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);

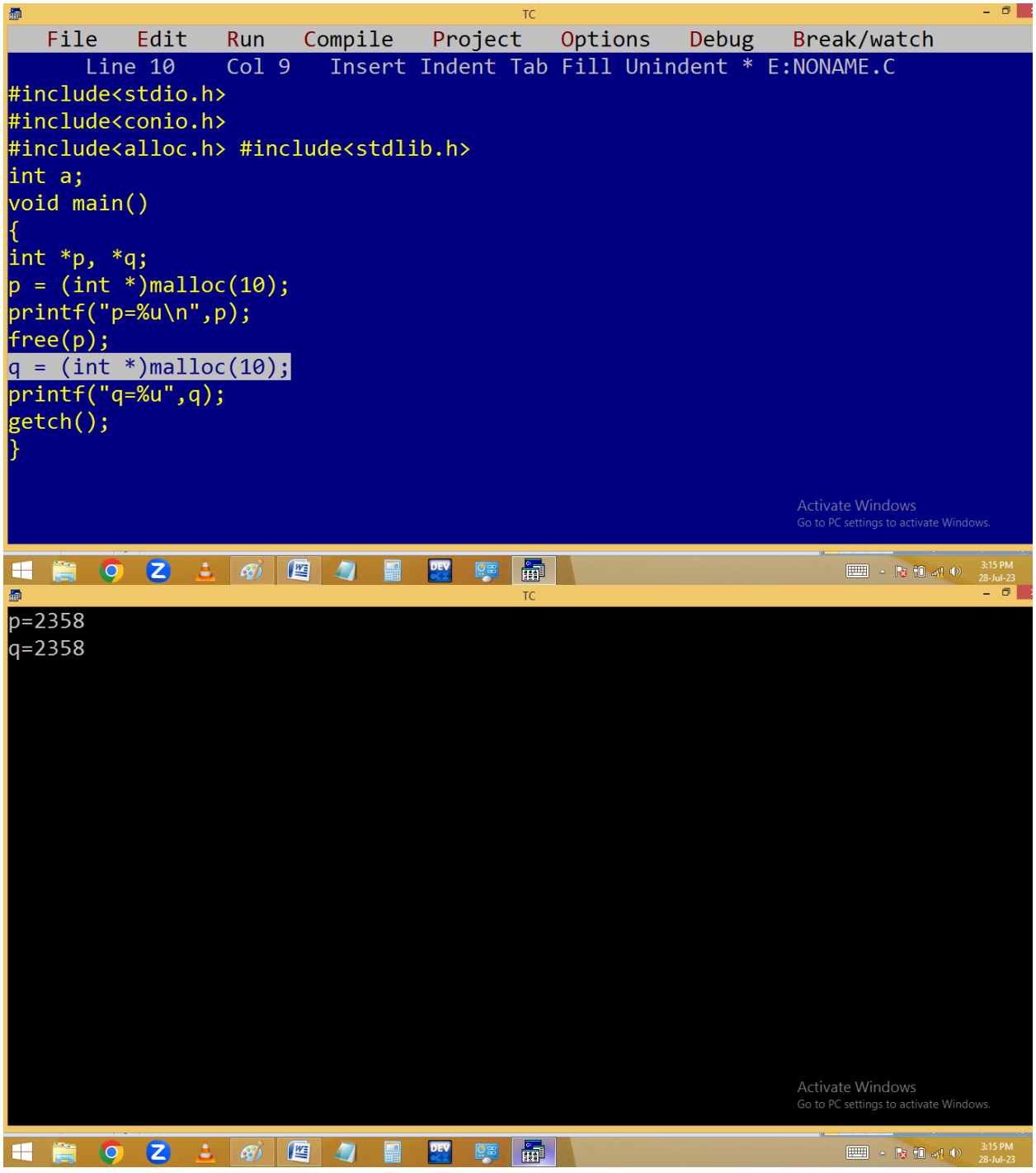
free(): It is used to release the memory allocated by malloc(), calloc() and realloc().

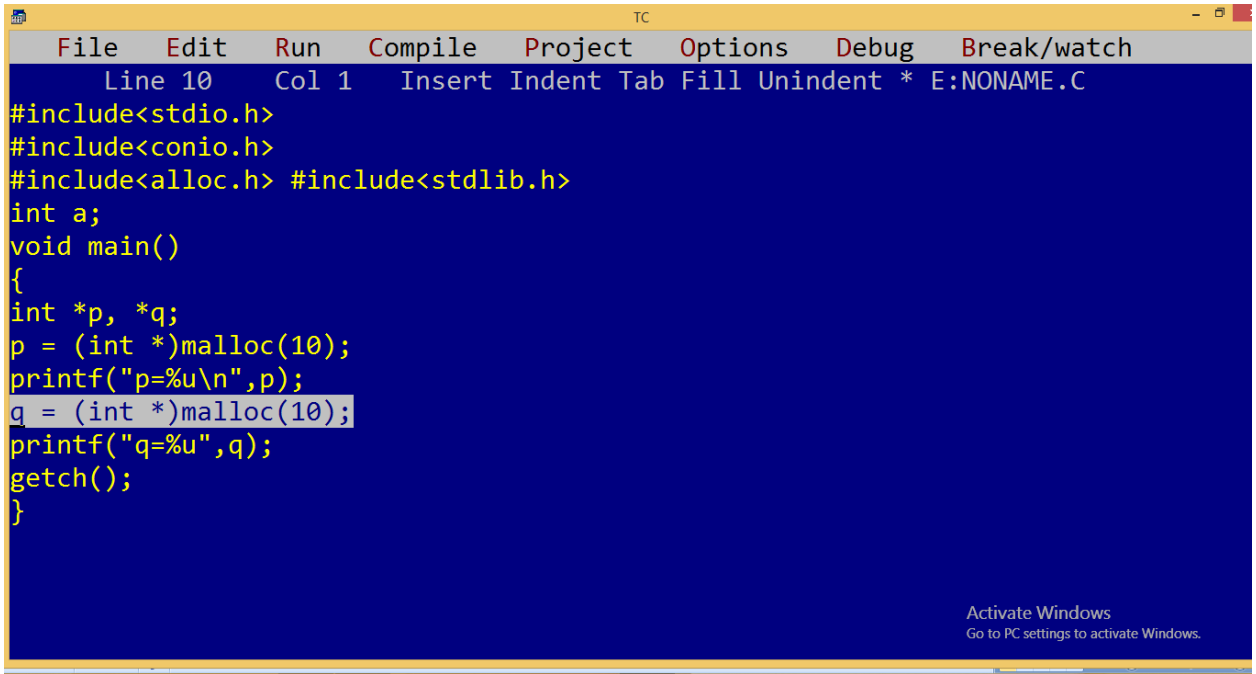
Syntax: void free(pointer);

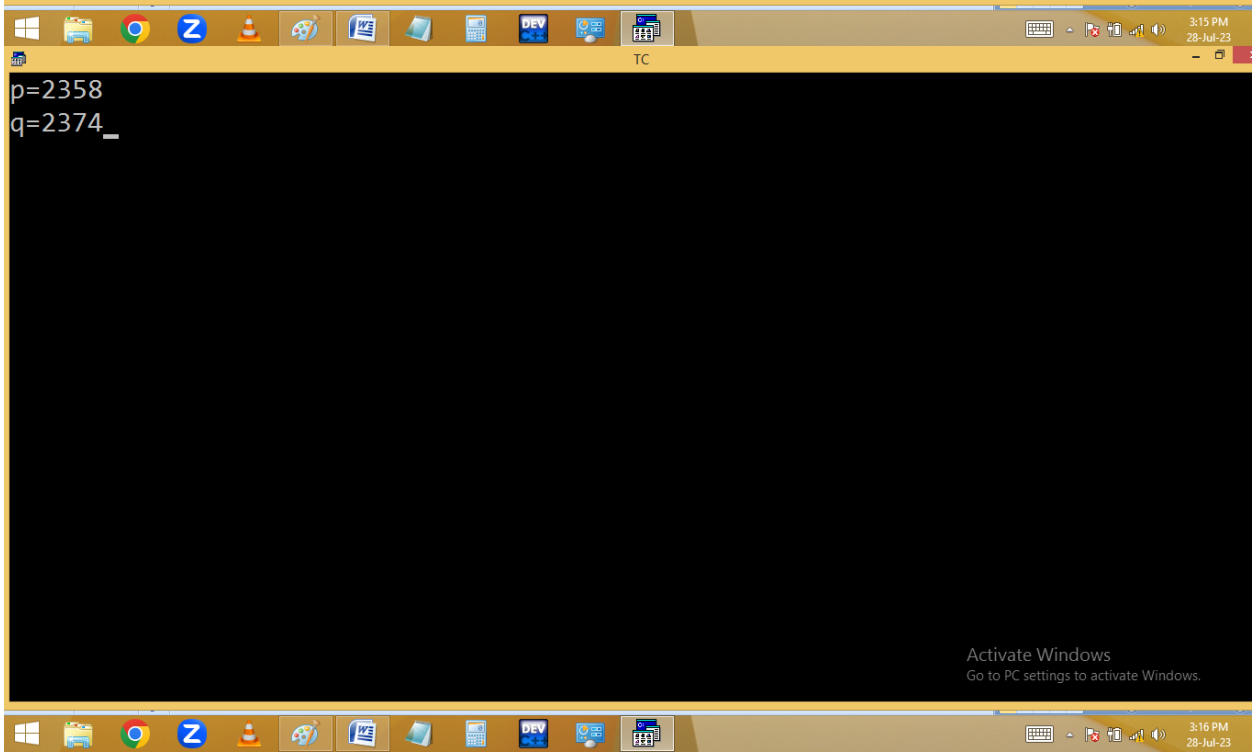
realloc(): It is used to extend the memory allocated by malloc() or calloc() at runtime. Working style is similar to malloc().

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();
}
Activate Windows
Go to PC settings to activate Windows.
3:15 PM
28-Jul-23
TC
p=2358
q=2358
Activate Windows
Go to PC settings to activate Windows.
3:15 PM
28-Jul-23


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();
}
Activate Windows
Go to PC settings to activate Windows.

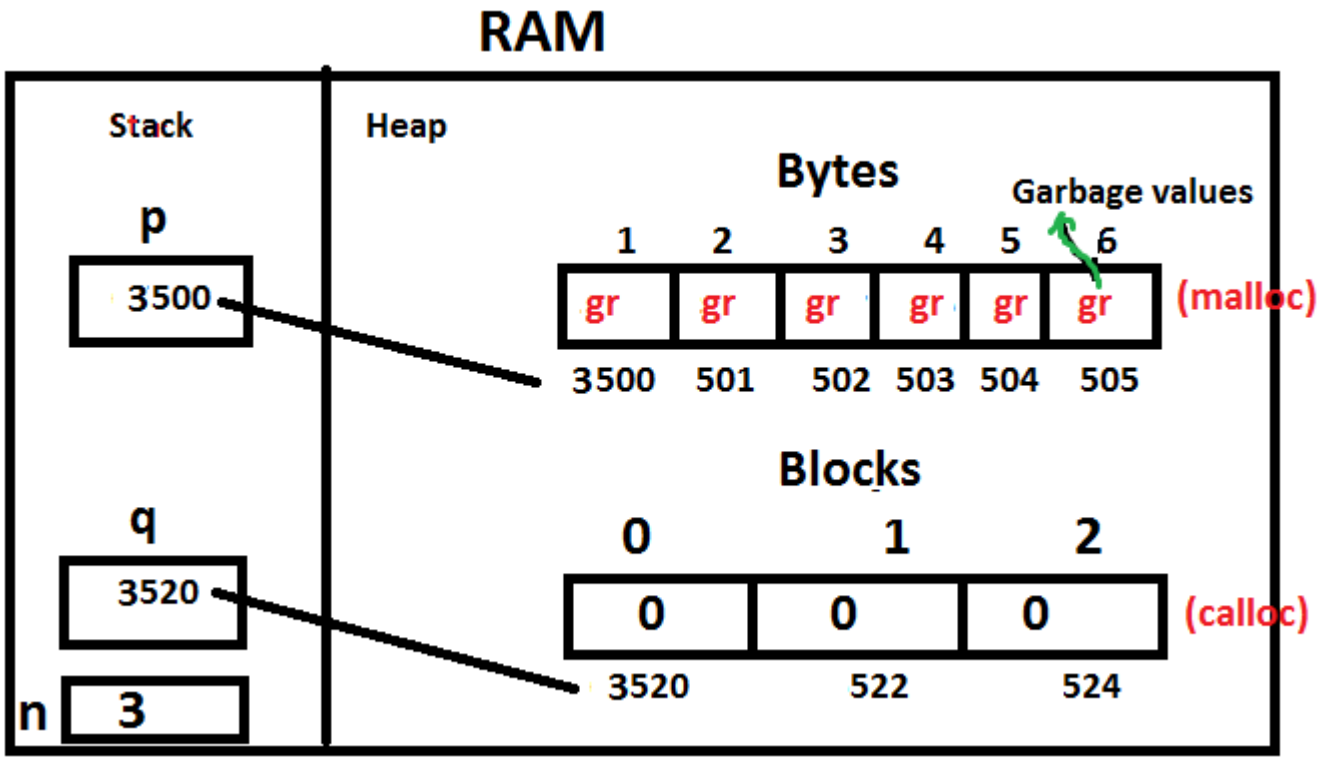

TC
p=2358
q=2374
Activate Windows
Go to PC settings to activate Windows.

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));
```



Eg:

Creating dynamic one-dimensional array:

realloc():

It is used to allocate the memory at middle or end stages of the program dynamically.

realloc() allocates the memory in the form of bytes.

It is used to modify the memory size allocated by malloc() and calloc().

The realloc() initial values are garbage.

It requires two arguments.

1. Pointername

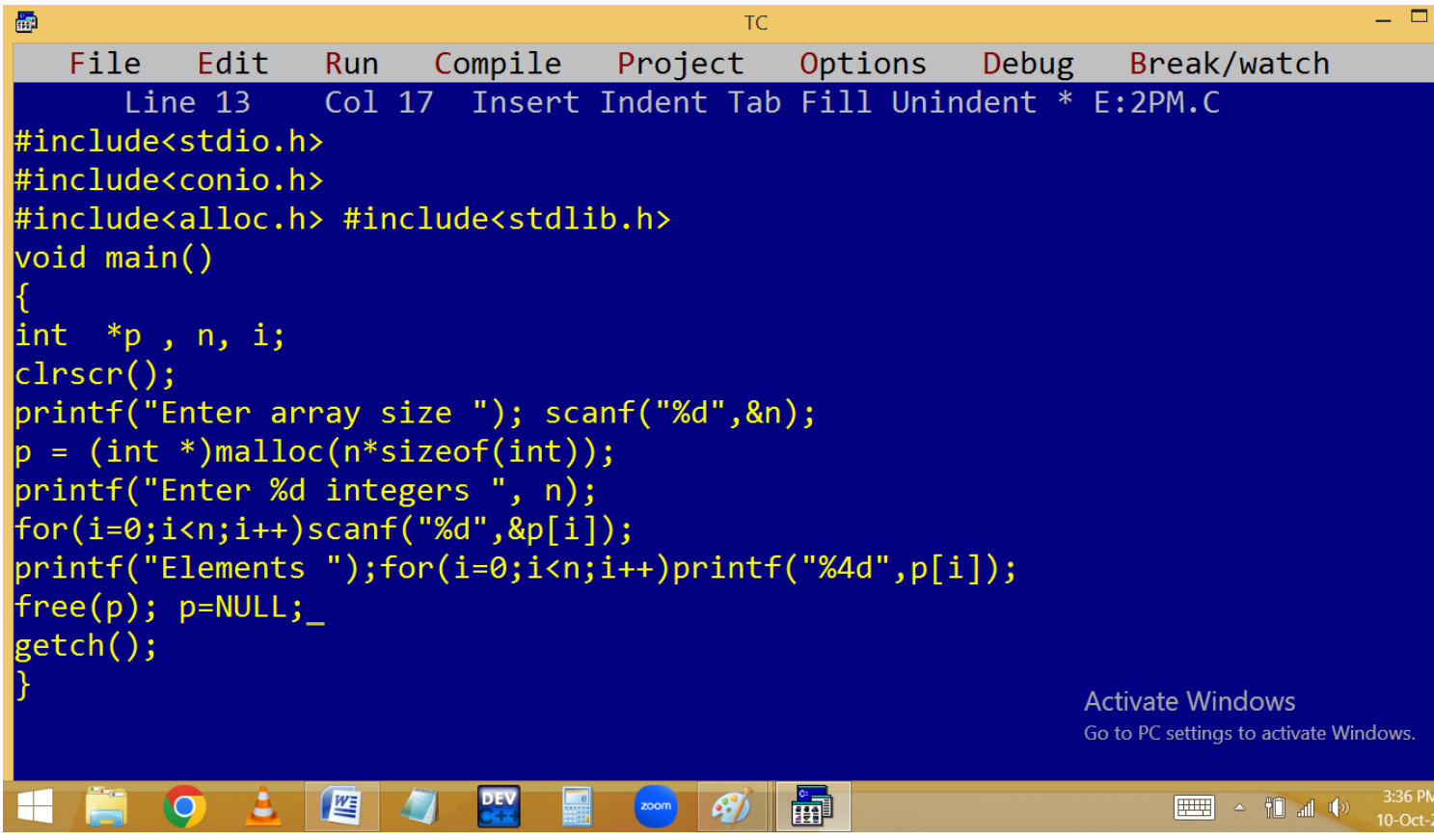
2. Newsize (bytes)

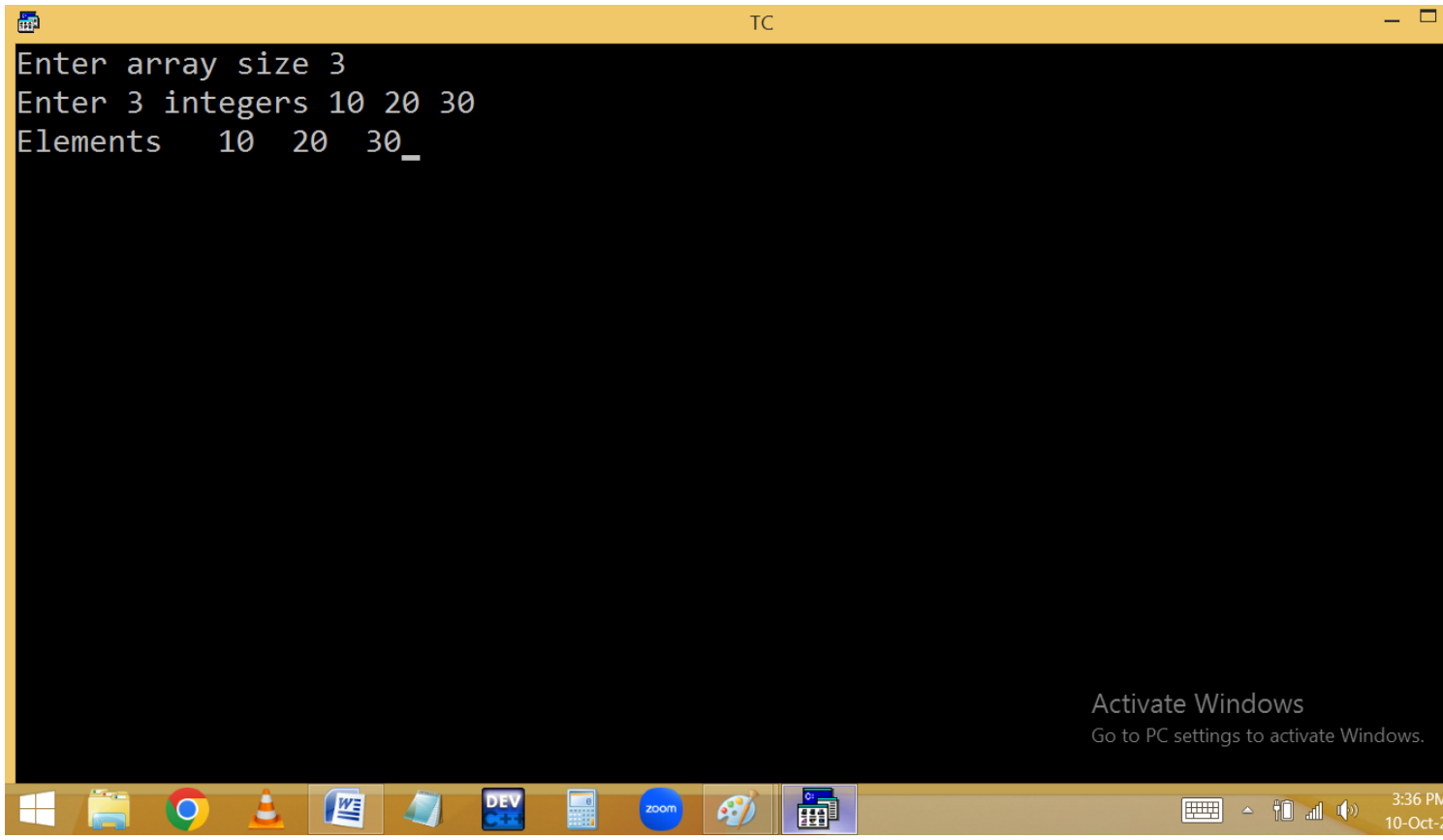
Syntax:

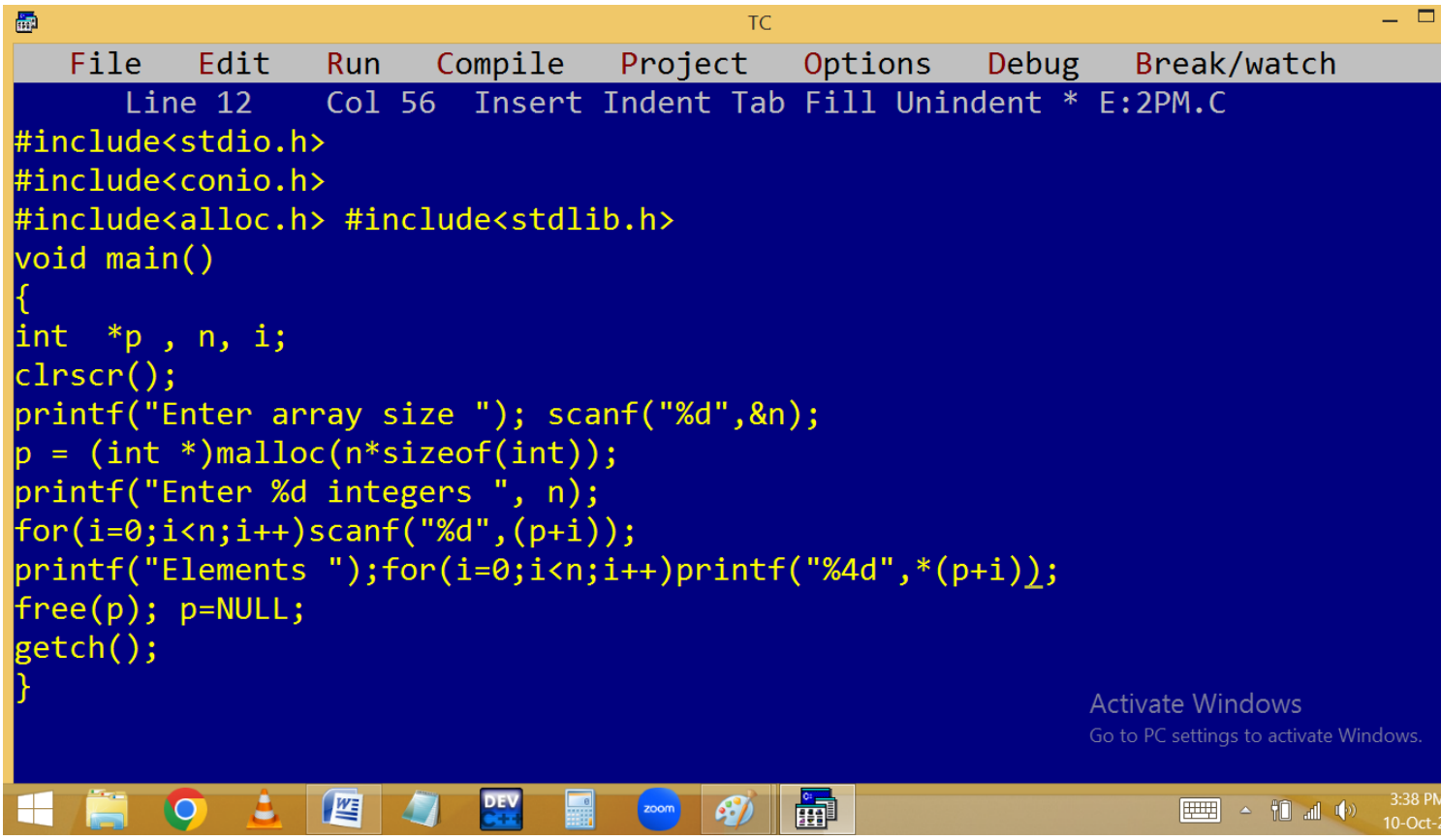
void * realloc(pointername, bytes);

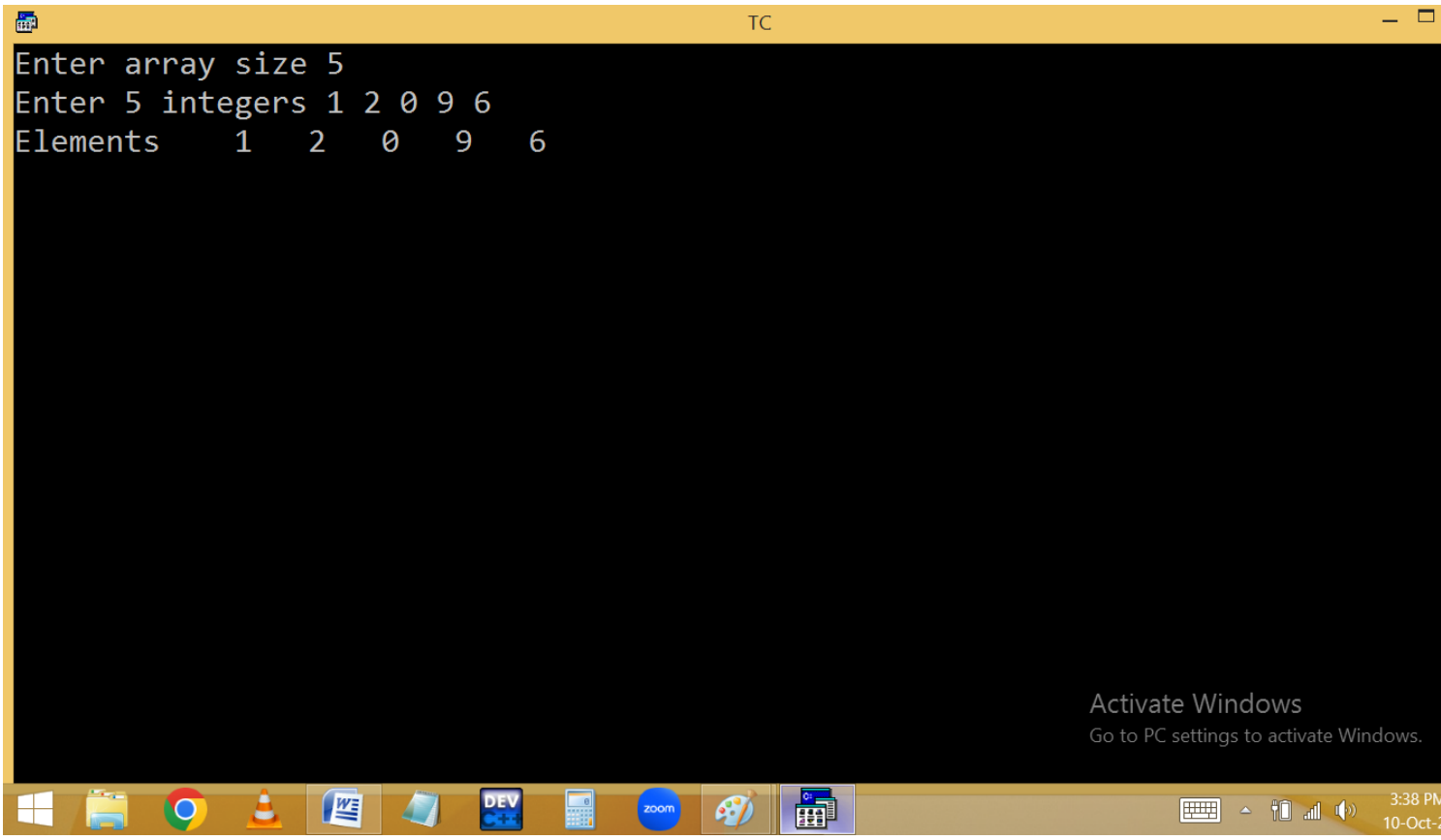
old address

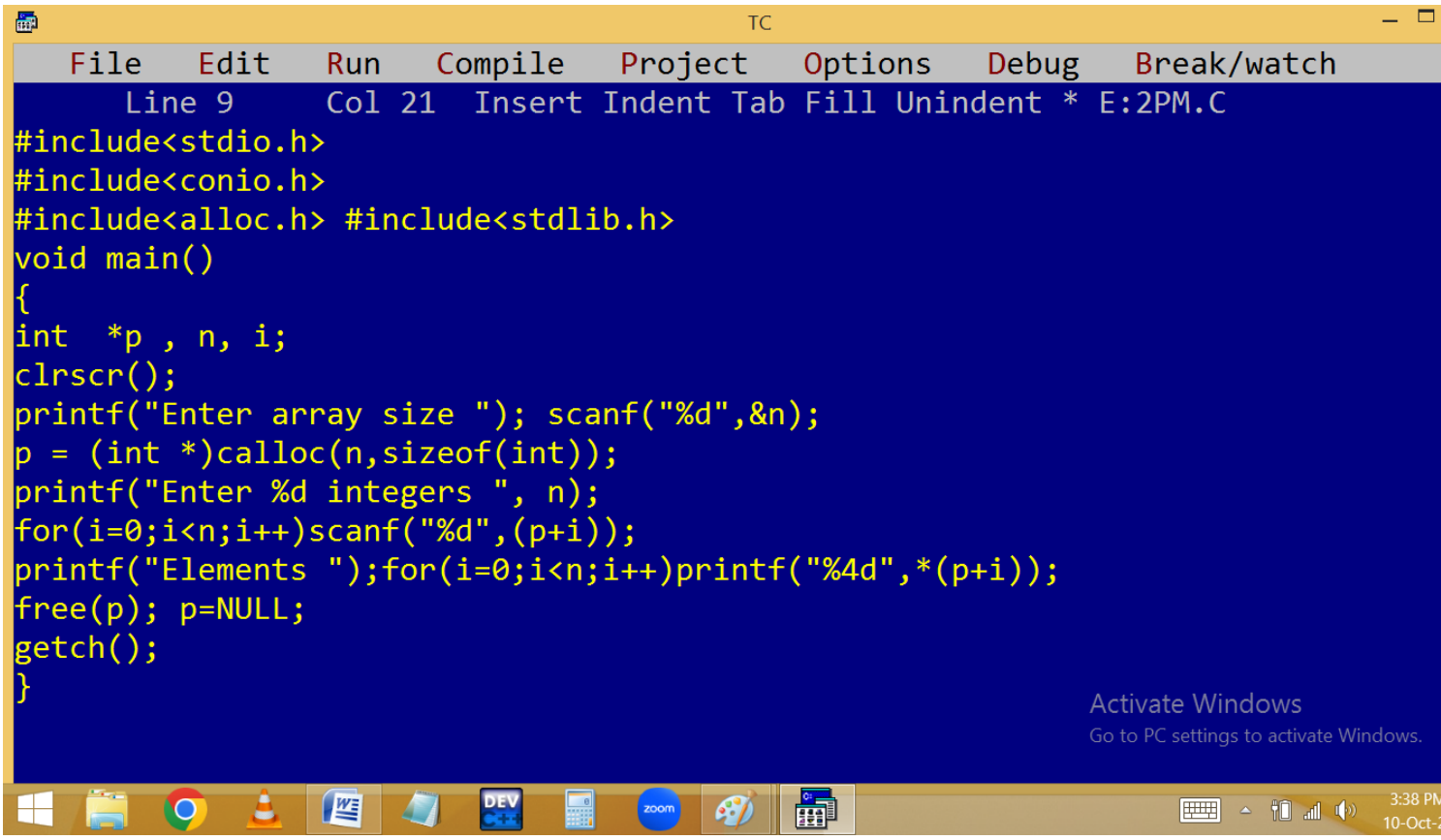
Creating a dynamic one dimensional array:

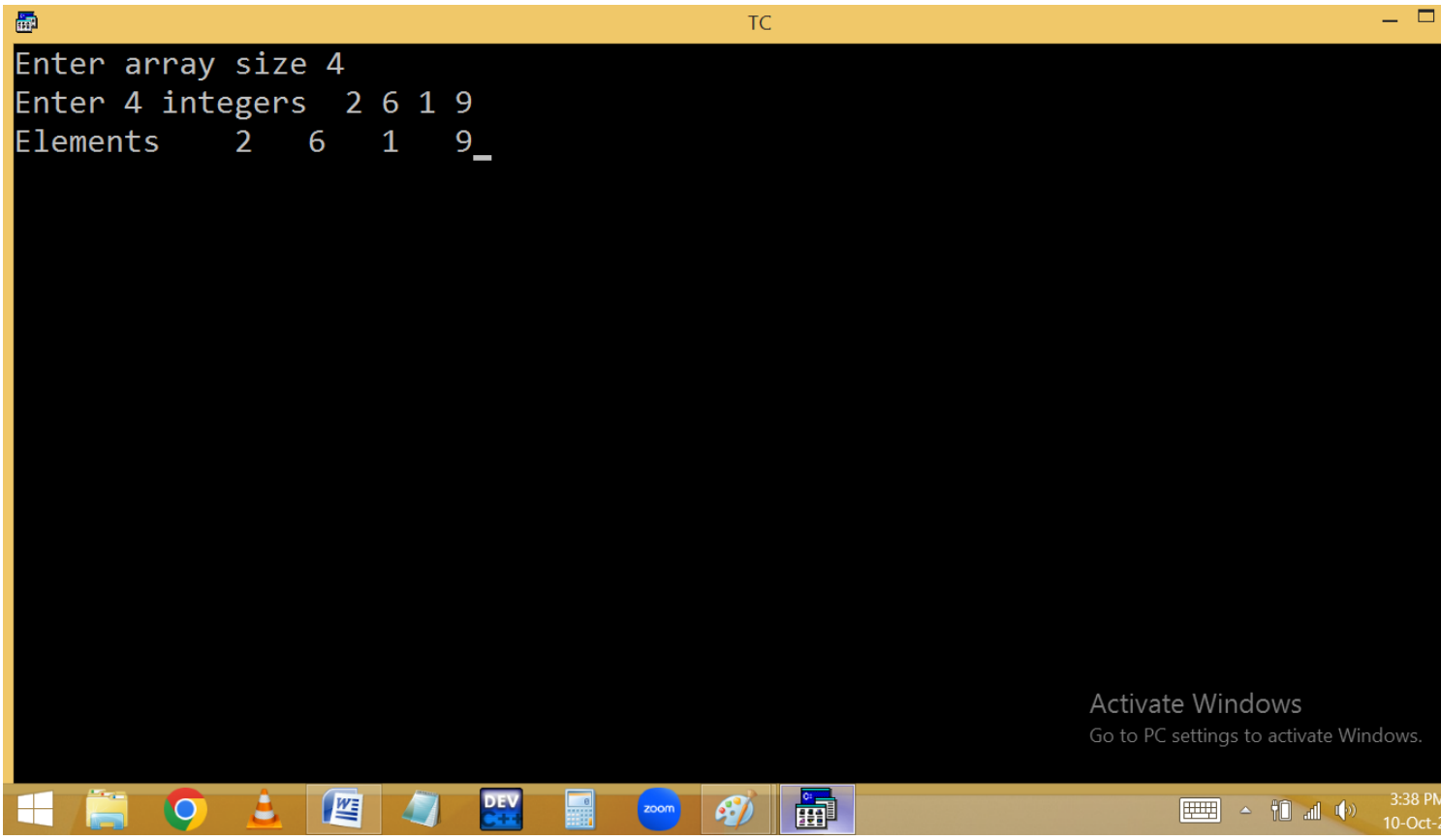



TC
Enter array size 3
Enter 3 integers 10 20 30
Elements 10 20 30
Activate Windows
Go to PC settings to activate Windows.
3:36 PM

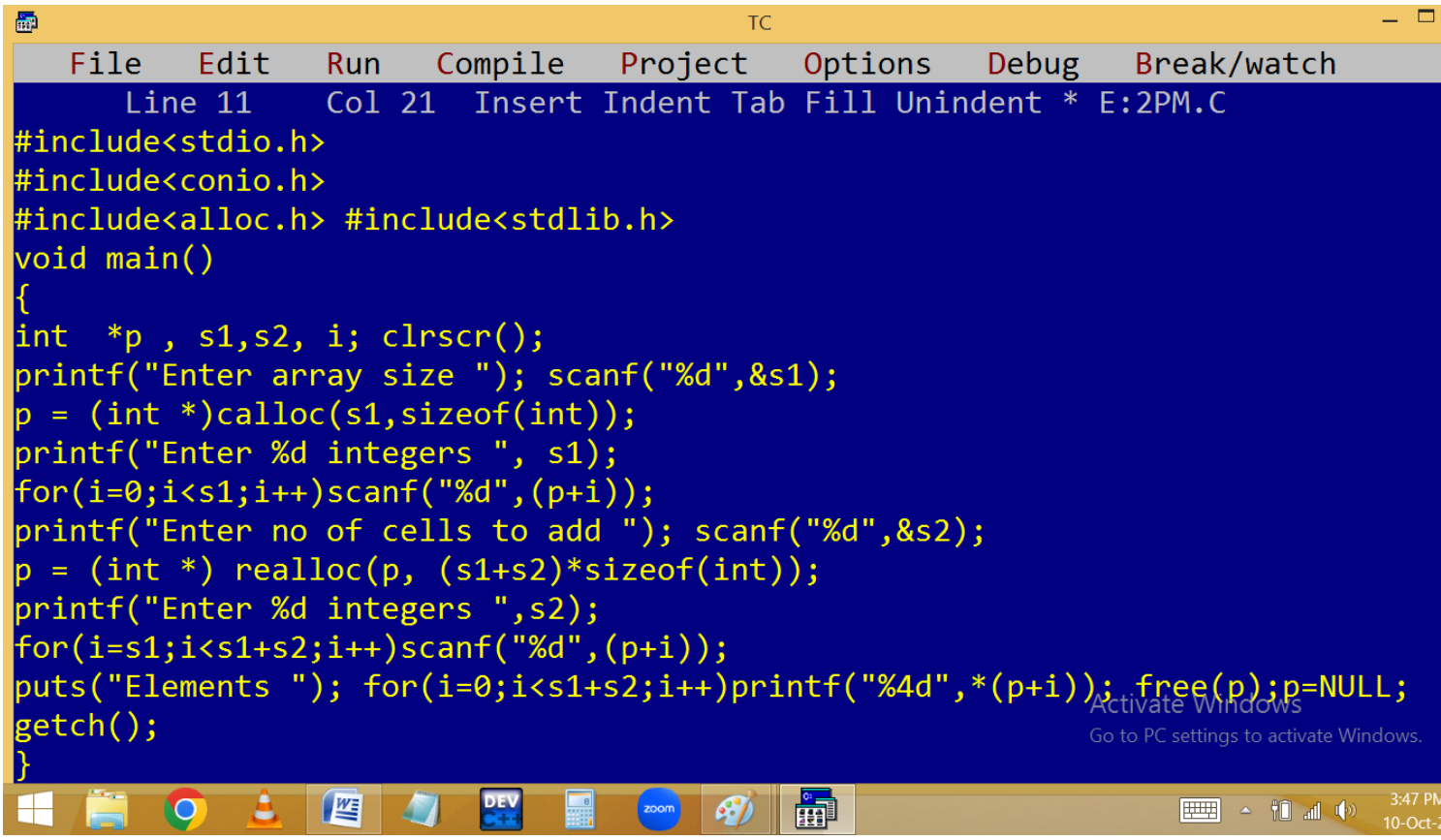

TC
File Edit Run Compile Project Options Debug Break/watch
Line 12 Col 56 Insert Indent Tab Fill Unindent * E:2PM.C
#include<stdio.h>
#include<conio.h>
#include<alloc.h> #include<stdlib.h>
void main()
{
int *p , n, i;
clrscr();
printf("Enter array size "); scanf("%d",&n);
p = (int *)malloc(n*sizeof(int));
printf("Enter %d integers ", n);
for(i=0;i<n;i++)scanf("%d",(p+i));
printf("Elements ");for(i=0;i<n;i++)printf("%4d",*(p+i));
free(p); p=NULL;
getch();
}
Activate Windows
Go to PC settings to activate Windows.

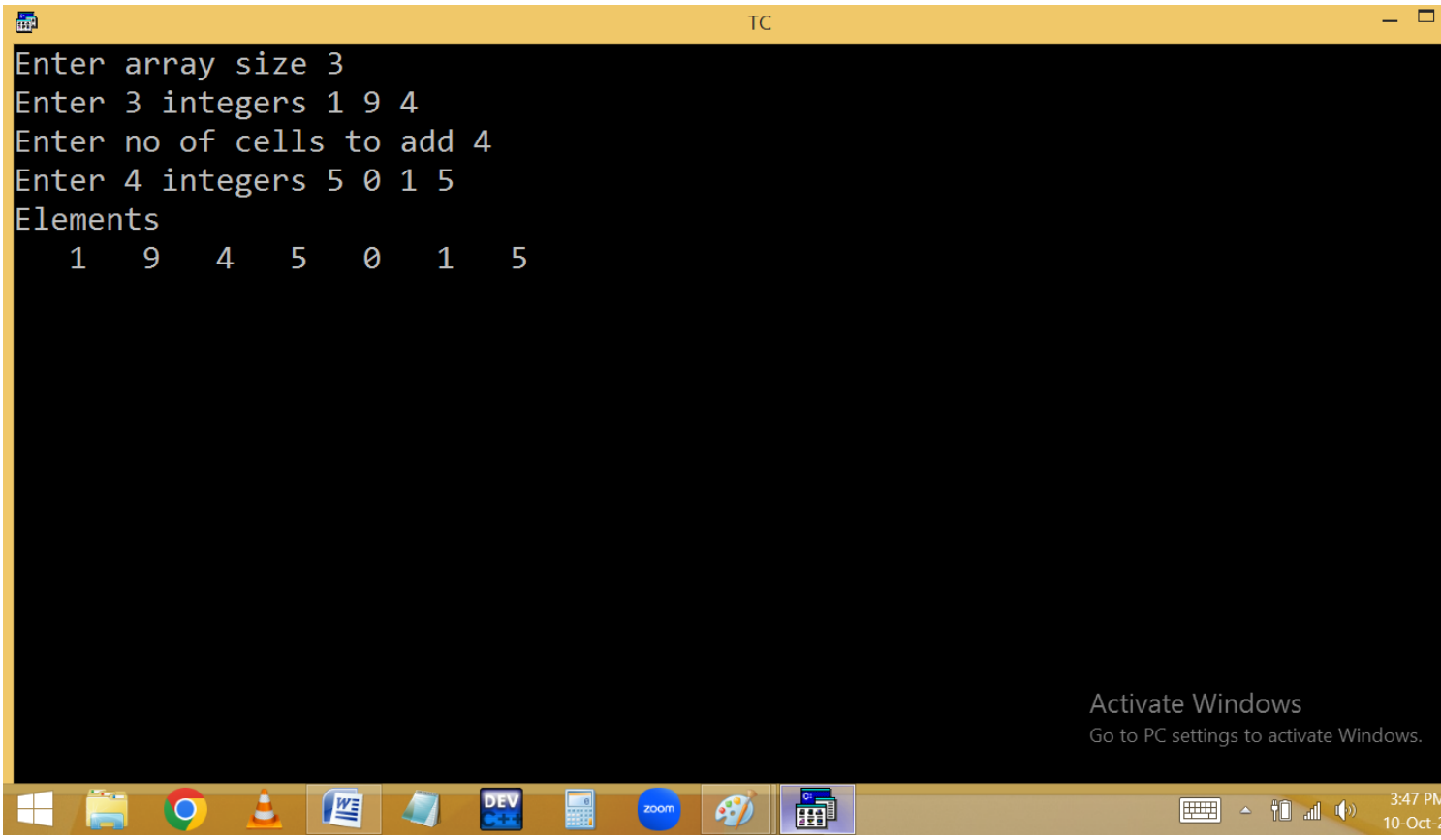

TC
Enter array size 5
Enter 5 integers 1 2 0 9 6
Elements 1 2 0 9 6
Activate Windows
Go to PC settings to activate Windows.
3:38 PM

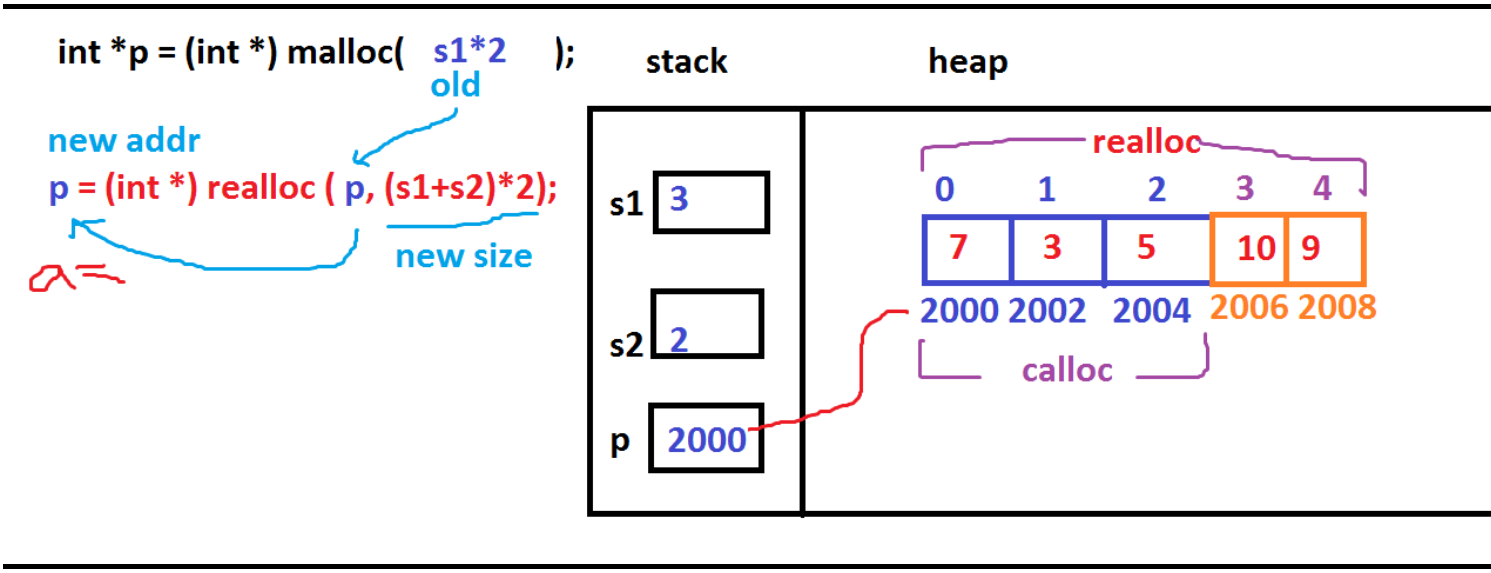

TC
File Edit Run Compile Project Options Debug Break/watch
Line 9 Col 21 Insert Indent Tab Fill Unindent * E:2PM.C
#include<stdio.h>
#include<conio.h>
#include<alloc.h> #include<stdlib.h>
void main()
{
int *p , n, i;
clrscr();
printf("Enter array size "); scanf("%d",&n);
p = (int *)calloc(n,sizeof(int));
printf("Enter %d integers ", n);
for(i=0;i<n;i++)scanf("%d",(p+i));
printf("Elements ");for(i=0;i<n;i++)printf("%4d",*(p+i));
free(p); p=NULL;
getch();
}
Activate Windows
Go to PC settings to activate Windows.
3:38 PM



Realloc() example:


TC
File Edit Run Compile Project Options Debug Break/watch
Line 11 Col 21 Insert Indent Tab Fill Unindent * E:2PM.C
#include<stdio.h>
#include<conio.h>
#include<alloc.h> #include<stdlib.h>
void main()
{
int *p , s1,s2, i; clrscr();
printf("Enter array size "); scanf("%d",&s1);
p = (int *)calloc(s1,sizeof(int));
printf("Enter %d integers ", s1);
for(i=0;i<s1;i++)scanf("%d",(p+i));
printf("Enter no of cells to add "); scanf("%d",&s2);
p = (int *) realloc(p, (s1+s2)*sizeof(int));
printf("Enter %d integers ",s2);
for(i=s1;i<s1+s2;i++)scanf("%d",(p+i));
puts("Elements "); for(i=0;i<s1+s2;i++)printf("%4d",*(p+i)); free(p);p=NULL;
getch();
}
Activate Windows
Go to PC settings to activate Windows.


TC
Enter array size 3
Enter 3 integers 1 9 4
Enter no of cells to add 4
Enter 4 integers 5 0 1 5
Elements
1 9 4 5 0 1 5
Activate Windows
Go to PC settings to activate Windows.


int *p = (int *) malloc(s1*2);
old
new addr
p = (int *) realloc (p, (s1+s2)*2);
new size
stack
heap
s1 3
s2 2
p 2000
realloc
0 1 2 3 4
7 3 5 10 9
2000 2002 2004 2006 2008
calloc