TC

File Edit Run Compile Project Options Debug

Line 18 Col 11 Insert Indent Tab Fill Unindent *

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[5]={10,20,30,40,50}, *p=a,i;
clrscr();
printf("array addr %u\n",a);
p++; printf("%d\n",*p);
*p++; printf("%d\n",*p);
(*p)++;printf("%d\n",*p);
p-=2;
--*p;printf("%d\n",*p);
p=p+3;
*p=100;
printf("%d\n",*p);
puts("elements");
for(i=0;i<5;i++)
printf("%4d",a[i]);
getch();
}
```

9:26 AM
11-Dec-24

```
array addr 65494
20
30
31
9
100
elements
   9   20   31  100   50
```

p++==>65494+1*2=65496

p(*p)==>value at 65496==>20 ✓

*p++==>65496+1*2=65498

p(*p)==>value at 65498==>30 ✓

(*p)++==>value at 65498++==>30++ ==> 31

p(*p)==>value at 65498 ==> 31 ✓

p=p-2; 65498-2*2=65494

--*p==>value at 65494-- ==> 10-- ==> 9

p(*p)==>value at 65494==> 9 ✓

p=p+3;65494+3*2=65500

*p=100

value at 65500=100 ==> 40=100

p(*p)==>value at 65500 ==> 100 ✓

	65494	96	98	500	502
p=a	9 ~~10~~	20	31 ~~30~~	100 ~~40~~	50
	0	1	2	3	4

TC

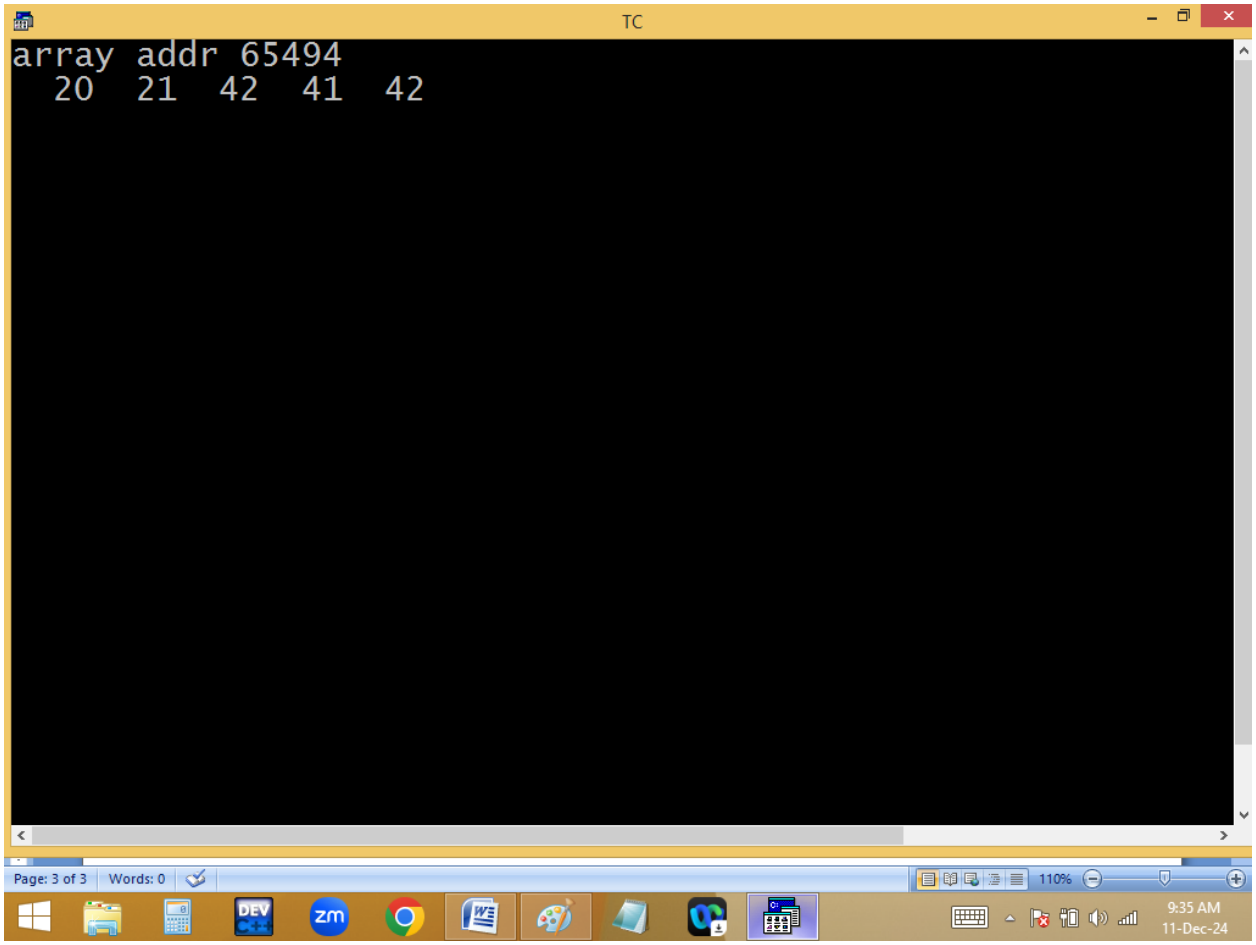
File Edit Run Compile Project Options Debug

Line 11 Col 11 Insert Indent Tab Fill Unindent *

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

Page: 2 of 2 Words: 0 110%

9:35 AM
11-Dec-24



p[0] = p[1]++; ⟨ p[0]=p[1] / p[1]++

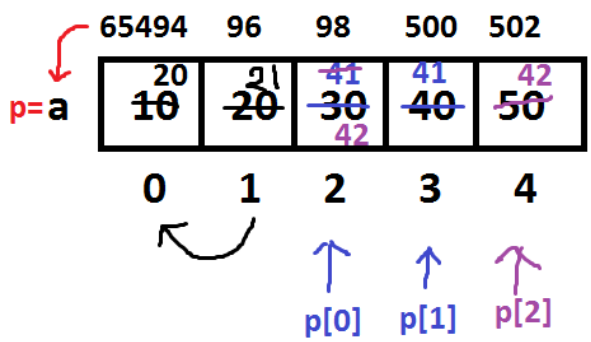
65494 = 65496 ==> 10=20

p[1]++ ==> 20++ ==> 21

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

*p = ++p[1]; ⟨ ++p[1] ==> 40++ ==> 41 / *p = p[1] ==> p[0]=p[1]==>30=41

p[2]=++*p; ⟨ ++*p ==> 41++ ==> 42 / p[2]=42



MEMORY MANAGEMENT

To store anything in our computer, we should have to allocate the memory first.

This memory allocation is conducted in two ways.

1. Static memory allocation.
2. Dynamic memory allocation.

In static memory allocation, the memory specified at compile/design time, based on the data type or array size. This type of memory management is called compile time memory management [**compiler indicates memory and O.S allocates the memory**].

In static memory allocation, the memory size is fixed at compile time and we can't

change this memory size at run time. It causes some times memory wastage / shortage.

To avoid this problem, the only solution is dynamic memory allocation.

In dynamic memory allocation, the memory is allocated at run time, based on the user input, instantly.
This type of memory management is called run time memory management.

To conduct dynamic memory allocation, we should have to use **pointers**.

In dynamic memory allocation the memory is allocated in **HEAP** area.

To manage the dynamic memory, we are using some predefined functions like

- malloc()
- calloc()
- realloc()
- free()

All these functions are available in **<alloc.h>**

malloc(), realloc(), calloc() functions are able to allocate the memory of **64KB** Maximum at a time.

To allocate more than 64KB memory, use the functions

- farmalloc()
- farcalloc()
- farrealloc().

Note:

when we are working with dynamic memory allocation, we have to allocate the

memory for any data type. Due to this all these functions return datatype is **void ***, which is a generic type. Due to this we should have to provide **explicit type casting** for all these functions.

malloc()	**calloc()**
Block Memory allocation	Contiguous memory blocks allocation
Allocates memory in bytes form	Allocates memory in blocks form.
Initial values garbage	Initial values 0
One argument required	Two arguments required
Used for normal variables	Used for array type variables

Syntax:

```
void  *  malloc(bytes);

void  * calloc(no of blocks, block_size);
```

free(): It is used to release the memory allocated by malloc(), calloc() and realloc().

Syntax: void free(pointer);

realloc(): It is used to extend the memory allocated by malloc() or calloc() at runtime. Working style is similar to malloc().

Syntax: **void * realloc(oldptr, newsize);**

free example:

TC

File Edit Run Compile Project Options Debug Break/watch

Line 10 Col 9 Insert Indent Tab Fill Unindent * E:NONAME.C

```
#include<stdio.h>
#include<conio.h>
#include<alloc.h> #include<stdlib.h>
int a;
void main()
{
int *p, *q;
p = (int *)malloc(10);
printf("p=%u\n",p);
free(p);
q = (int *)malloc(10);
printf("q=%u",q);
getch();
}
```


TC

```
p=2358
q=2358
```

TC

File Edit Run Compile Project Options Debug Break/watch

Line 10 Col 1 Insert Indent Tab Fill Unindent * E:NONAME.C

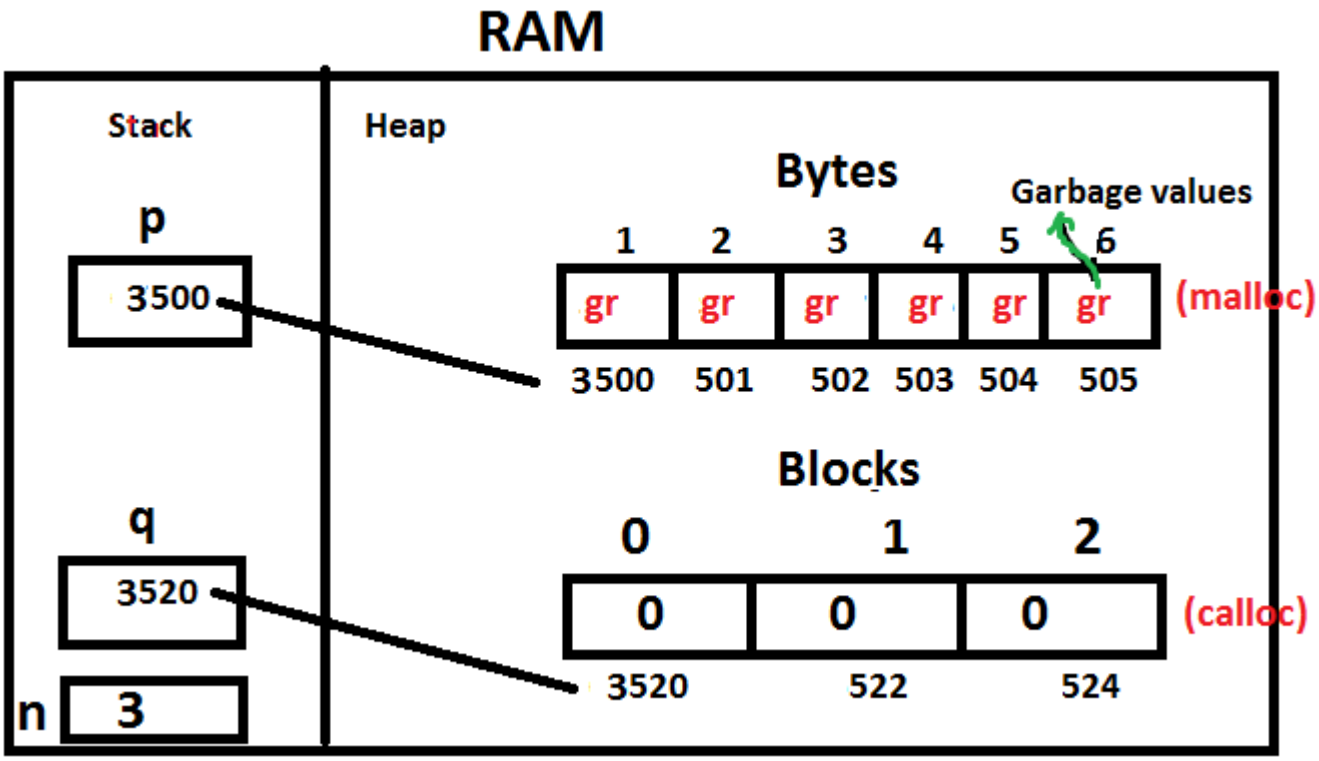
```
#include<stdio.h>
#include<conio.h>
#include<alloc.h> #include<stdlib.h>
int a;
void main()
{
int *p, *q;
p = (int *)malloc(10);
printf("p=%u\n",p);
q = (int *)malloc(10);
printf("q=%u",q);
getch();
}
```


TC

```
p=2358
q=2374_
```

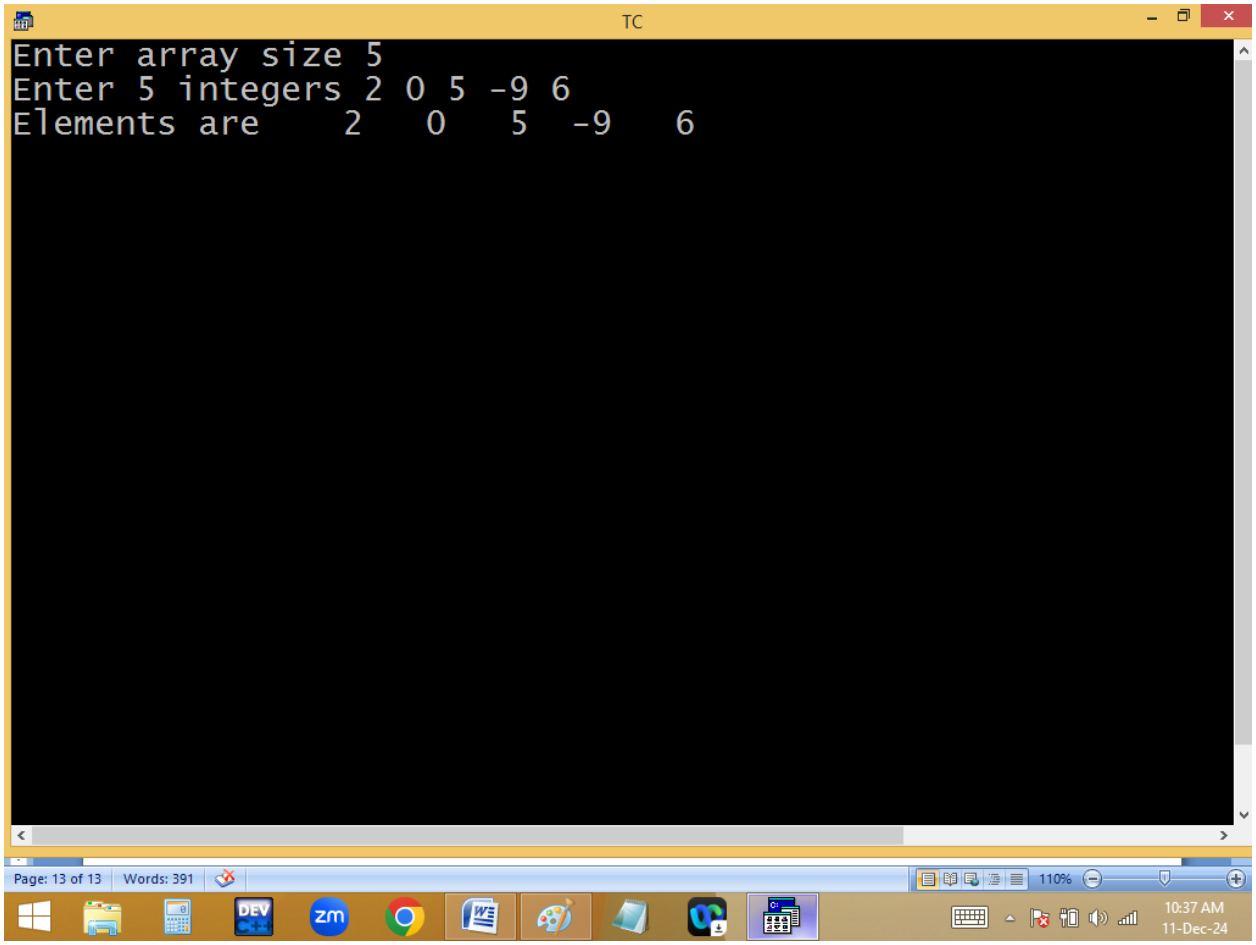
allocating memory for 3 integers using malloc(), calloc().

```
int *p, *q, n=3;

p = (int  *)malloc(n * sizeof(int));

q = (int  *)calloc(n ,  sizeof(int));
```



Creating dynamic one-dimensional array:

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>  #include<alloc.h>
void main()
{
int n, *p, i;
clrscr();
printf("Enter array size ");scanf("%d",&n);
p = (int *)malloc(n*sizeof(int));
printf("Enter %d integers ", n);
for(i=0;i<n;i++)scanf("%d",&p[i]);
printf("Elements are ");
for(i=0;i<n;i++)printf("%4d",p[i]);
free(p);
p=NULL;
getch();
}
```


TC
Enter array size 5
Enter 5 integers 2 0 5 -9 6
Elements are 2 0 5 -9 6
Page: 13 of 13
Words: 391
110%
10:37 AM
11-Dec-24

TC

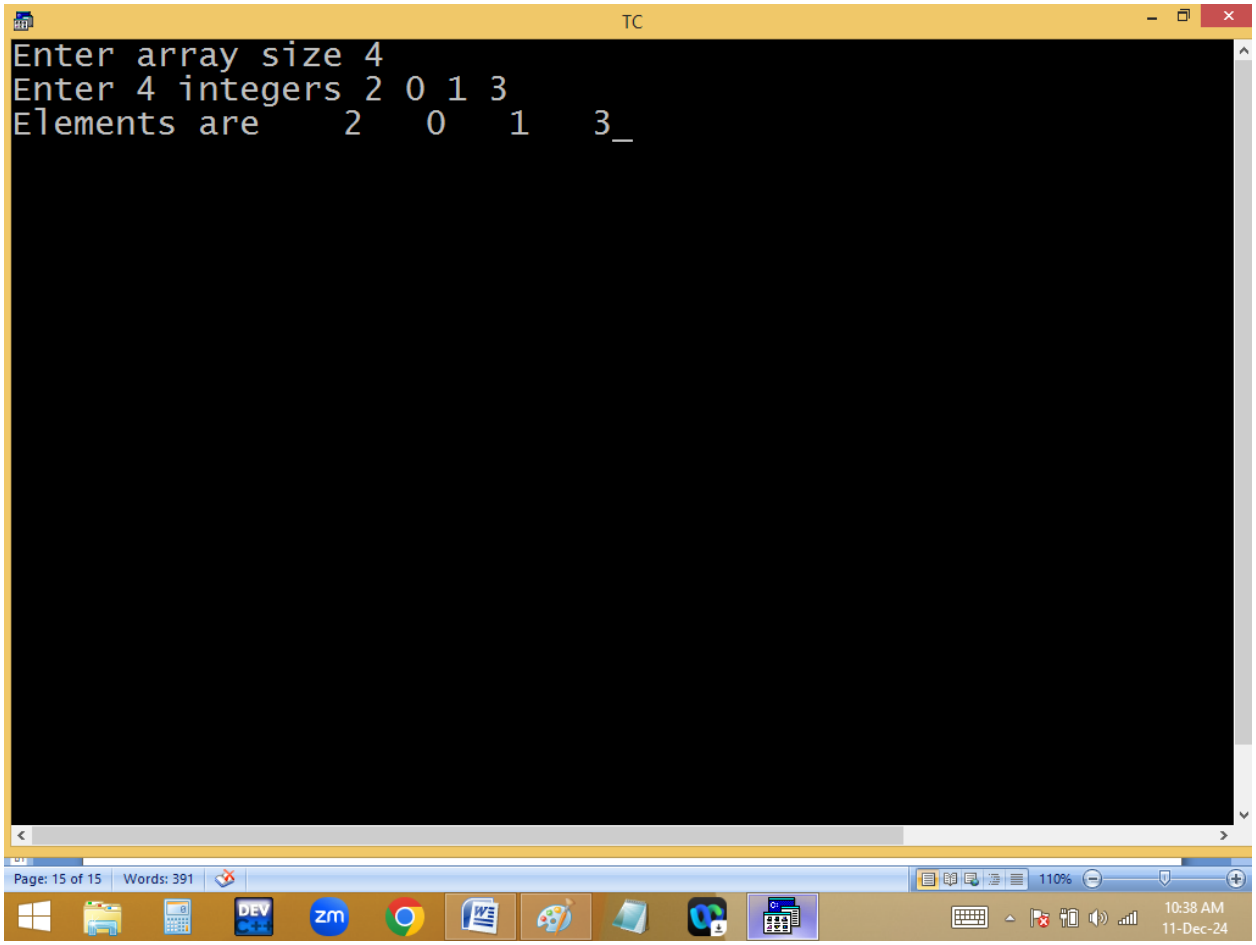
File Edit Run Compile Project Options Debug

Line 9 Col 21 Insert Indent Tab Fill Unindent *

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>  #include<alloc.h>
void main()
{
int n, *p, i;
clrscr();
printf("Enter array size ");scanf("%d",&n);
p = (int *)calloc(n,sizeof(int));
printf("Enter %d integers ", n);
for(i=0;i<n;i++)scanf("%d",&p[i]);
printf("Elements are ");
for(i=0;i<n;i++)printf("%4d",p[i]);
free(p);
p=NULL;
getch();
}
```

Page: 15 of 15 Words: 391 110%

10:38 AM
11-Dec-24


TC
Enter array size 4
Enter 4 integers 2 0 1 3
Elements are 2 0 1 3_
Page: 15 of 15
Words: 391
110%
10:38 AM
11-Dec-24

TC

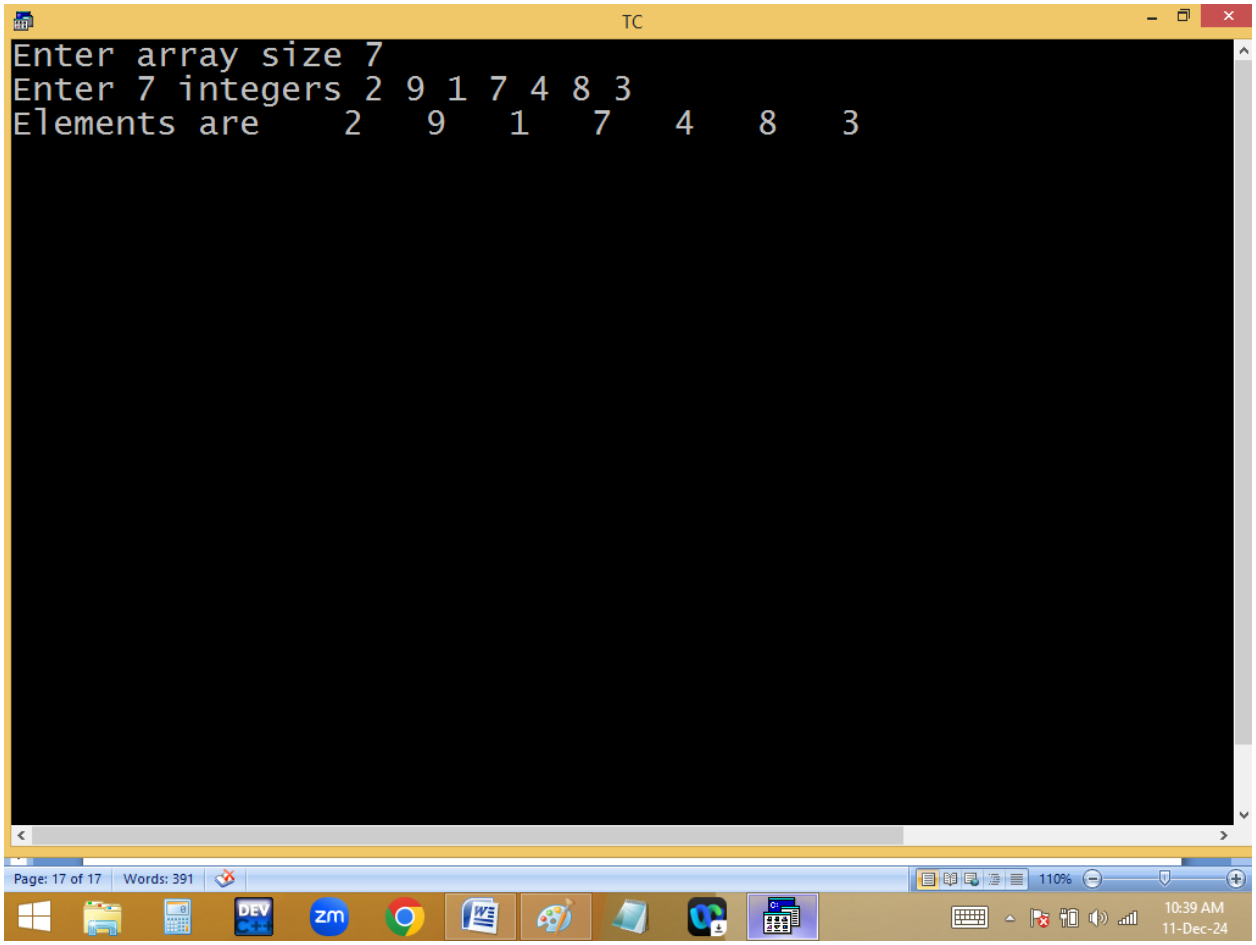
File Edit Run Compile Project Options Debug

Line 13 Col 36 Insert Indent Tab Fill Unindent *

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>  #include<alloc.h>
void main()
{
int n, *p, i;
clrscr();
printf("Enter array size ");scanf("%d",&n);
p = (int *)calloc(n,sizeof(int));
printf("Enter %d integers ", n);
for(i=0;i<n;i++)scanf("%d",p+i);
printf("Elements are ");
for(i=0;i<n;i++)printf("%4d",*(p+i));
free(p);
p=NULL;
getch();
}
```

Page: 16 of 16 Words: 391 110%

10:39 AM
11-Dec-24


TC
Enter array size 7
Enter 7 integers 2 9 1 7 4 8 3
Elements are 2 9 1 7 4 8 3
Page: 17 of 17
Words: 391
110%
10:39 AM
11-Dec-24