

```
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();
}
```

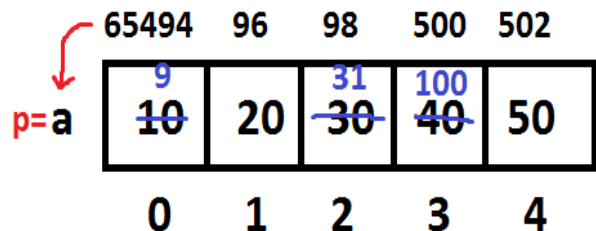
9:26 AM
11-Dec-24

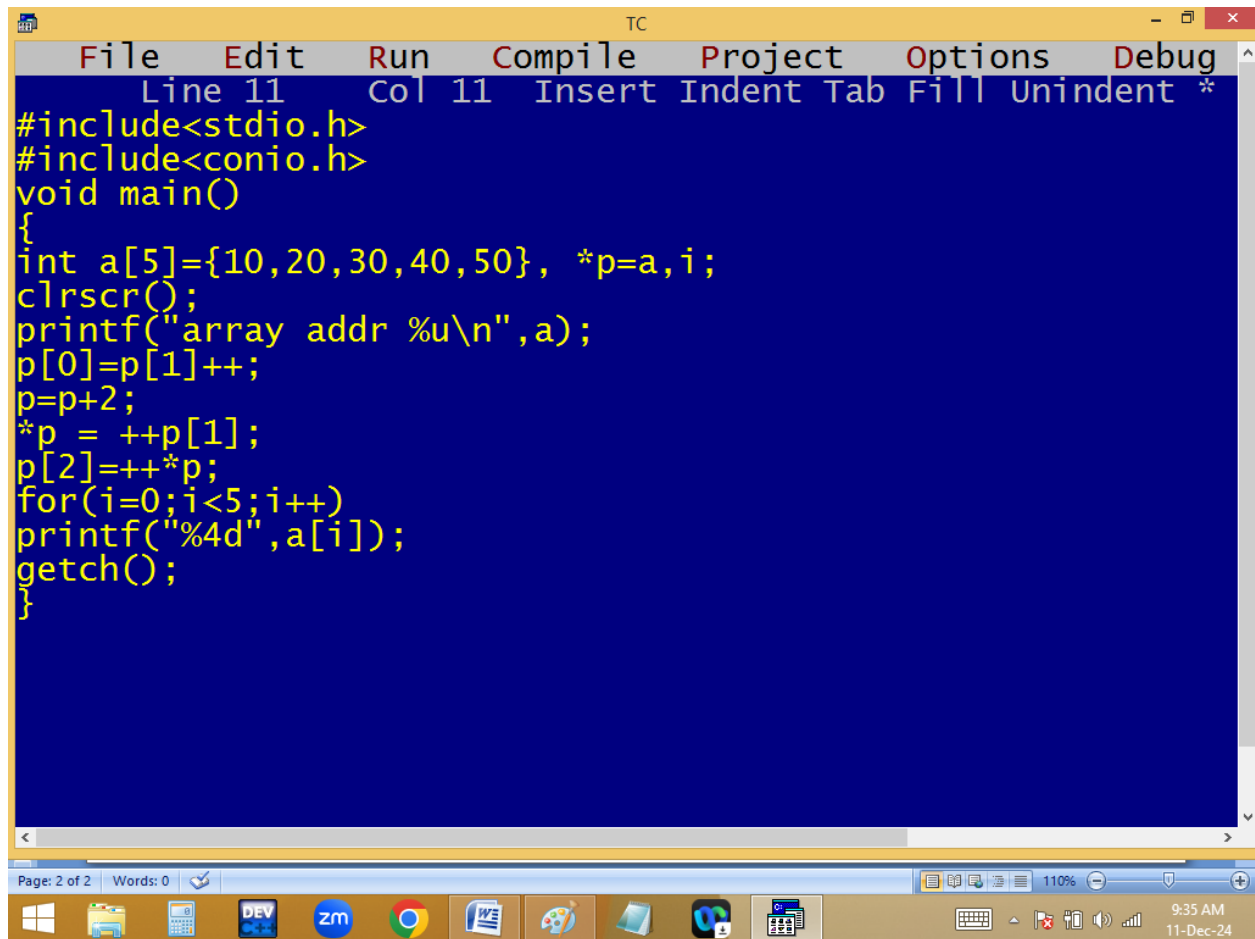
```

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

```

$p++ \Rightarrow 65494 + 1 \times 2 = 65496$
 $p(*p) \Rightarrow \text{value at } 65496 \Rightarrow 20 \checkmark$
 $*p++ \Rightarrow 65496 + 1 \times 2 = 65498$
 $p(*p) \Rightarrow \text{value at } 65498 \Rightarrow 30 \checkmark$
 $(*p)++ \Rightarrow \text{value at } 65498 \Rightarrow 30 \Rightarrow 31$
 $p(*p) \Rightarrow \text{value at } 65498 \Rightarrow 31 \checkmark$
 $p = p - 2; 65498 - 2 \times 2 = 65494$
 $--*p \Rightarrow \text{value at } 65494 \Rightarrow 10 \Rightarrow 9$
 $p(*p) \Rightarrow \text{value at } 65494 \Rightarrow 9 \checkmark$
 $p = p + 3; 65494 + 3 \times 2 = 65500$
 $*p = 100$
 $\text{value at } 65500 = 100 \Rightarrow 40 = 100$
 $p(*p) \Rightarrow \text{value at } 65500 \Rightarrow 100 \checkmark$





```
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();
}
```

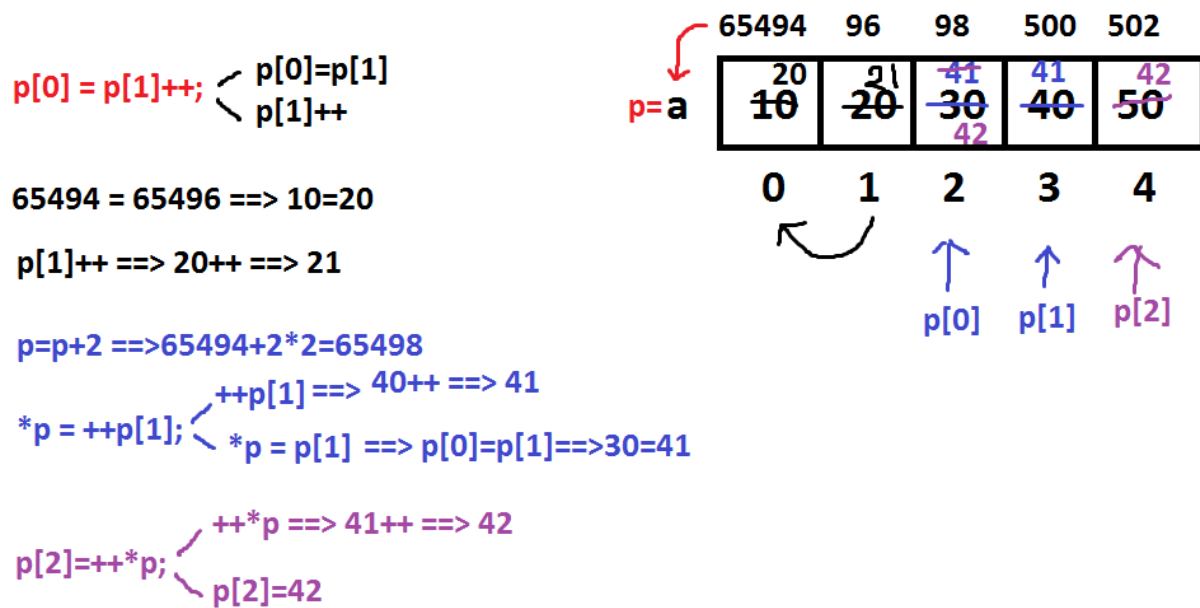
Page: 2 of 2 Words: 0 110%

9:35 AM
11-Dec-24

The screenshot shows a Windows desktop environment. A window titled "TC" (Turbo C) is open, displaying the following assembly code:

```
array addr 65494
    20  21  42  41  42
```

The taskbar at the bottom contains several icons: Windows Start button, File Explorer, Calculator, DEV (Developer Tools), Zoom (zm), Google Chrome, Microsoft Word, Paint, File Explorer, and a folder icon. The system tray on the right shows the date and time as 9:35 AM on 11-Dec-24, along with icons for keyboard, volume, and network status.



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);
```

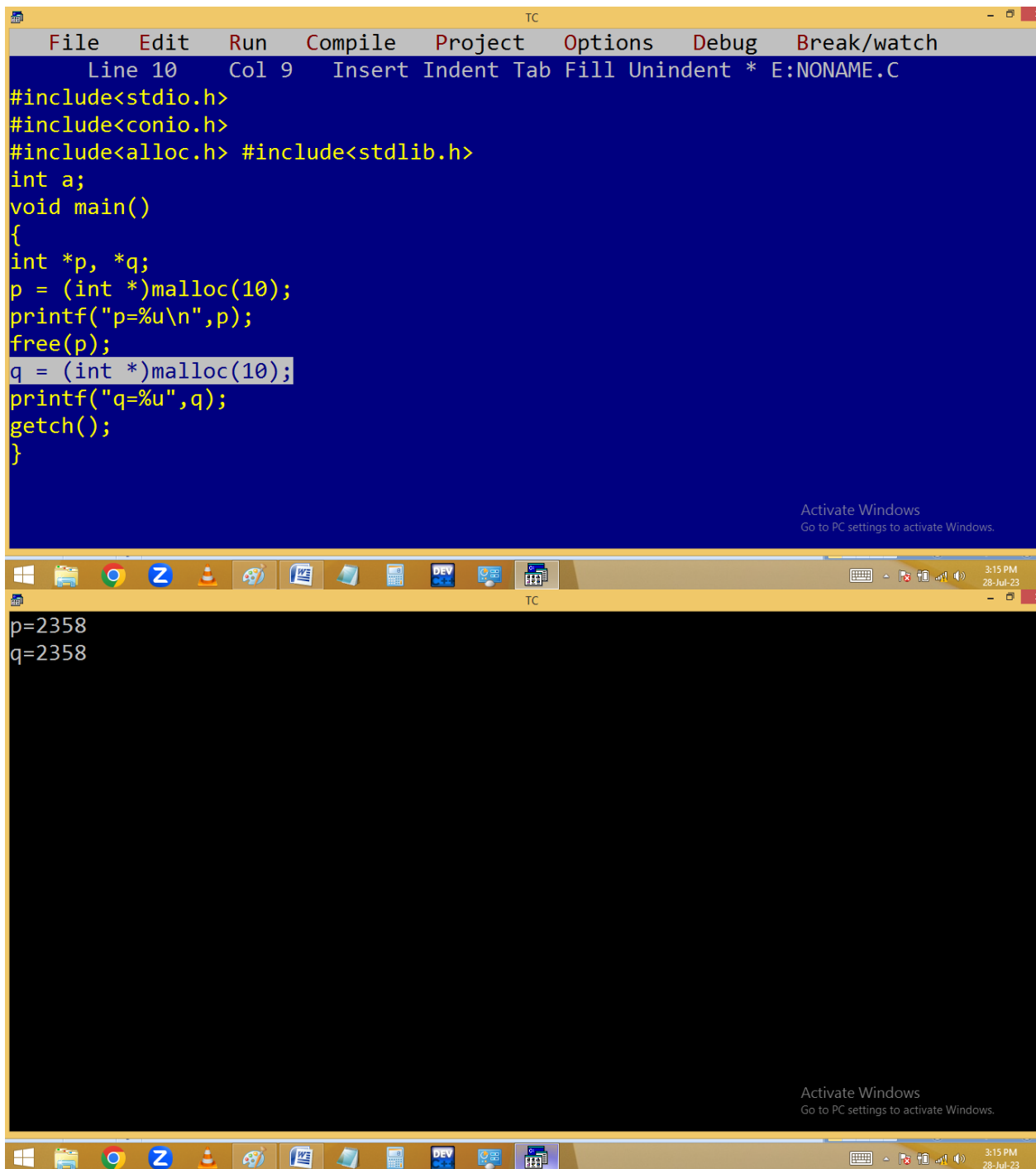

free(): It is used to release the memory allocated by malloc(), calloc() and realloc().

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:



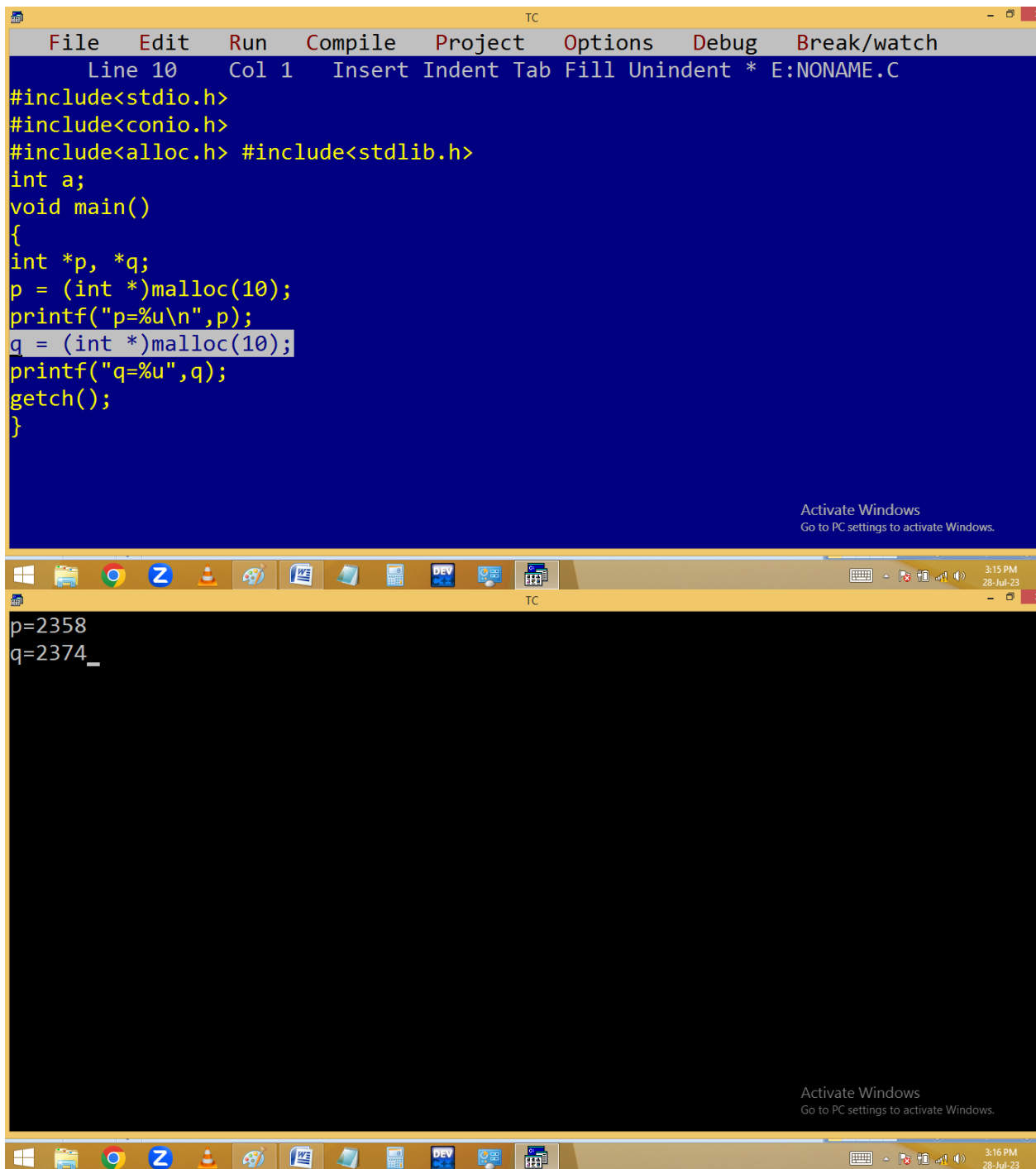
The image shows a Turbo C++ (TC) IDE window. The top menu bar includes File, Edit, Run, Compile, Project, Options, Debug, and Break/watch. The status bar at the top indicates 'Line 10 Col 9 Insert Indent Tab Fill Unindent * E:NONAME.C'. The code editor contains the following C program:

```
#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();
}
```

The output window at the bottom shows the execution results:

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

Both the code editor and the output window display an 'Activate Windows' watermark in the bottom right corner, stating 'Go to PC settings to activate Windows.' The Windows taskbar at the bottom shows the time as 3:15 PM on 28-Jul-23.



The image shows a screenshot of the Turbo C++ (TC) IDE. The top window displays a C program with the following code:

```
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();
}
```

The bottom window shows the output of the program:

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

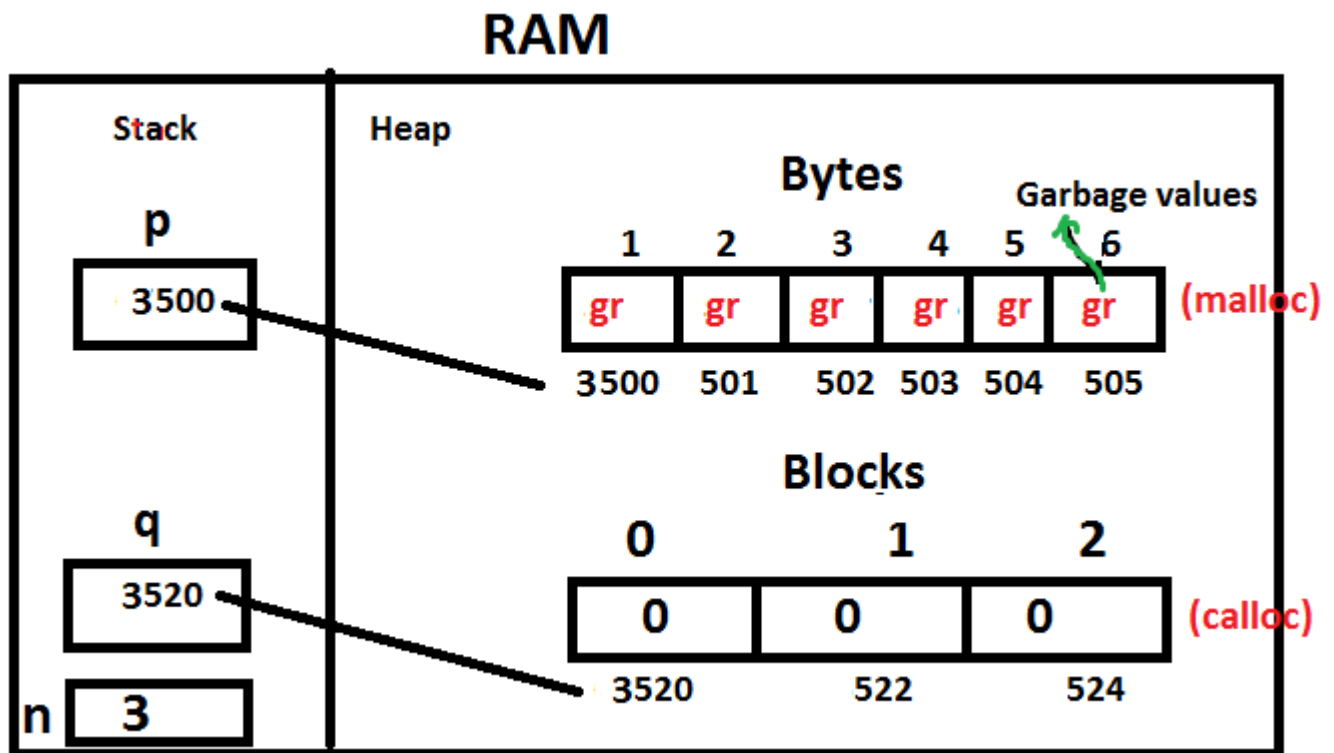
Both windows include a status bar at the bottom with the text "Activate Windows Go to PC settings to activate Windows." and a taskbar at the very bottom showing various application icons and the system clock.

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

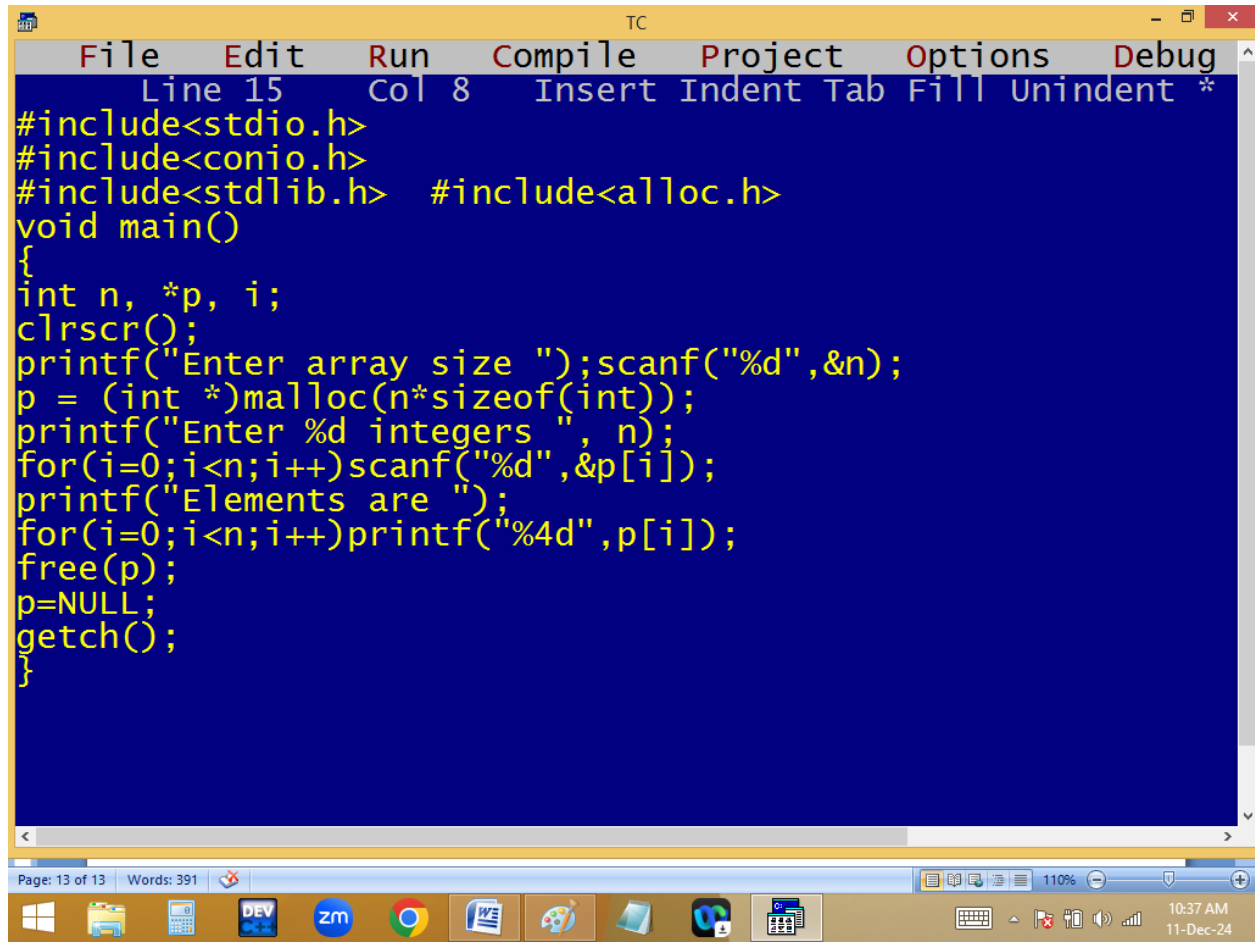
```
int *p, *q, n=3;
```

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

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



Creating dynamic one-dimensional array:



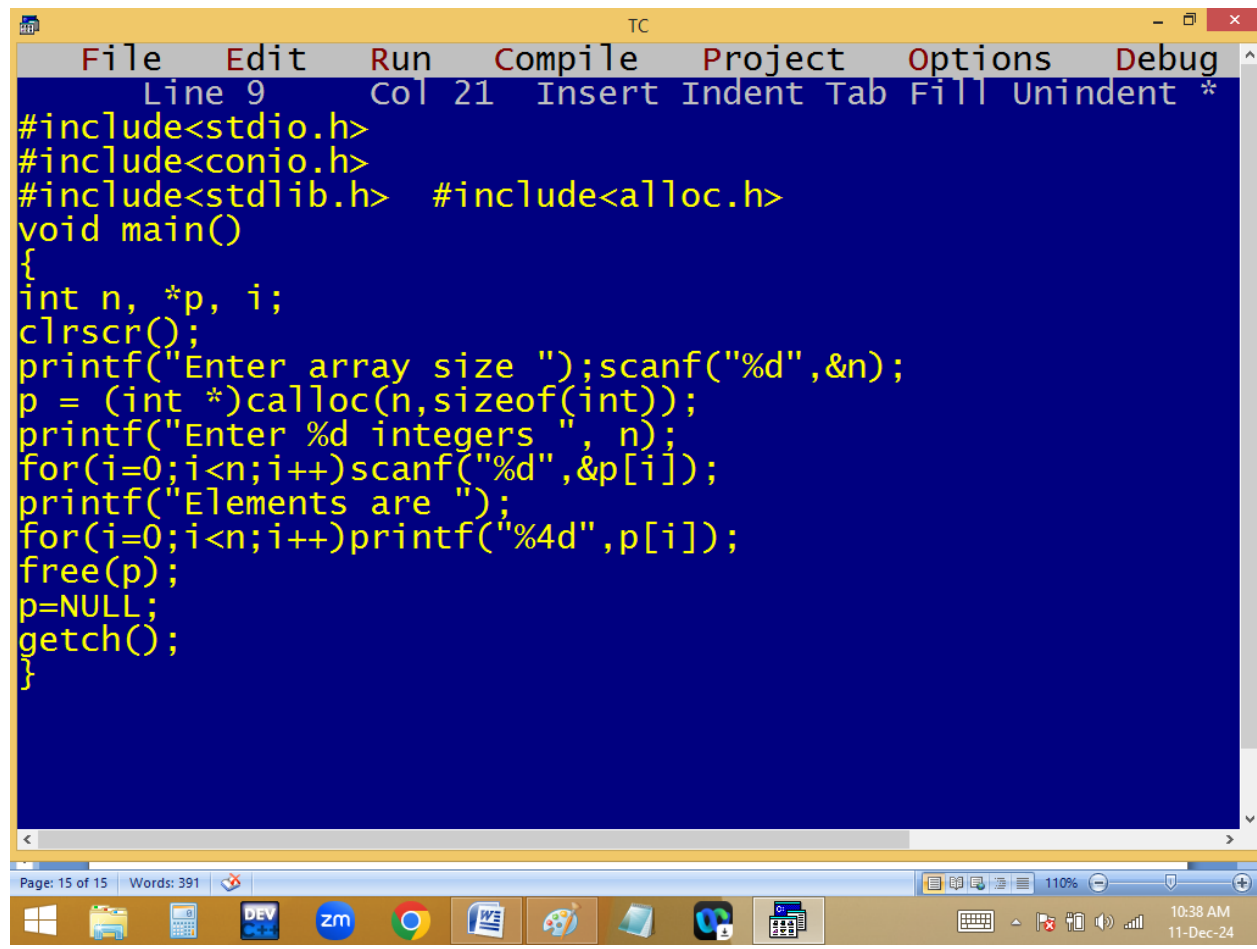
The screenshot shows the Turbo C++ (TC) IDE with a blue background. The menu bar includes File, Edit, Run, Compile, Project, Options, and Debug. The status bar at the top indicates 'Line 15 Col 8' and lists actions: Insert, Indent, Tab, Fill, Unindent, and *. The code is written in yellow text on the blue background. It includes headers for stdio.h, conio.h, stdlib.h, and alloc.h. The main function declares variables n, p, and i, clears the screen, prompts for an array size, allocates memory using malloc, prompts for integers, reads them into the array, prints the elements, frees the memory, sets p to NULL, and waits for a key press before exiting.

```
TC
File Edit Run Compile Project Options Debug
Line 15 Col 8 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 *)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();
}
```

Page: 13 of 13 Words: 391 110% 10:37 AM 11-Dec-24

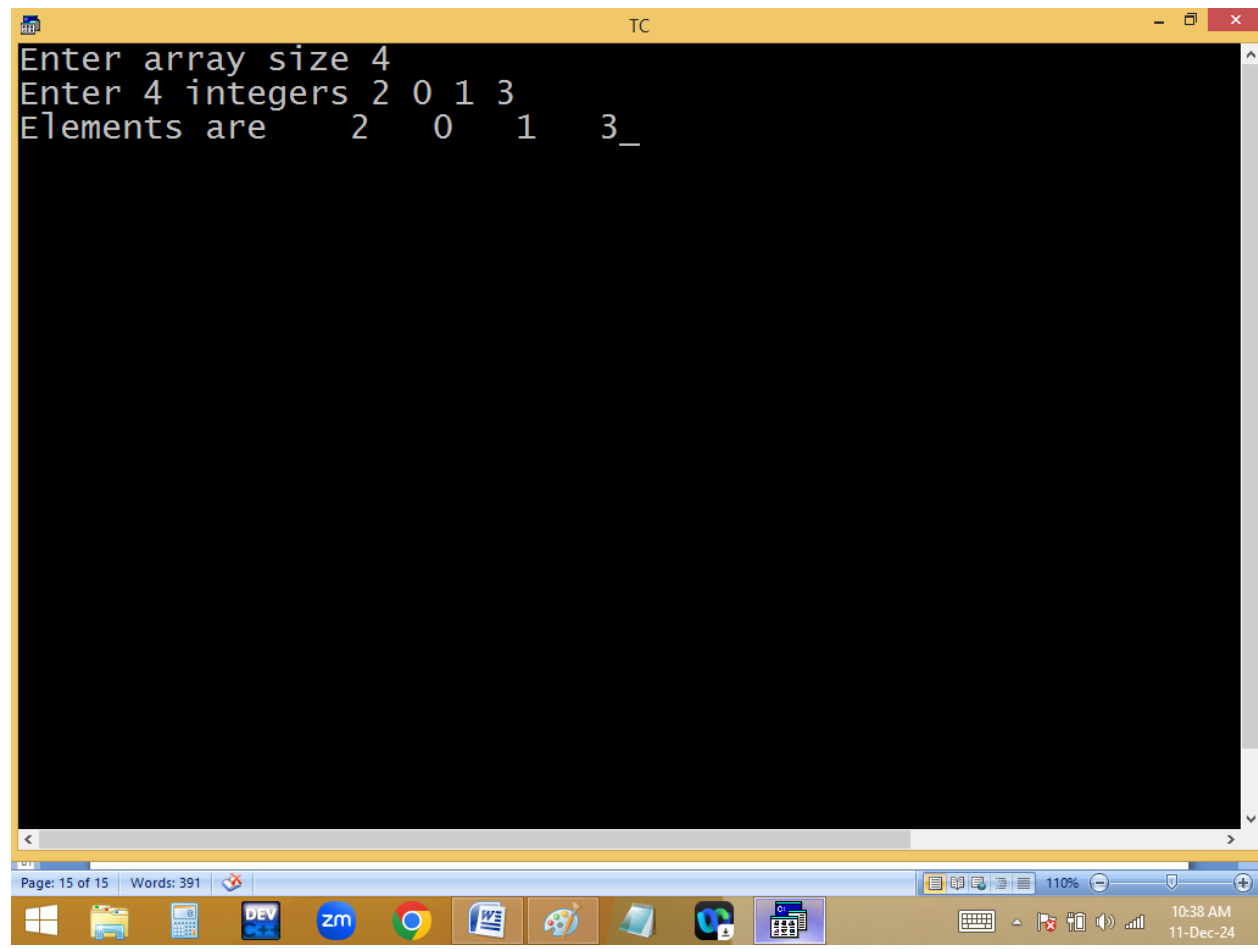
```
TC
Enter array size 5
Enter 5 integers 2 0 5 -9 6
Elements are 2 0 5 -9 6
```

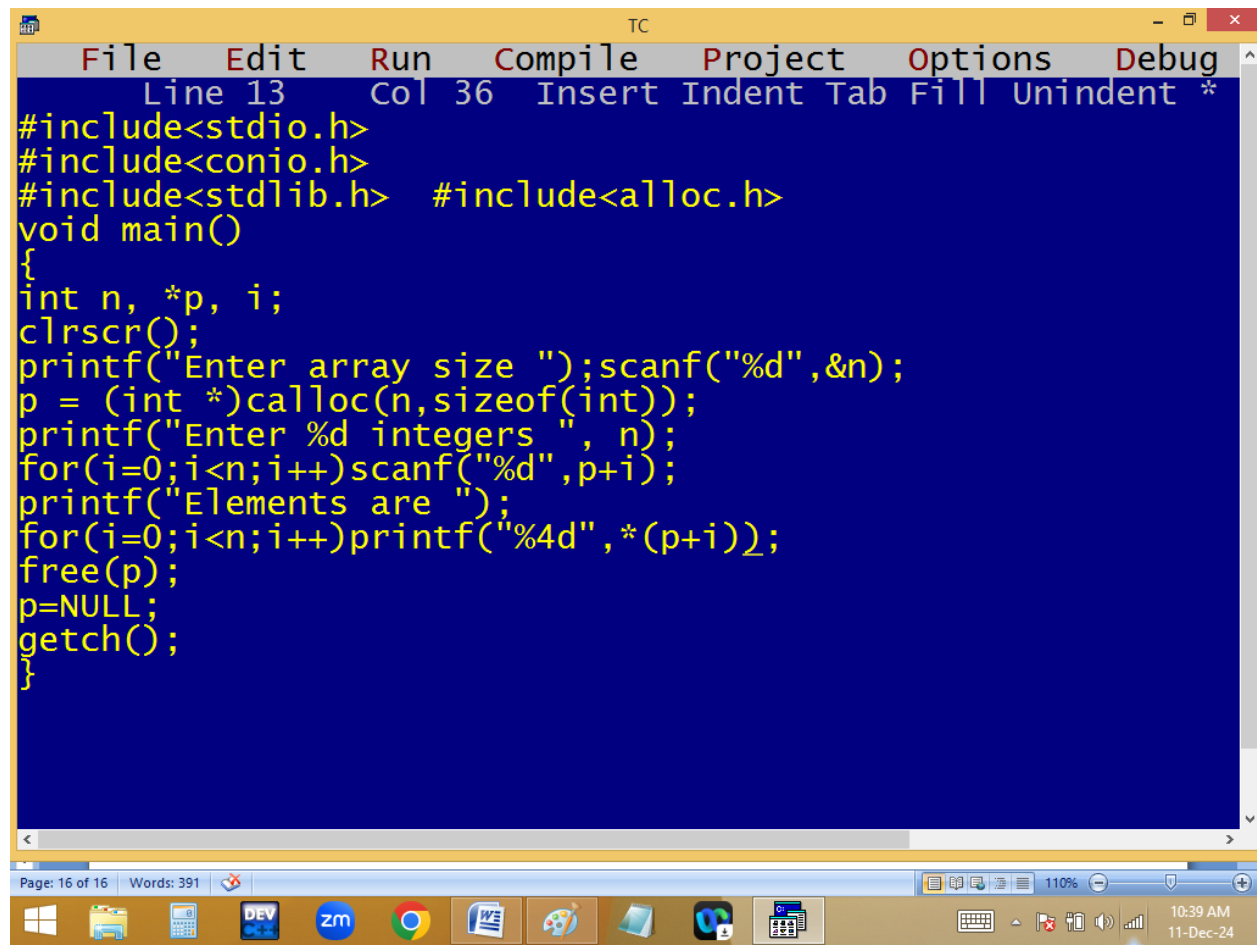
Page: 13 of 13 Words: 391 110% 10:37 AM 11-Dec-24



The image shows a screenshot of the Turbo C++ (TC) IDE. The window title is "TC". The menu bar includes "File", "Edit", "Run", "Compile", "Project", "Options", and "Debug". The status bar at the top indicates "Line 9", "Col 21", and lists editing actions: "Insert", "Indent", "Tab", "Fill", "Unindent", and "*". The main editing area has a dark blue background with yellow text. It contains a C program that uses `malloc` and `free` to dynamically allocate memory for an array. The program prompts the user to enter an array size and then integers, and then prints the elements. The bottom status bar shows "Page: 15 of 15" and "Words: 391". The Windows taskbar at the bottom includes icons for various applications and the system clock showing "10:38 AM" on "11-Dec-24".

```
#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();
}
```





The image shows a screenshot of the Turbo C++ (TC) IDE. The window title is "TC". The menu bar includes "File", "Edit", "Run", "Compile", "Project", "Options", and "Debug". The status bar at the top indicates "Line 13", "Col 36", and "Insert Indent Tab Fill Unindent *". The code is written in yellow text on a dark blue background. It includes headers for `stdio.h`, `conio.h`, `stdlib.h`, and `alloc.h`. The `main` function declares variables `n`, `*p`, and `i`, clears the screen with `clrscr()`, prompts the user for an array size, allocates memory with `calloc`, prompts for integers, reads them with `scanf`, prints them with `printf` in a formatted way, frees the memory, sets `p` to `NULL`, and ends with `getch()`. The bottom status bar shows "Page: 16 of 16" and "Words: 391". The Windows taskbar at the bottom includes icons for Windows, File Explorer, Calculator, DEV C++, Zoom, Google Chrome, Word, Paint, and other applications. The system clock shows "10:39 AM" and "11-Dec-24".

```
#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();
}
```

```
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