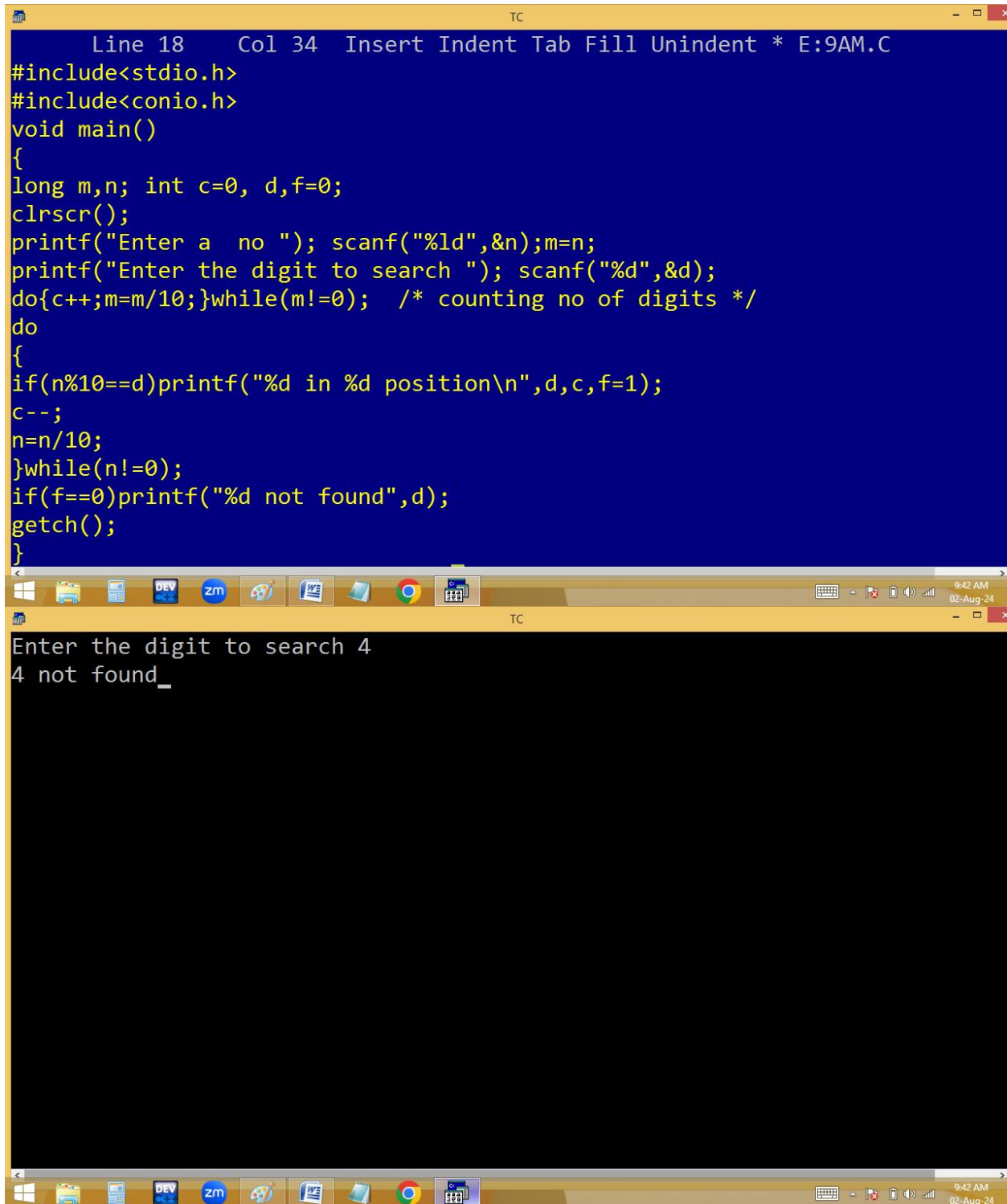


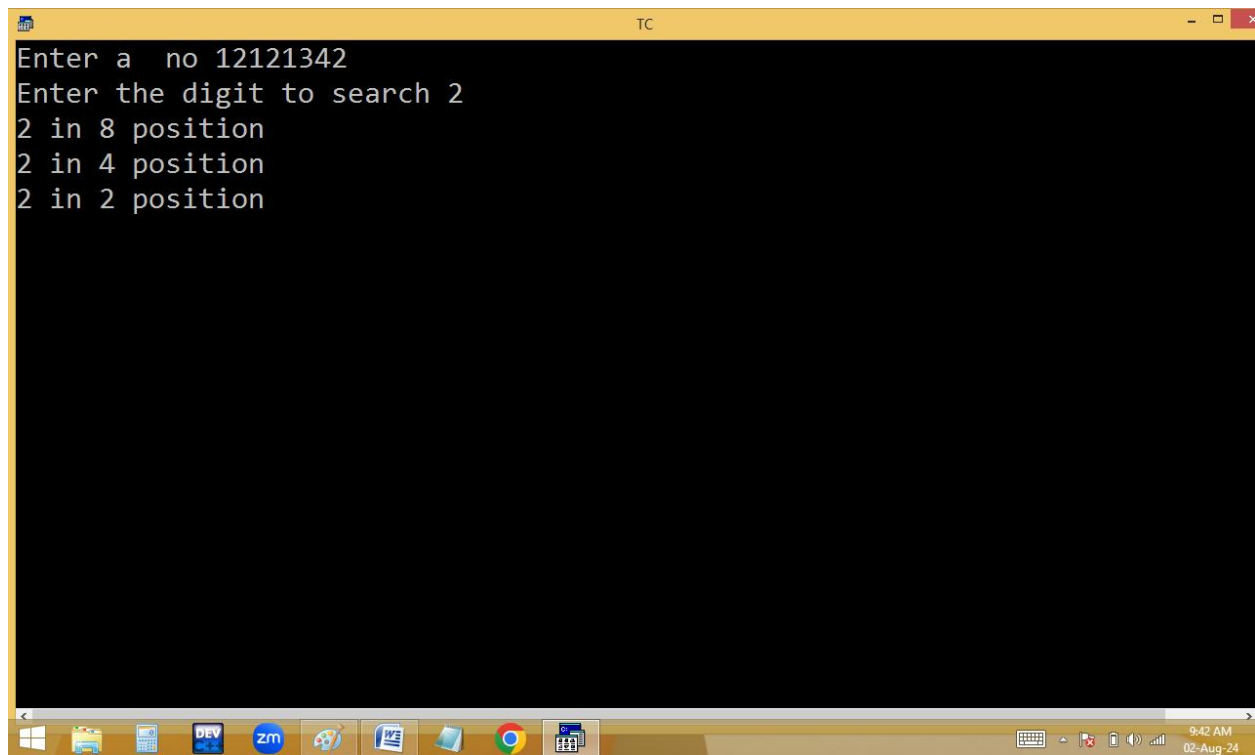
## Finding the no position:

1084 → 8 position is 3



```
Line 18   Col 34   Insert Indent Tab Fill Unindent * E:9AM.C
#include<stdio.h>
#include<conio.h>
void main()
{
long m,n; int c=0, d,f=0;
clrscr();
printf("Enter a no "); scanf("%ld",&n);m=n;
printf("Enter the digit to search "); scanf("%d",&d);
do{c++;m=m/10;}while(m!=0); /* counting no of digits */
do
{
if(n%10==d)printf("%d in %d position\n",d,c,f=1);
c--;
n=n/10;
}while(n!=0);
if(f==0)printf("%d not found",d);
getch();
}
```

Enter the digit to search 4  
4 not found\_

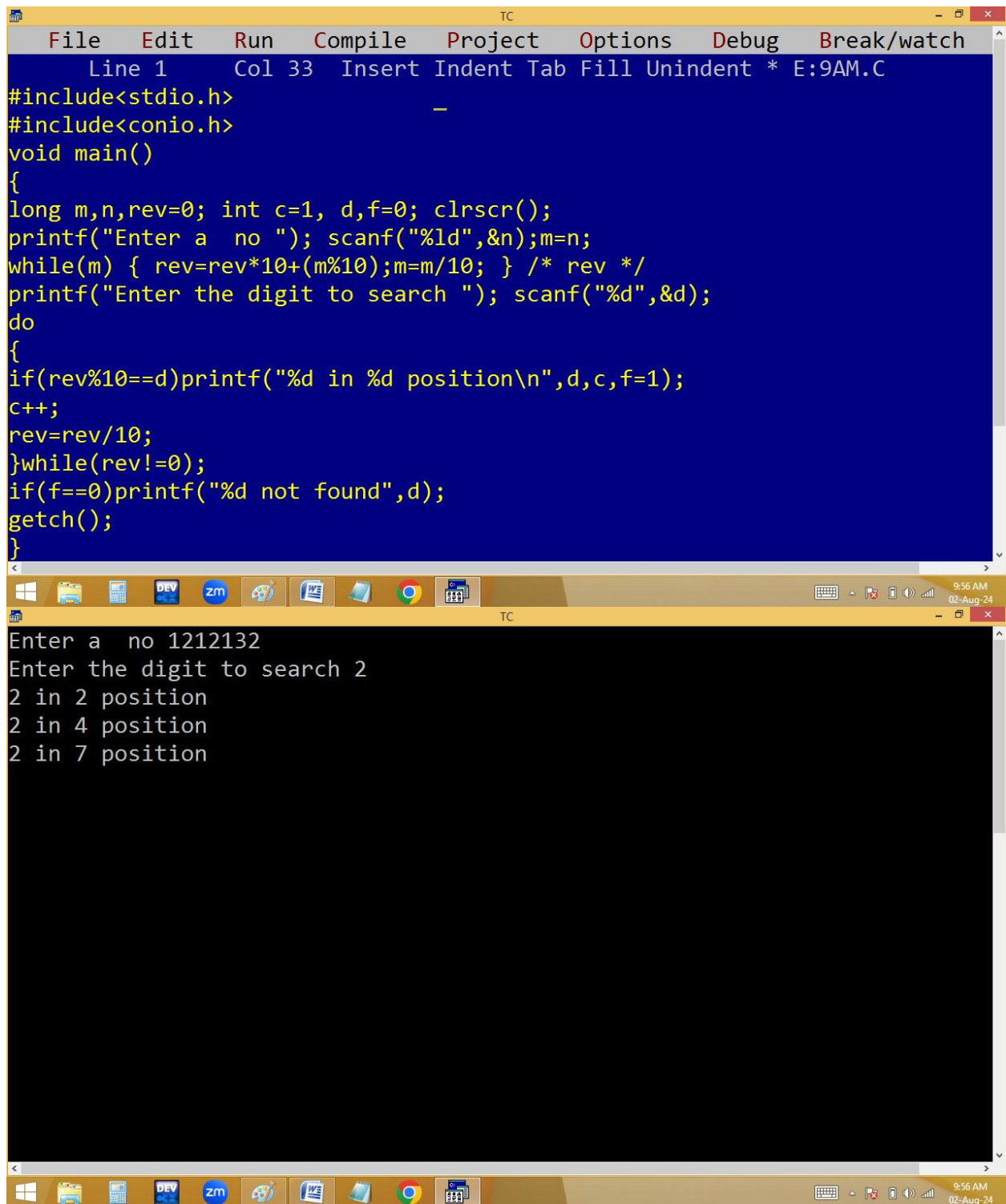


The screenshot shows a Windows 10 desktop with a yellow-themed taskbar. A Turbo C++ (TC) window is open, displaying a program that searches for a digit in a number. The program's output is as follows:

```
Enter a no 12121342
Enter the digit to search 2
2 in 8 position
2 in 4 position
2 in 2 position
```

The taskbar includes icons for Windows, File Explorer, DEV, zm, a game controller, Word, a folder, Chrome, and a calendar. The system tray on the right shows the time as 9:42 AM on 02-Aug-24.

Method2:



The image shows a screenshot of the Turbo C++ (TC) IDE. The top window displays the source code of a C program designed to find the position of a digit within a given number. The code uses a while loop to reverse the number and a do-while loop to check for the digit's position. The bottom window shows the program's execution output.

```
File Edit Run Compile Project Options Debug Break/watch
Line 1 Col 33 Insert Indent Tab Fill Unindent * E:9AM.C
#include<stdio.h>
#include<conio.h>
void main()
{
long m,n,rev=0; int c=1, d,f=0; clrscr();
printf("Enter a no "); scanf("%ld",&n);m=n;
while(m) { rev=rev*10+(m%10);m=m/10; } /* rev */
printf("Enter the digit to search "); scanf("%d",&d);
do
{
if(rev%10==d)printf("%d in %d position\n",d,c,f=1);
c++;
rev=rev/10;
}while(rev!=0);
if(f==0)printf("%d not found",d);
getch();
}
```

Enter a no 1212132  
Enter the digit to search 2  
2 in 2 position  
2 in 4 position  
2 in 7 position

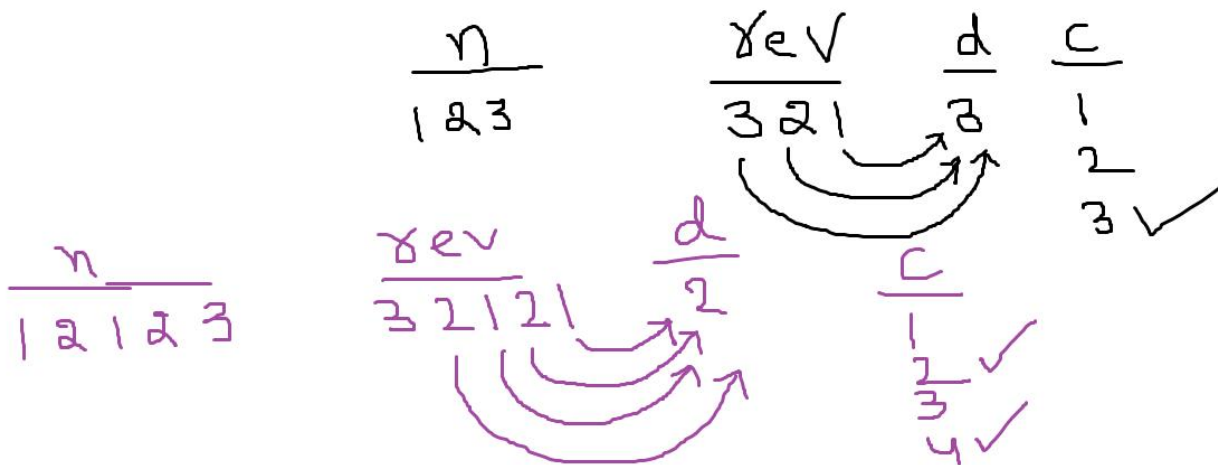
```
TC
Enter a no 123
Enter the digit to search 3
3 in 3 position
_
```

```
TC
Enter a no 123
Enter the digit to search 4
4 not found_
```

```
TC
Enter a no 0
Enter the digit to search 0
0 in 1 position
```

```
TC
Enter a no -1232
Enter the digit to search 2
2 not found
```

```
TC
Enter a no -1212
Enter the digit to search -2
-2 in 2 position
-2 in 4 position
```



```

do
{
if(rev%10==d) ✓
p("%d in %d pos\n",d, c,f=1);
c++; ✓
rev=rev/10; ✓
}while(n!=0); ✓

```

<u>n</u>	<u>rev</u>	<u>d</u>	<u>c</u>
1232	2321 % 10 = 1 = 2	1	1
	232 % 10 = 2 = 2	2	2 ✓
	23 % 10 = 3 = 2	3	3
	2 % 10 = 2 = 2	2	4 ✓

### No to text conversion:

102 ➔ One Zero Two

```
#include<stdio.h>
```

```
#include<conio.h>
```

```
void main()
```

```
{
```

```
long m,n,rev=0; int c=1, d,f=0; clrscr();
```

```
printf("Enter a no "); scanf("%ld",&n);if(n<0)printf("-",n=-n);
```

```
m=n;
```

```
while(m) { rev=rev*10+(m%10);m=m/10; } /* rev */
```

```
do
```

```
{
```

```
switch(rev%10)
{
case 0: printf("Zero");break;
case 1: printf("One");break;
case 2: printf("Two");break;
case 3: printf("Three");break;
case 4: printf("Four");break;
case 5: printf("Five");break;
case 6: printf("Six");break;
case 7: printf("Seven");break;
case 8: printf("Eight");break;
case 9: printf("Nine");break;
}printf(" ");rev=rev/10;
}while(rev!=0);
while(n!=0 && n%10==0)printf("Zero ",n=n/10);
getch();
}
```



```
TC
Enter a no 100
One Zero Zero _
```

```
TC
Enter a no -1002900
-One Zero Zero Two Nine Zero Zero
```

```

Enter a no 0
Zero

```

```

do
{
switch(rev%10)
{
case 0: p("Zero");b;
case 1: p("One");b;
case 2: p("Two");b;
case 9: p("Nine");
}
p(" "); rev=rev/10; ✓
while(rev);

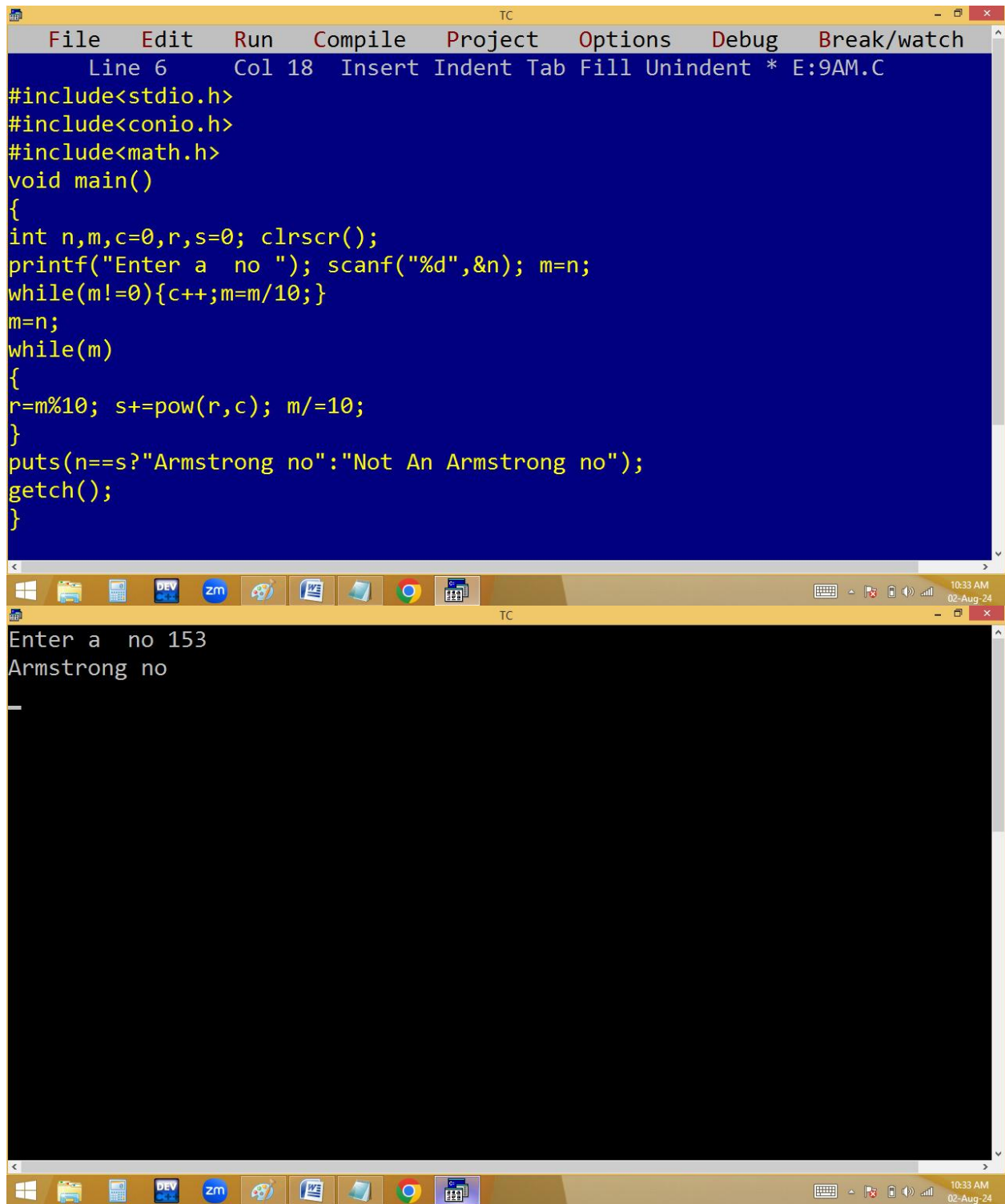
```

$\frac{n}{102}$        $\frac{rev}{208 \div 10 = 1 \text{ one}}$   
 $20 \div 10 = 0 \text{ Zero}$   
 $2 \div 10 = 2 \text{ Two}$

$\frac{n}{100 \div 10 = 0}$        $\frac{rev}{001 \text{ one Zero Zero}}$   
 $10 \div 10 = 0$   
 $1 \div 10 = 0$   
while(n%10==0 && n!=0) p("Zero ", n=n/10);

## Finding Armstrong no:

1 to 9, 153, 370, 371, 407, 1634, 8208,....



The image shows a screenshot of a Turbo C++ (TC) IDE. The top window displays the source code for a program that checks if a number is an Armstrong number. The code includes headers for `stdio.h`, `conio.h`, and `math.h`. It defines a `main` function that prompts the user to enter a number, reads it into `n`, and then uses a `while` loop to calculate the sum of the cubes of its digits. The result is compared to the original number to determine if it is an Armstrong number. The bottom window shows the program's execution: it prompts "Enter a no", the user enters "153", and the program outputs "Armstrong no".

```
File Edit Run Compile Project Options Debug Break/watch
Line 6 Col 18 Insert Indent Tab Fill Unindent * E:9AM.C
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
int n,m,c=0,r,s=0; clrscr();
printf("Enter a no "); scanf("%d",&n); m=n;
while(m!=0){c++;m=m/10;}
m=n;
while(m)
{
r=m%10; s+=pow(r,c); m/=10;
}
puts(n==s?"Armstrong no":"Not An Armstrong no");
getch();
}
```

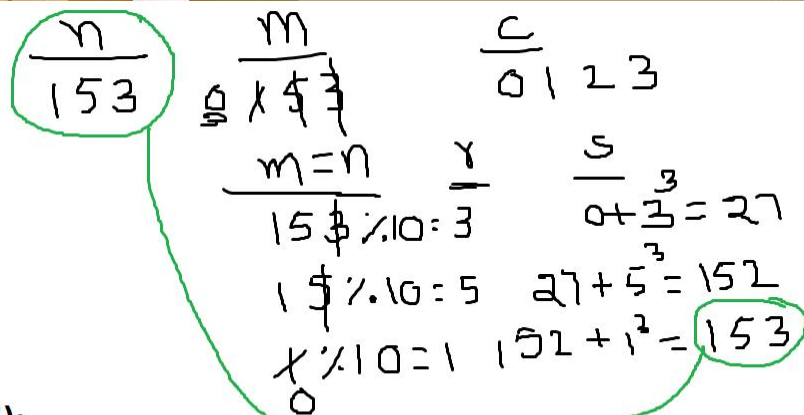
Enter a no 153  
Armstrong no

```
TC
Enter a no 1634
Armstrong no
```

```
TC
Enter a no 9
Armstrong no
_
```

```
TC
Enter a no 143
Not An Armstrong no
```

```
while( m!=0 )
{
  r= m % 10; ✓
  s += pow( r, c );
  m/=10; ✓
}
puts(n==s?"Arm":"Not");
```



143 = 1 + 64 + 27 = 92

## **for loop:**

**It is an entry control loop.**

**for is a keyword.**

**It is also used to repeat a program several times based on a condition.**

**When compared with while and do while, for loop is looking to be smart. In for it is compulsory to maintain two semicolons.**

**For works without condition also and default condition is always 1 i.e. true.**

**Generally for loop is having 3 expressions.**

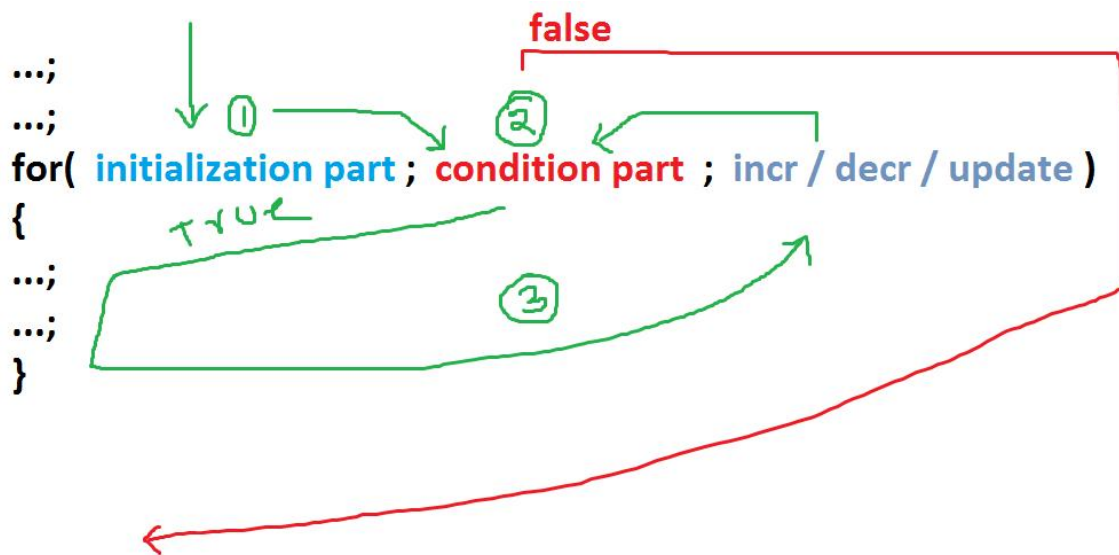
- 1. Initialization**
- 2. Test condition / expression**
- 3. Increment/decrement / updation**

**At first entry of for loop the initialization part is executed and later the test condition is checked. If the condition is true then the for block statements are executed. After completion of the block, the increment or decrement part is executed. Later once again the test condition is evaluated. If it is true then once again for block statements are executed. Like this the process is continued until the condition becomes false. Here the initialization part is executed only once, at the time of loop beginning.**

**It is mandatory to maintain 2 semicolon ( ; ) in a for loop.**

If the for loop is having more than three expressions, it is mandatory to separate the expressions with , separator.

If the for loop is having less than three expressions, then leave the expressions with empty semicolon.





```
for( exp ; exp ; exp )  
{  
}
```

```
for( exp, exp ; exp ; exp, exp )  
{  
}
```

```
for()  
{  
}
```

```
for( ; exp ; )  
{  
}
```

```
for( ; ; )  
{  
}
```

---