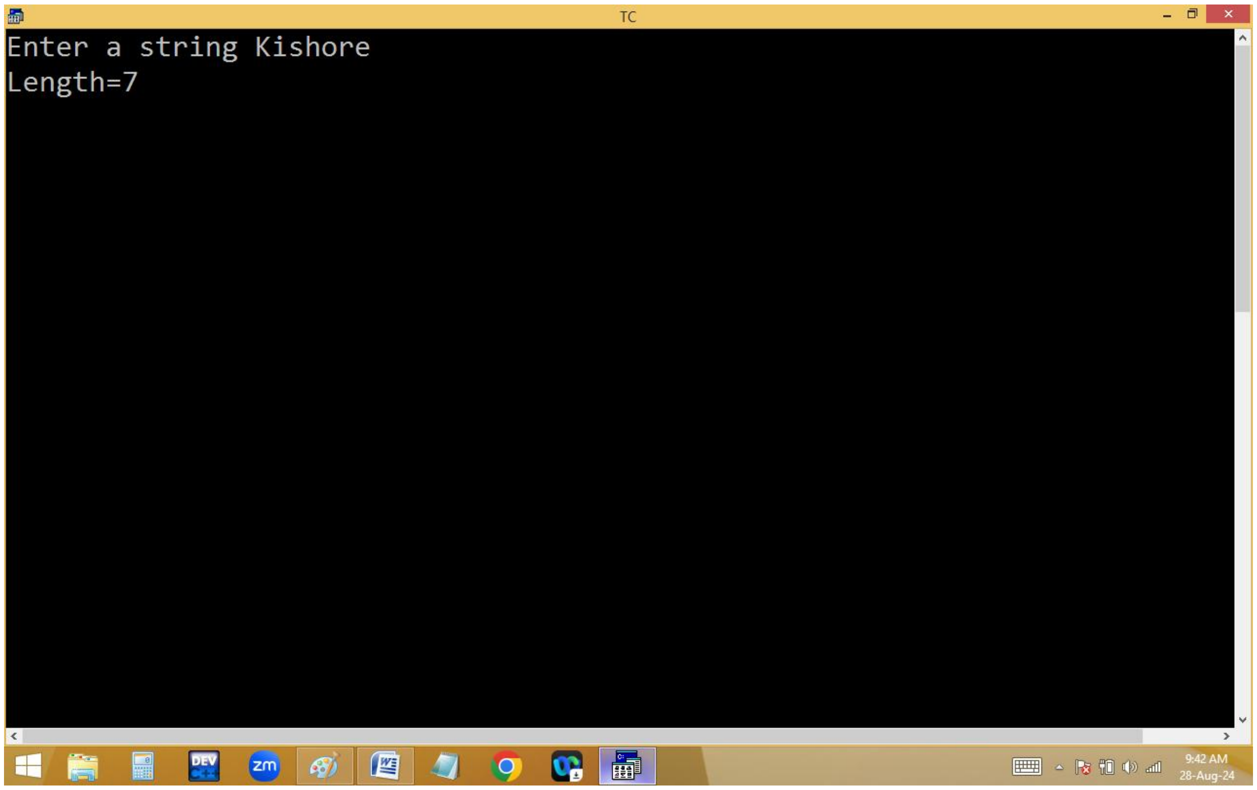


```
for( ;  *p!='\0' ; p++ )  puts(  p );
                              65500 to \0 ==> abcd

                              65501 to \0 ==> bcd
                              65502 to \0 ==> cd
                              65503 to \0 ==> d
                              65504 = '\0' != '\0' ==> false
```

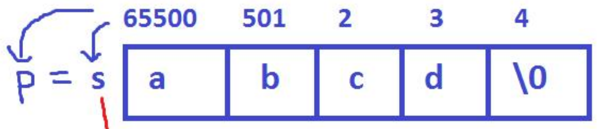
Find string length using pointers only?


TC
File Edit Run Compile Project Options Debug Break/watch
Line 10 Col 25 Insert Indent Tab Fill Unindent * E:9AM.C
9:42 AM
28-Aug-24

```c
#include<stdio.h>
#include<conio.h>
void main()
{
char s[100], *p;
clrscr();
printf("Enter a string "); gets(s);
p=s;  /* p=&s[0]; */
for( ; *p ;p++ );
printf("Length=%d",p-s);
getch();
}
```



```
Enter a string Kishore
Length=7
```



for(; *p!='\0' ; p++) puts(p);

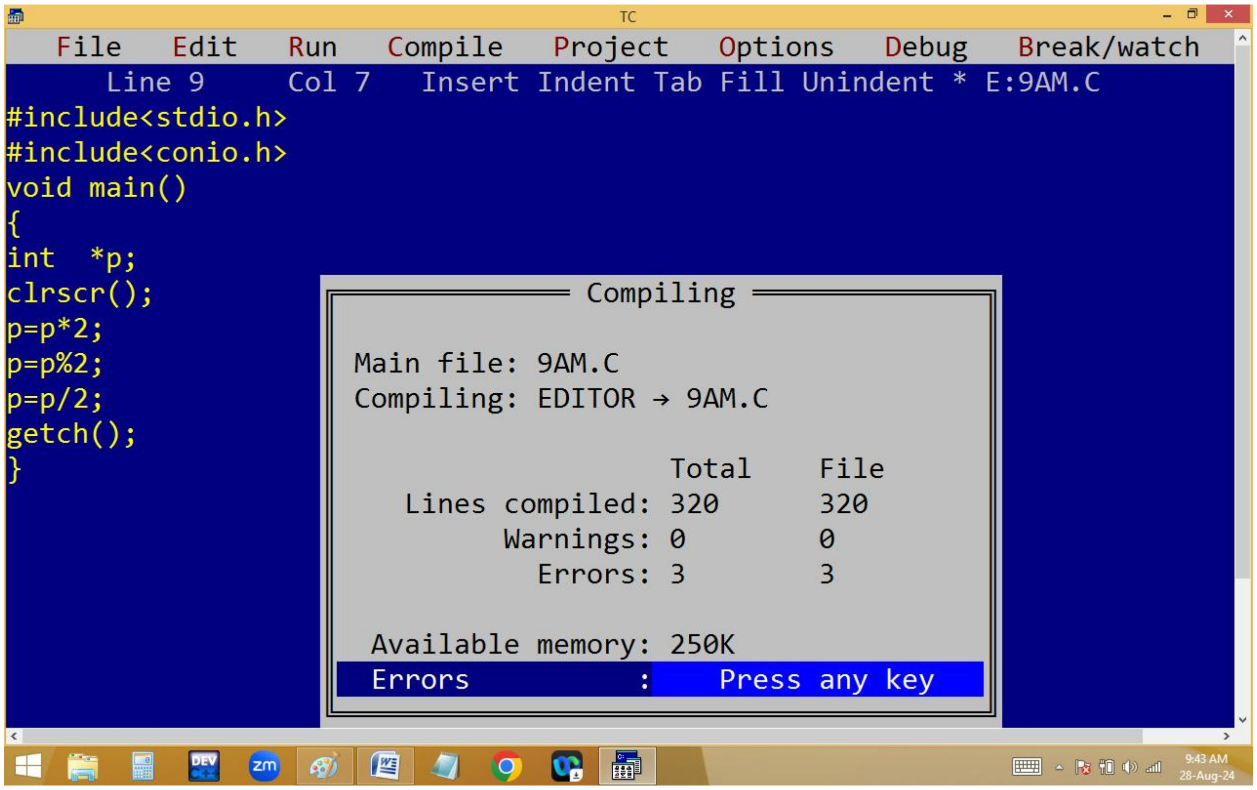
65500 to \0 ==> abcd

65501 to \0 ==> bcd
65502 to \0 ==> cd
65503 to \0 ==> d
65504 = '\0' != '\0' ==> false

p("Length=%d", p-s);

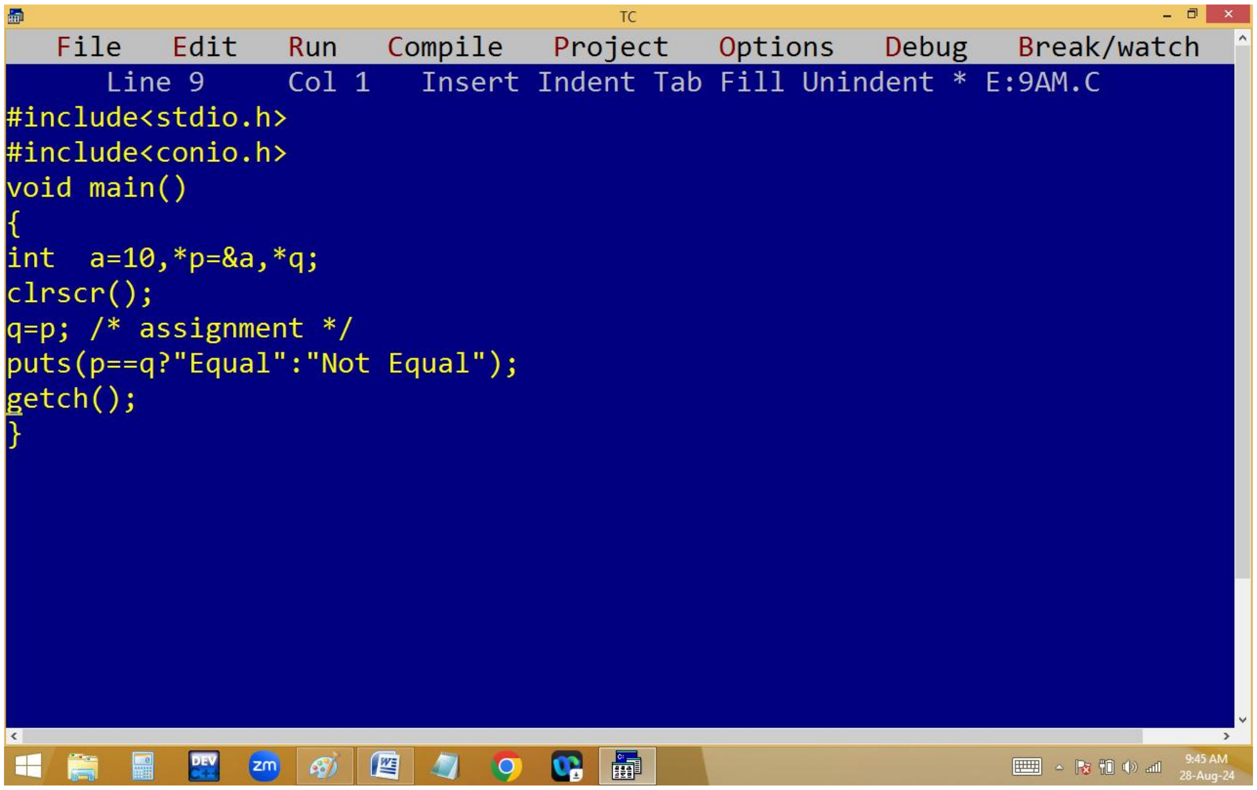
Pointer arithmetic:

Like normal variables we can do +,-,++,--,=,== on pointers also. But we can't do *, %, / on pointers.

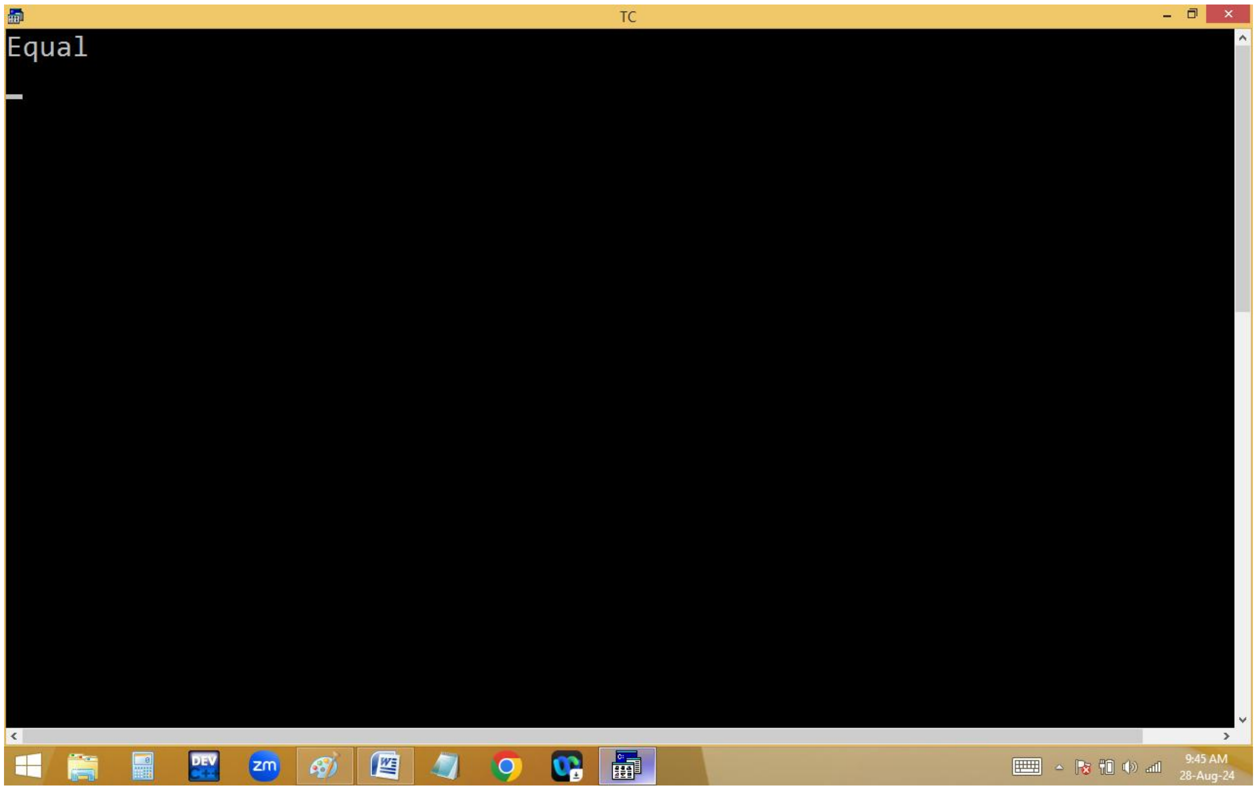

TC
File Edit Run Compile Project Options Debug Break/watch
Line 9 Col 7 Insert Indent Tab Fill Unindent * E:9AM.C

```
#include<stdio.h>
#include<conio.h>
void main()
{
int  *p;
clrscr();
p=p*2;
p=p%2;
p=p/2;
getch();
}
```

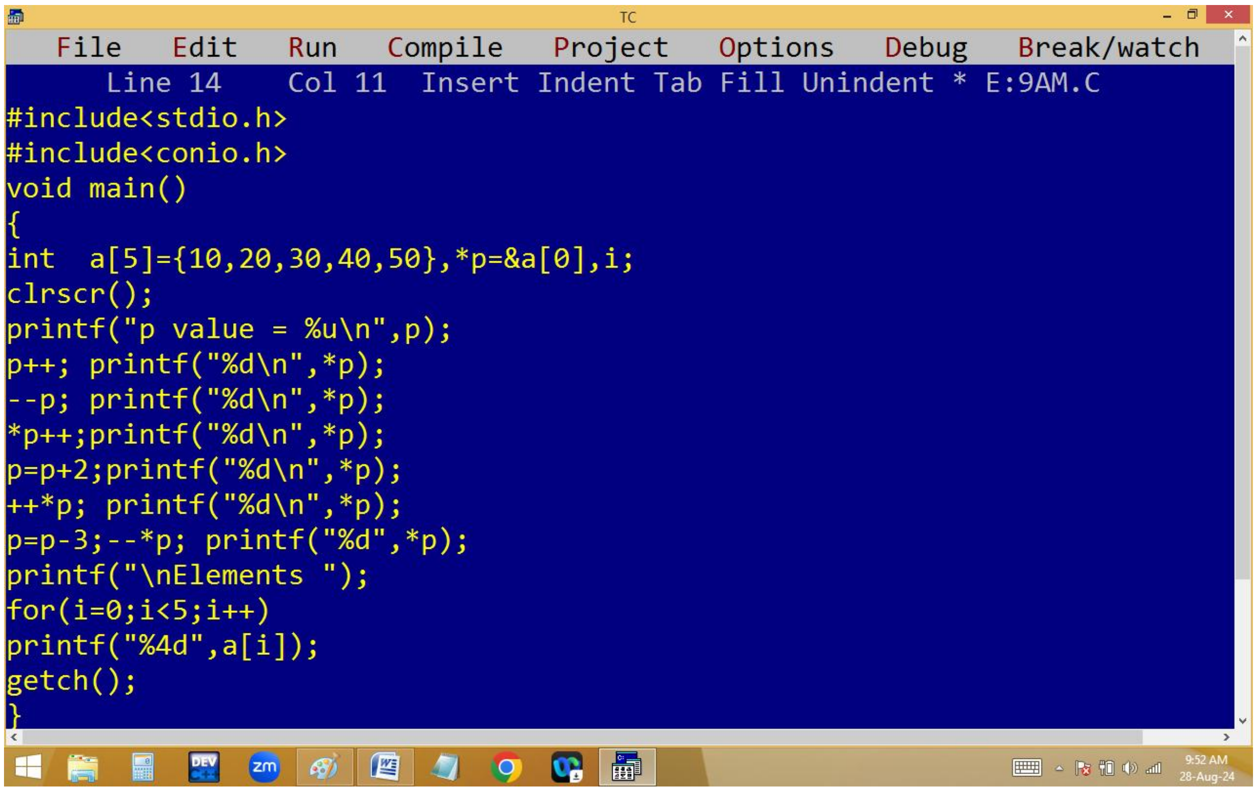
```
Compiling

Main file: 9AM.C
Compiling: EDITOR → 9AM.C

                  Total     File
   Lines compiled: 320      320
         Warnings: 0        0
           Errors: 3        3

 Available memory: 250K
 Errors          :    Press any key
```


TC
File Edit Run Compile Project Options Debug Break/watch
Line 9 Col 1 Insert Indent Tab Fill Unindent * E:9AM.C
9:45 AM
28-Aug-24

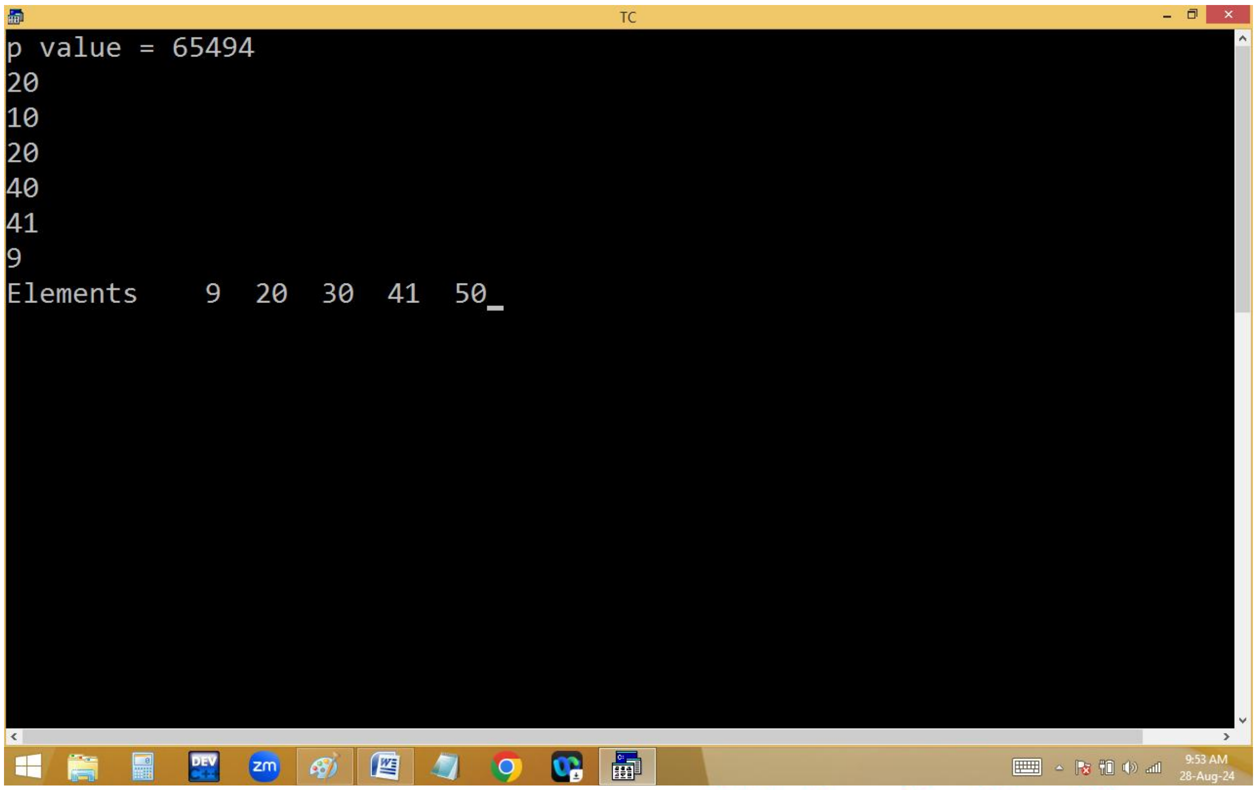
```
#include<stdio.h>
#include<conio.h>
void main()
{
int  a=10,*p=&a,*q;
clrscr();
q=p; /* assignment */
puts(p==q?"Equal":"Not Equal");
getch();
}
```



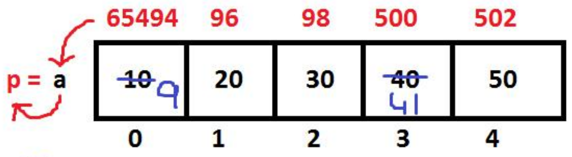
```
Equal
```


TC
File Edit Run Compile Project Options Debug Break/watch
Line 14 Col 11 Insert Indent Tab Fill Unindent * E:9AM.C
9:52 AM
28-Aug-24

```
#include<stdio.h>
#include<conio.h>
void main()
{
int  a[5]={10,20,30,40,50},*p=&a[0],i;
clrscr();
printf("p value = %u\n",p);
p++; printf("%d\n",*p);
--p; printf("%d\n",*p);
*p++;printf("%d\n",*p);
p=p+2;printf("%d\n",*p);
++*p; printf("%d\n",*p);
p=p-3;--*p; printf("%d",*p);
printf("\nElements ");
for(i=0;i<5;i++)
printf("%4d",a[i]);
getch();
}
```

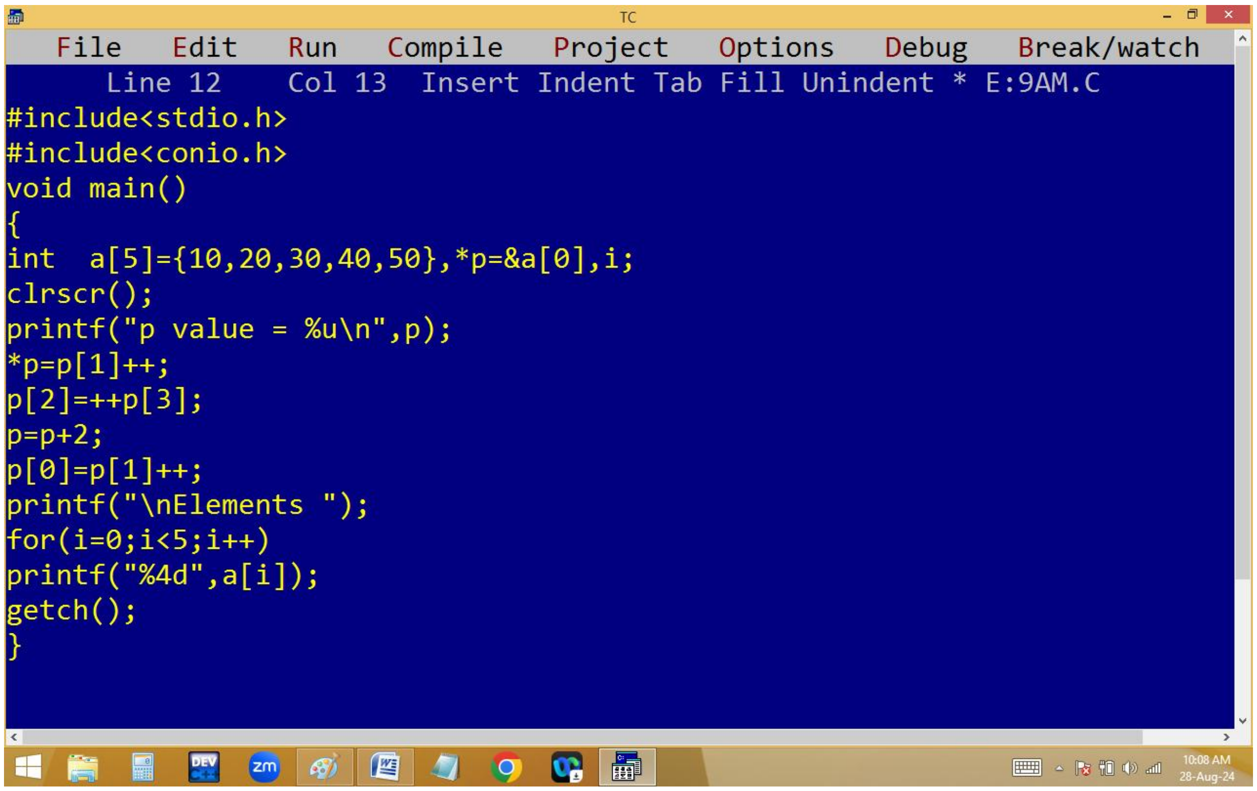


```
p value = 65494
20
10
20
40
41
9
Elements    9  20  30  41  50_
```

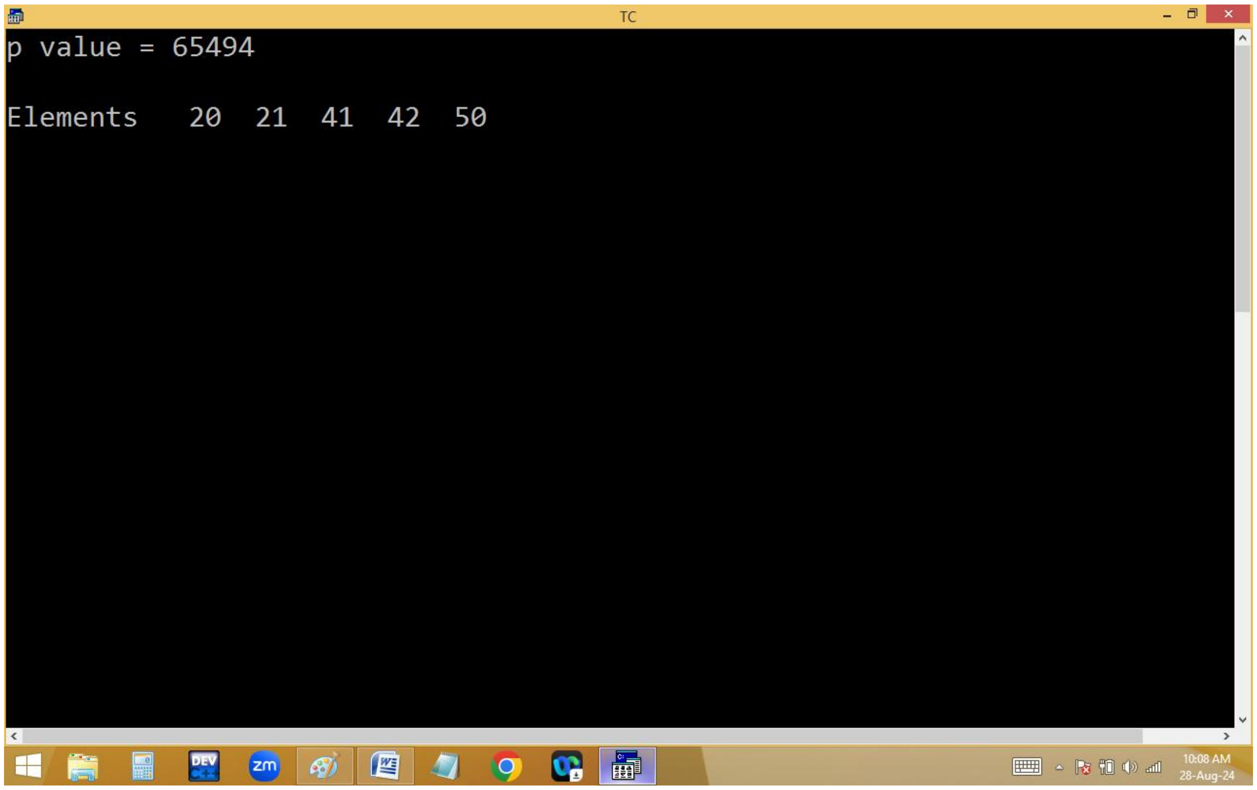


p++; 65494+1*2=65496; p(*p)==>value at 65496==>20
--p; 65496-1*2=65494; p(*p)==>value at 65494==>10
*p++==>65494+1*2=65496; p(*p)==>value at 65496=20
p=p+2==>65496+2*2=65500;p(*p)==>value at 65500==> 40
++*p==>value at 65500++ ==> 41 ; p(*p)==>value at 65500==> 41

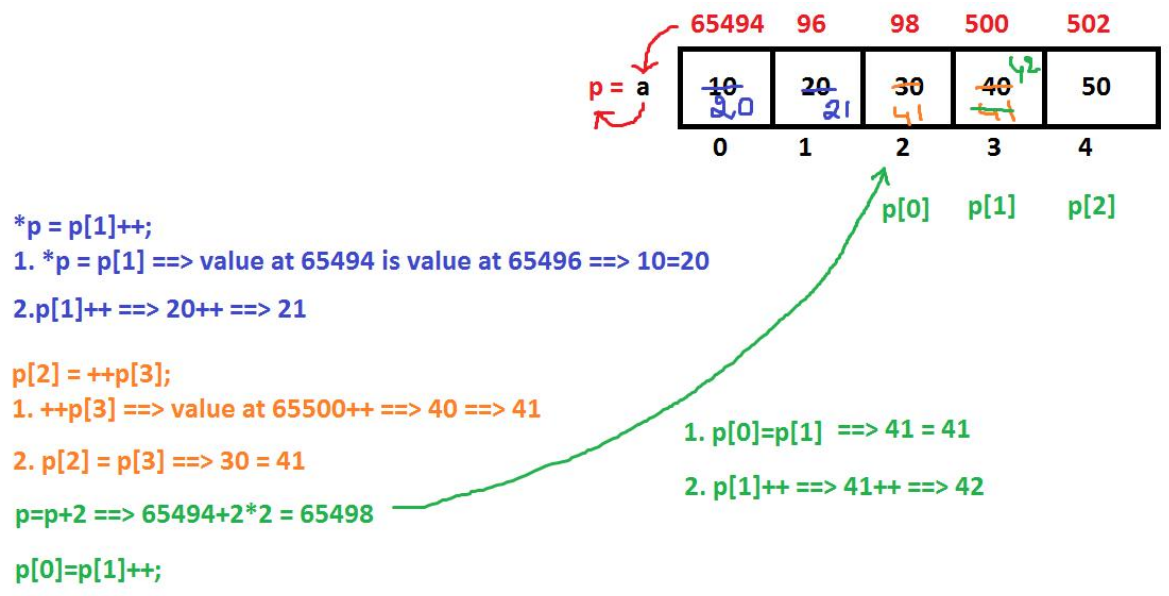
p=p-3==>65500-3*2=65494; --*p ==>value at 65494-- ==> 10-- ==> 9 ; p(*p)==>value at 65494 ==> 9


TC
File Edit Run Compile Project Options Debug Break/watch
Line 12 Col 13 Insert Indent Tab Fill Unindent * E:9AM.C

```c
#include<stdio.h>
#include<conio.h>
void main()
{
int  a[5]={10,20,30,40,50},*p=&a[0],i;
clrscr();
printf("p value = %u\n",p);
*p=p[1]++;
p[2]=++p[3];
p=p+2;
p[0]=p[1]++;
printf("\nElements ");
for(i=0;i<5;i++)
printf("%4d",a[i]);
getch();
}
```



```
p value = 65494

Elements   20  21  41  42  50
```



*p = p[1]++;
1. *p = p[1] ==> value at 65494 is value at 65496 ==> 10=20
2.p[1]++ ==> 20++ ==> 21

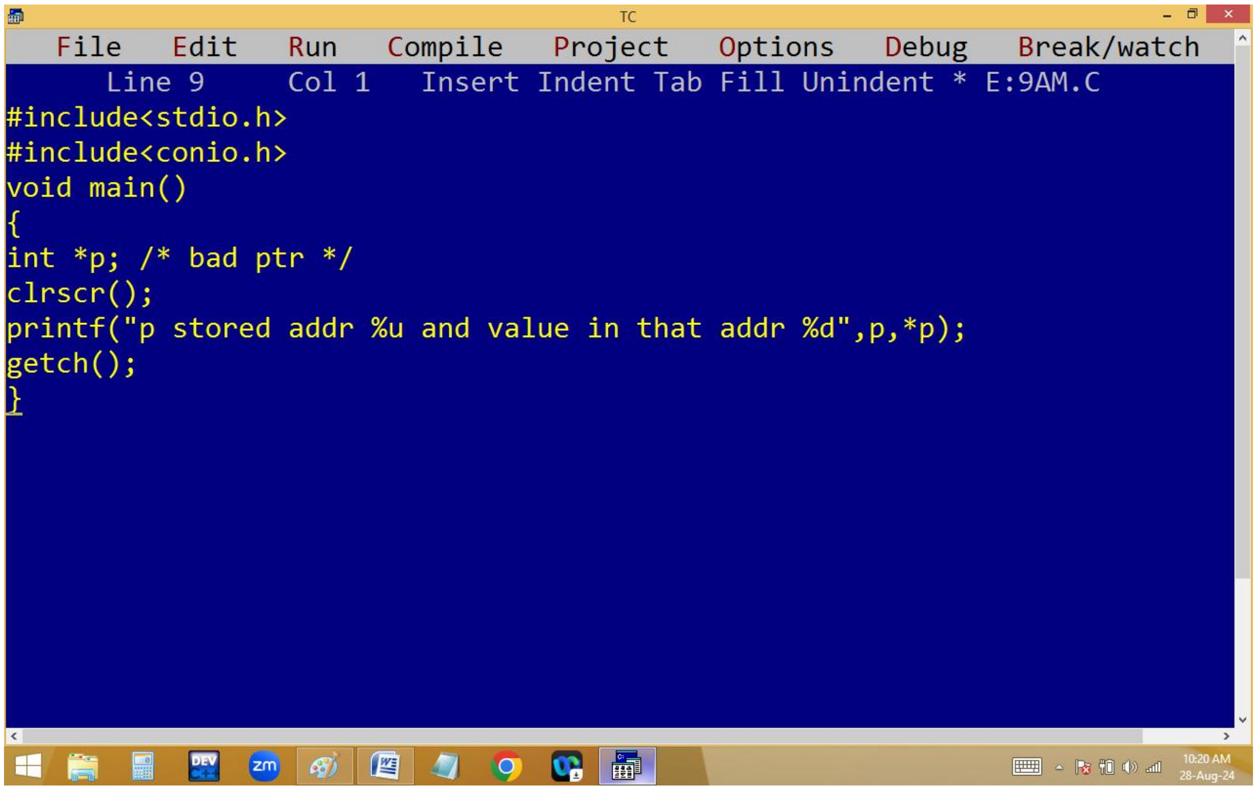
p[2] = ++p[3];
1. ++p[3] ==> value at 65500++ ==> 40 ==> 41
2. p[2] = p[3] ==> 30 = 41

p=p+2 ==> 65494+2*2 = 65498

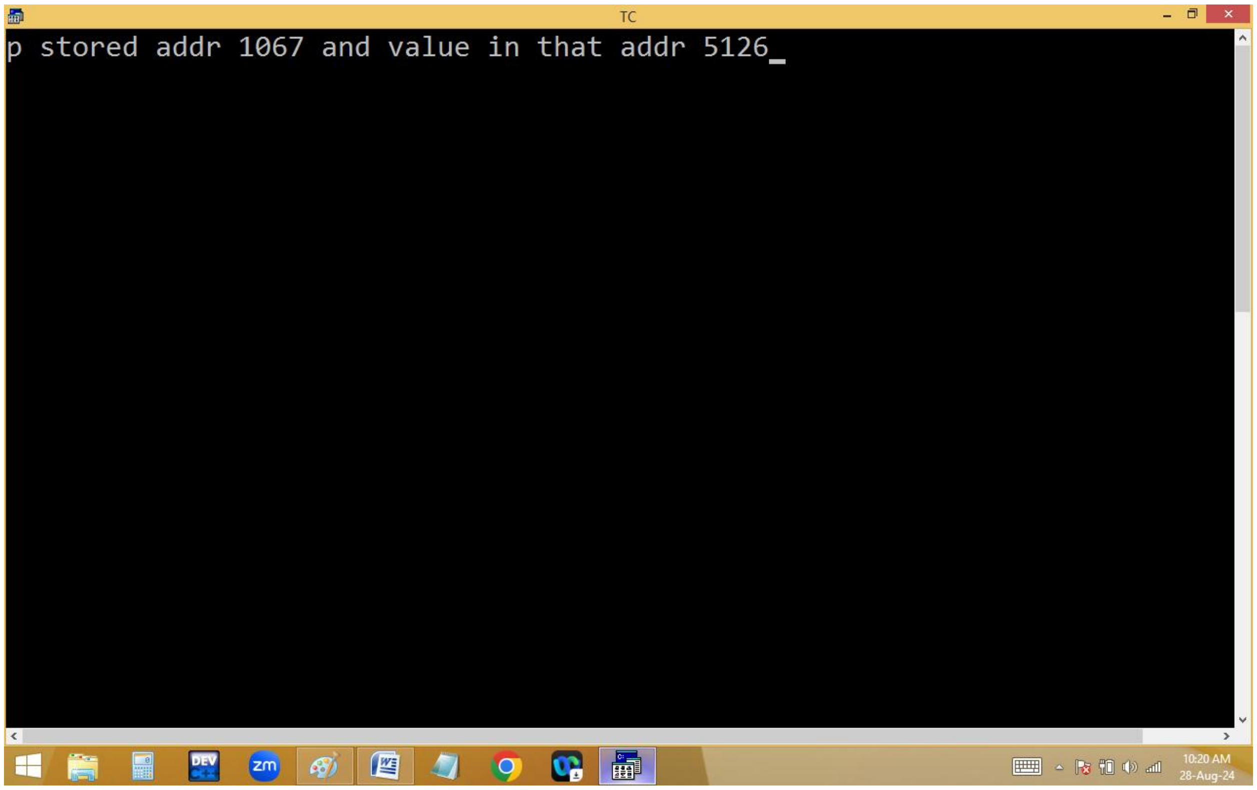
p[0]=p[1]++;

1. p[0]=p[1] ==> 41 = 41
2. p[1]++ ==> 41++ ==> 42

Bad/wild pointer: A pointer is declared but not initialized. In this situation the pointer is storing some unknown address and value. This kind of pointer is called bad / wild pointer.

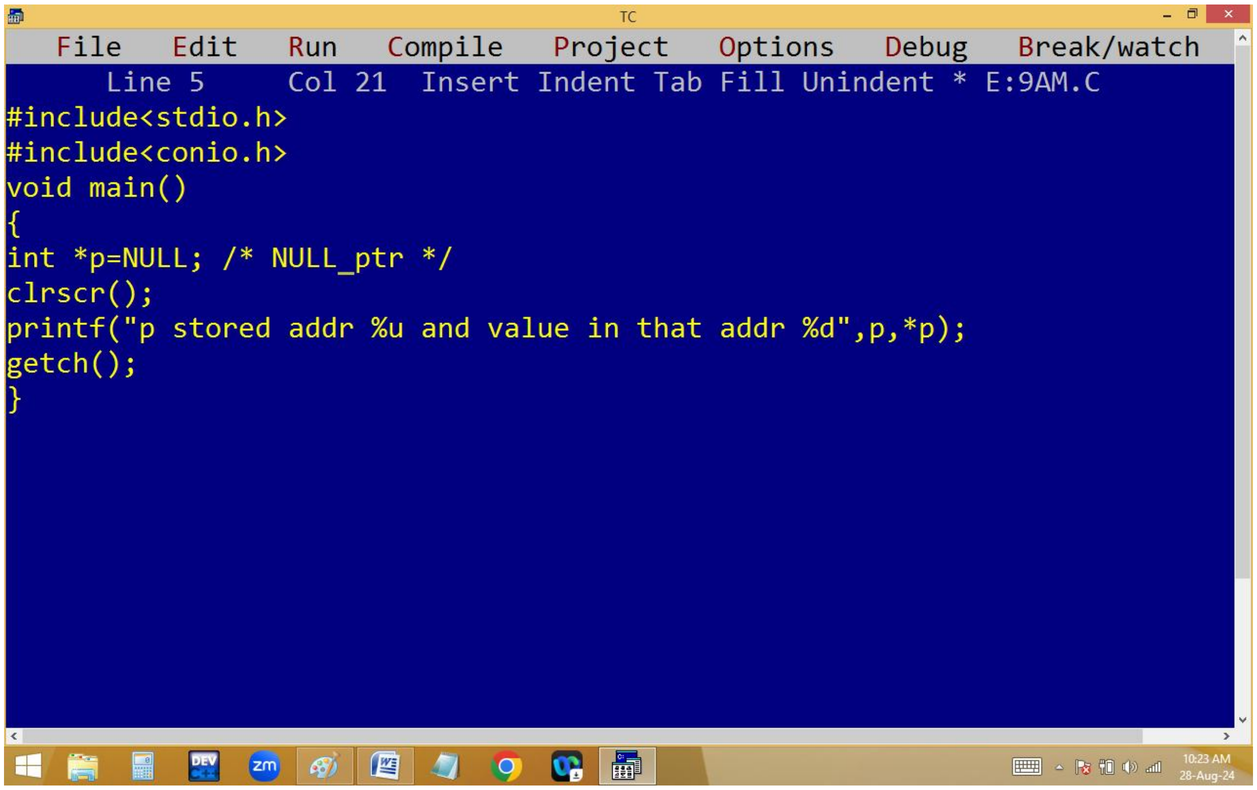

TC
File Edit Run Compile Project Options Debug Break/watch
Line 9 Col 1 Insert Indent Tab Fill Unindent * E:9AM.C
10:20 AM
28-Aug-24

```
#include<stdio.h>
#include<conio.h>
void main()
{
int *p; /* bad ptr */
clrscr();
printf("p stored addr %u and value in that addr %d",p,*p);
getch();
}
```

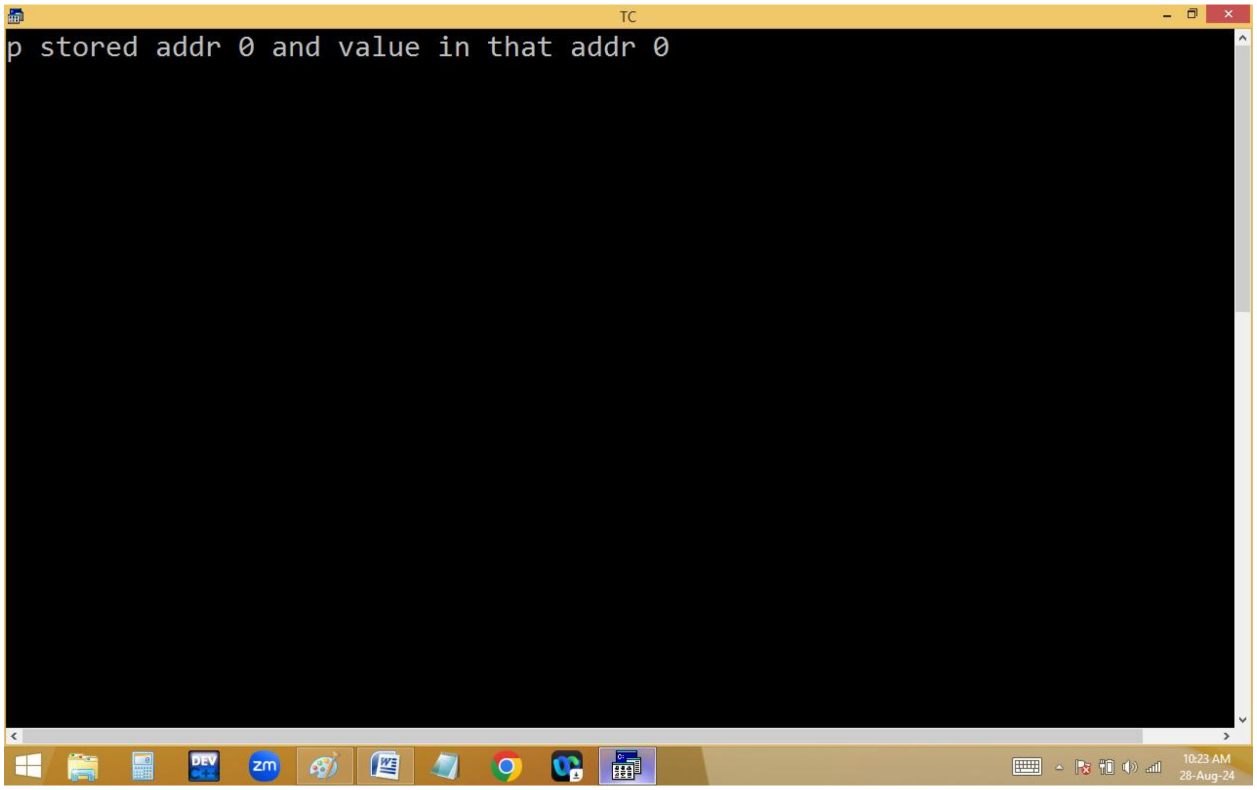


```
p stored addr 1067 and value in that addr 5126_
```

NULL pointer: When a pointer initialized with 0 or NULL then it is a NULL pointer. To avoid bad and dangling pointers we are using NULL pointer.


TC
File Edit Run Compile Project Options Debug Break/watch
Line 5 Col 21 Insert Indent Tab Fill Unindent * E:9AM.C
10:23 AM
28-Aug-24

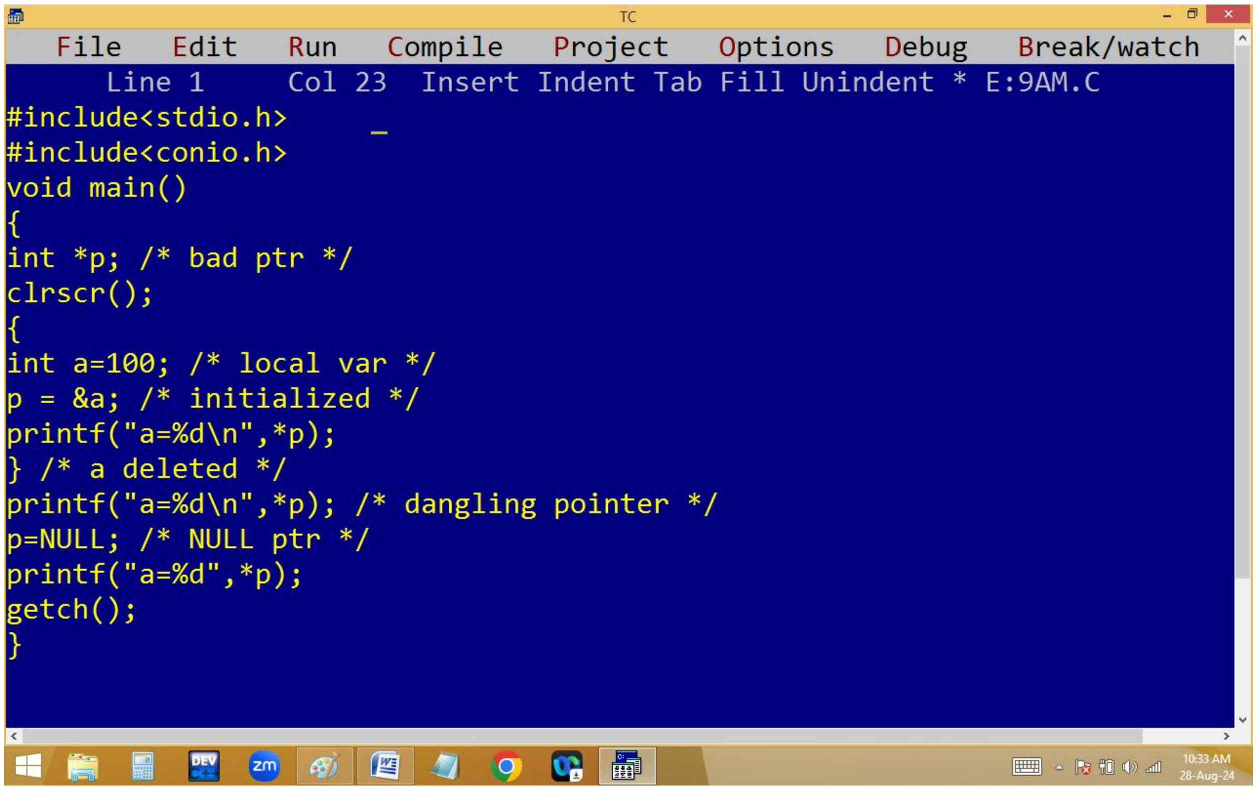
```
#include<stdio.h>
#include<conio.h>
void main()
{
int *p=NULL; /* NULL_ptr */
clrscr();
printf("p stored addr %u and value in that addr %d",p,*p);
getch();
}
```



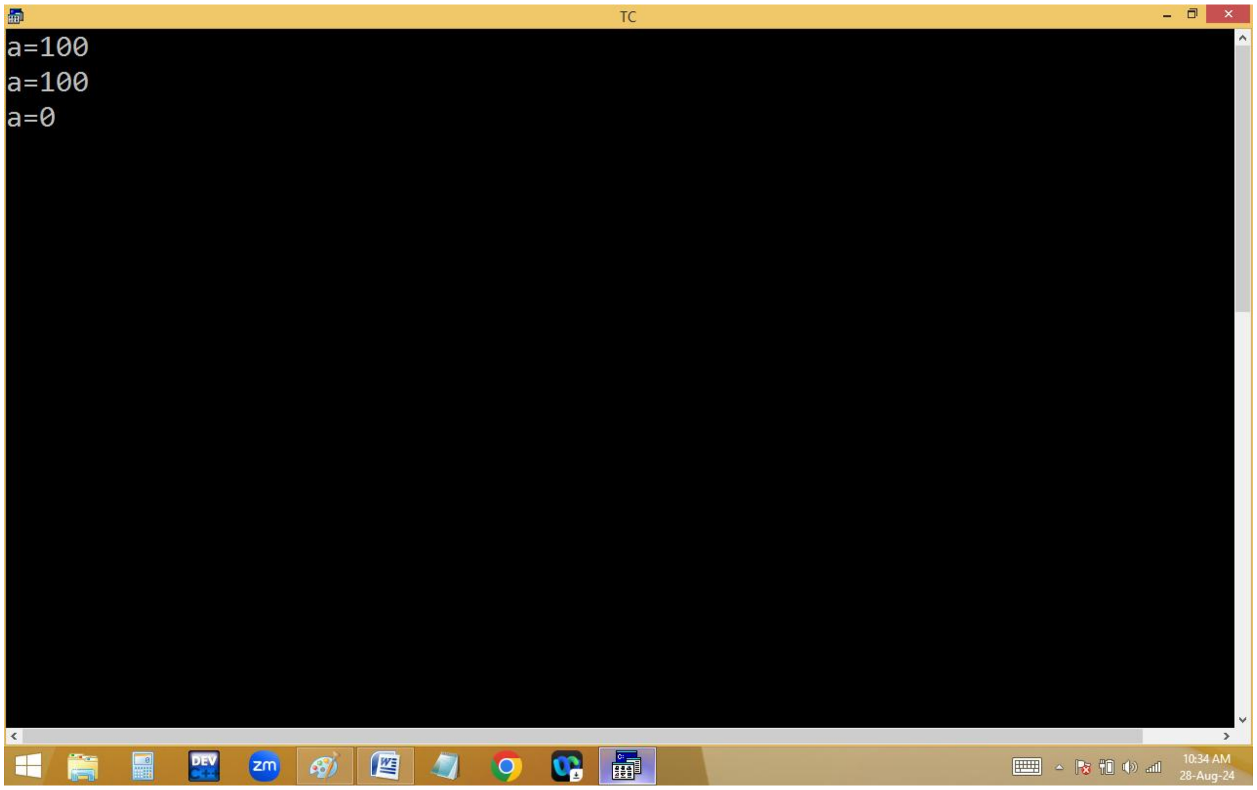
```
p stored addr 0 and value in that addr 0
```

Dangling pointer:

A pointer is declared and later some address also assigned. After some time that variable deleted from memory. But still the pointer is storing the deleted variable address. This kind of pointer is called dangling pointer. To avoid this initialize with NULL pointer.

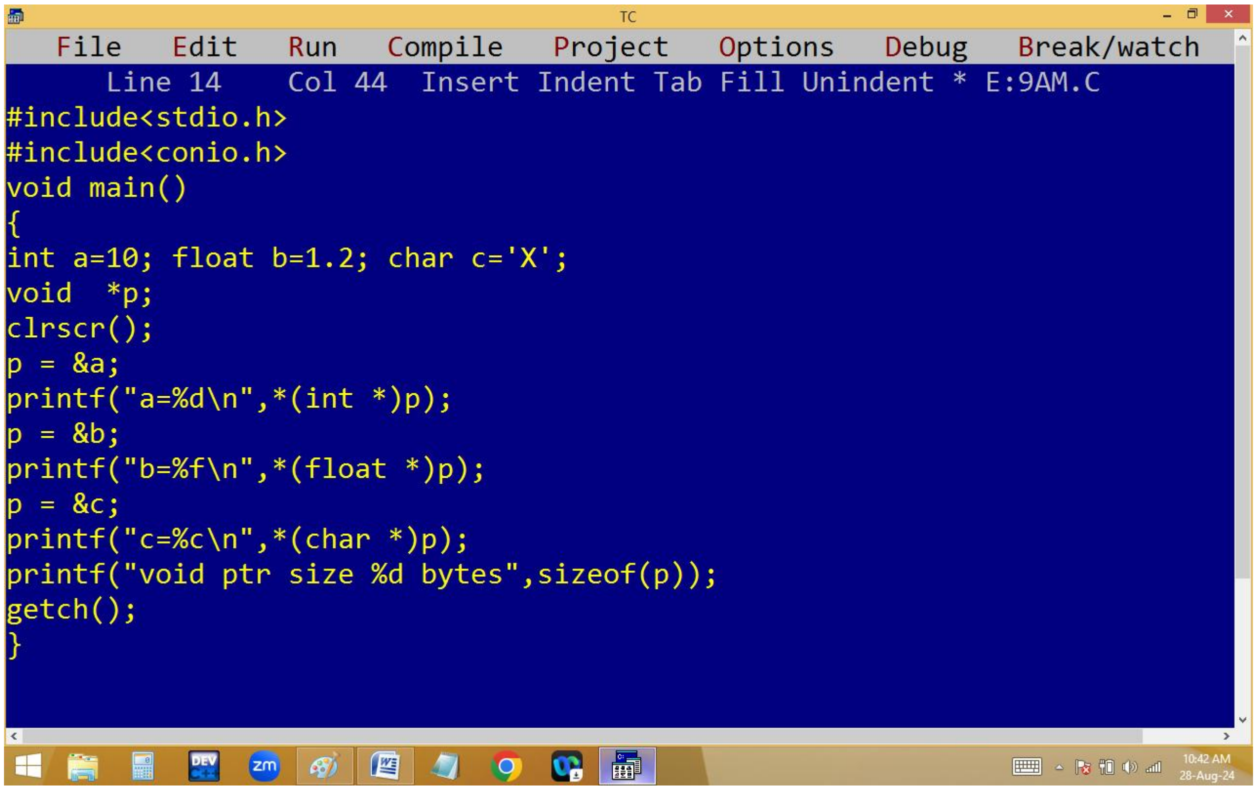

TC
File Edit Run Compile Project Options Debug Break/watch
Line 1 Col 23 Insert Indent Tab Fill Unindent * E:9AM.C
10:33 AM
28-Aug-24

```c
#include<stdio.h>
#include<conio.h>
void main()
{
int *p; /* bad ptr */
clrscr();
{
int a=100; /* local var */
p = &a; /* initialized */
printf("a=%d\n",*p);
} /* a deleted */
printf("a=%d\n",*p); /* dangling pointer */
p=NULL; /* NULL ptr */
printf("a=%d",*p);
getch();
}
```

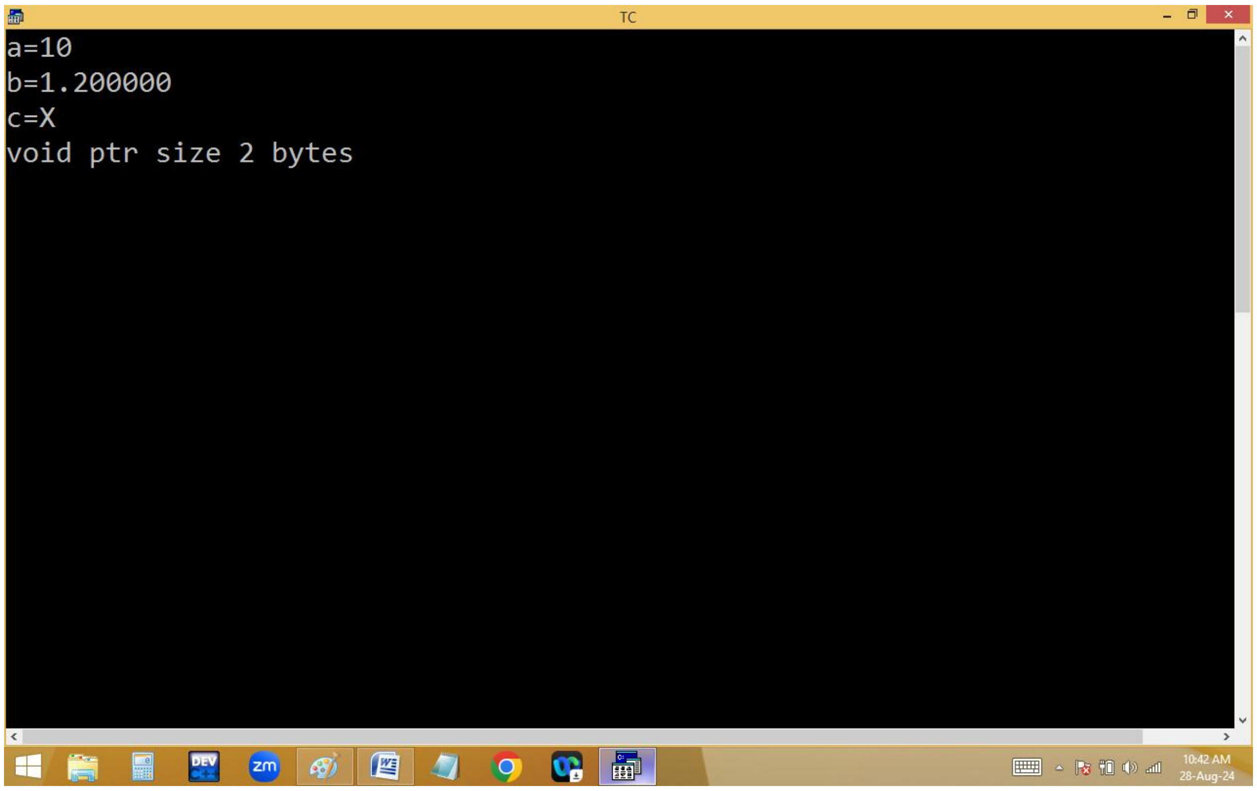


```
a=100
a=100
a=0
```

void / generic pointer: void pointer can store any type of variable address and it is used to handle dynamic arrays. It takes 2 bytes. Before going to use void pointer, explicit type casting should be provided.


TC
File Edit Run Compile Project Options Debug Break/watch
Line 14 Col 44 Insert Indent Tab Fill Unindent * E:9AM.C
10:42 AM
28-Aug-24

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10; float b=1.2; char c='X';
void  *p;
clrscr();
p = &a;
printf("a=%d\n",*(int *)p);
p = &b;
printf("b=%f\n",*(float *)p);
p = &c;
printf("c=%c\n",*(char *)p);
printf("void ptr size %d bytes",sizeof(p));
getch();
}
```



```
a=10
b=1.200000
c=X
void ptr size 2 bytes
```