

STRUCTURE

It is a user defined data type.

It is a complex data type.

It is collection of heterogeneous variables.

Structure is a user defined, complex data type where we can store and manage more than one variable of different data types under one name.

Structure allows to store both primitive and derived data types (arrays, pointers) at one place, under one name.

In real time applications, data is stored in the form of objects. In this situation,

we need structures. Structures are the foundation for object oriented.

Primitive and derived data types are designed to work with basic data types like int, float and char.

Primitive and derived data types don't support real time requirements. Hence we have to use the user defined data type structure.

Structure allows to carry different types of variables at a time.

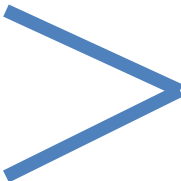
When structure address is available, automatically all the variables address also available. Due to this search time is reduced.

Structure allows to store information in the form of records.

In data files we are using structures very much.

Syntax:

```
struct [ <structure-tag-name> ]  
{  
    datatype    variable;  
    datatype    variable;  
}
```



structure members

[structure_variables] ;

Here struct is a keyword.

Structure tag name is used to identify the structure and it is optional, but required when structure variables declared in other places of the program.

The variables that are declared inside a structure are called **structure members**.

Structure size is sum of all the structure members datatype size.

Without structure members [empty] structure size is **1 byte**.

Structure variables are the **instances** [**copies**] of the structure.

Structure is a **blue-print** [original copy] to create the structure variables.

Structure variable is the **physical representation of a structure.**

When structure variables declared then only memory allocated for structure members.

Every structure should be end with ;

To access the structure members we should have to use the following syntax.

structurevariable.structuremember;

It is called calling / accessing / invoking the structure members.

Here **• (dot)** operator is called

- Member access operator
- Field access operator
- Member of operator
- Membership operator
- Belongs to operator

We can declare structure variables in other places of the program by using below syntax.

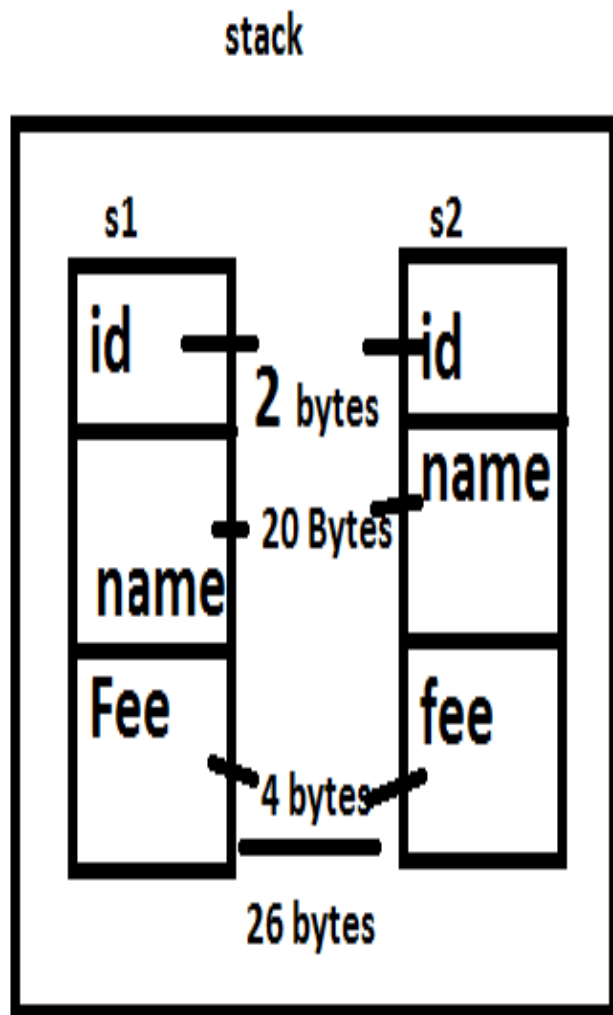
```
struct  structure-tag-name  structure-  
variables;
```

Eg: struct stu s1, s2;

Memory allocation for structure variables

```
struct stu ——— stru tag name  
{  
  int id;  
  char name[20];  
  float fee;  
}  
s1, s2; ——— stru variables
```

stru members



s1.id=100; /* calling structure member */

```
s2.id=200;
```

Finding structure size:

TC

Edit

Line 17 Col 58 Insert Indent Tab Fill Unindent * E:2PM.C

```
#include<stdio.h>
#include<conio.h>
struct stu
{
int id;
char name[20];
float fee;
}s;
struct empty
{
};
void main()
{
struct empty e; /* stru var */
printf("stu size=%d, empty size=%d\n",sizeof(s),sizeof(e));
getch();
}
```

Activate Windows
Go to PC settings to activate Windows.

3:20 PM
16-Oct-23

TC

```
stu size=26, empty size=1
```

Activate Windows
Go to PC settings to activate Windows.

Page: 8 of 10 Words: 473

110%

3:20 PM
16-Oct-23

Eg: Direct initialization of structure members:

It is the process of passing values for structure members, without using scanf() at design time using =.

Note: In direct initialization of structure members, the passing values datatype and structure members datatype should be matched.

When all the structure members are not initialized, they will store the default values as follows.

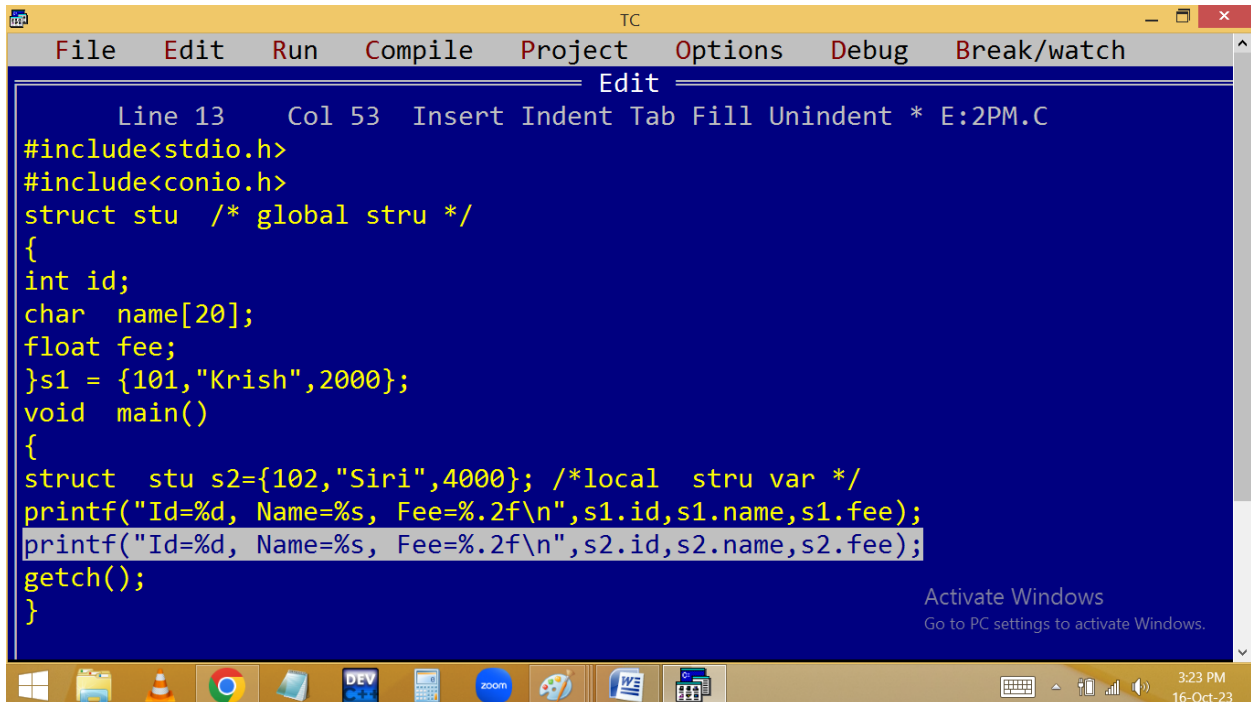
Int – 0

Float – 0.000000

Char – blank space

structures stores 0, 0.00 and blank in int, float and char.

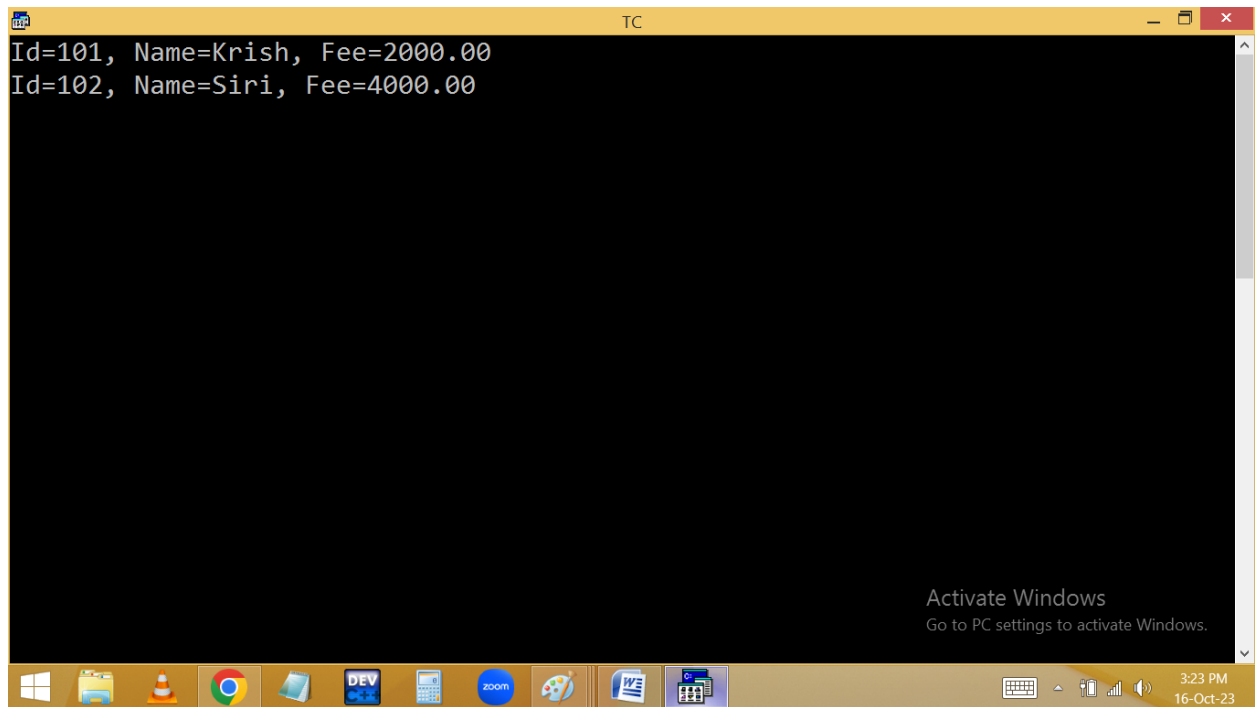
when it is a local structure, without initialization, stores garbage values.



The screenshot shows the Turbo C++ (TC) IDE interface. The menu bar includes File, Edit, Run, Compile, Project, Options, Debug, and Break/watch. The status bar at the top indicates 'Line 13 Col 53 Insert Indent Tab Fill Unindent * E:2PM.C'. The code in the editor is as follows:

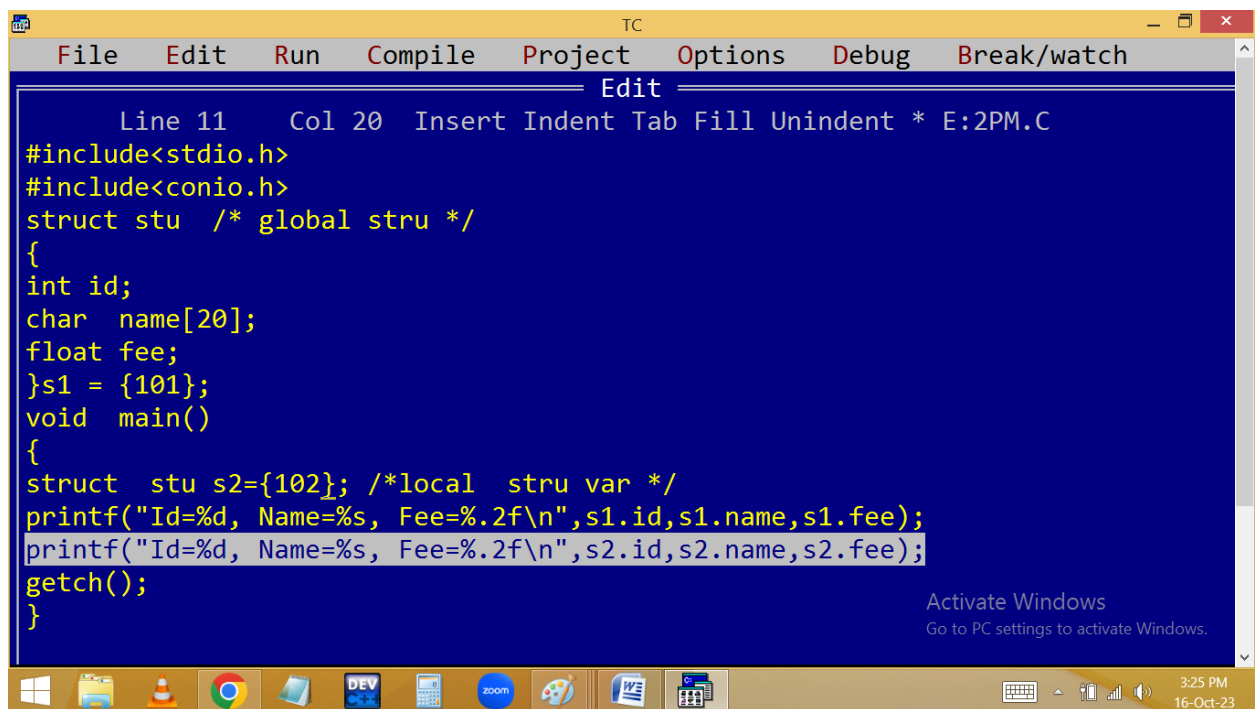
```
#include<stdio.h>
#include<conio.h>
struct stu /* global stru */
{
int id;
char name[20];
float fee;
}s1 = {101,"Krish",2000};
void main()
{
struct stu s2={102,"Siri",4000}; /*local stru var */
printf("Id=%d, Name=%s, Fee=%.2f\n",s1.id,s1.name,s1.fee);
printf("Id=%d, Name=%s, Fee=%.2f\n",s2.id,s2.name,s2.fee);
getch();
}
```

An 'Activate Windows' watermark is visible in the bottom right corner of the code editor area. The Windows taskbar at the bottom shows various application icons and the system clock indicating 3:23 PM on 16-Oct-23.



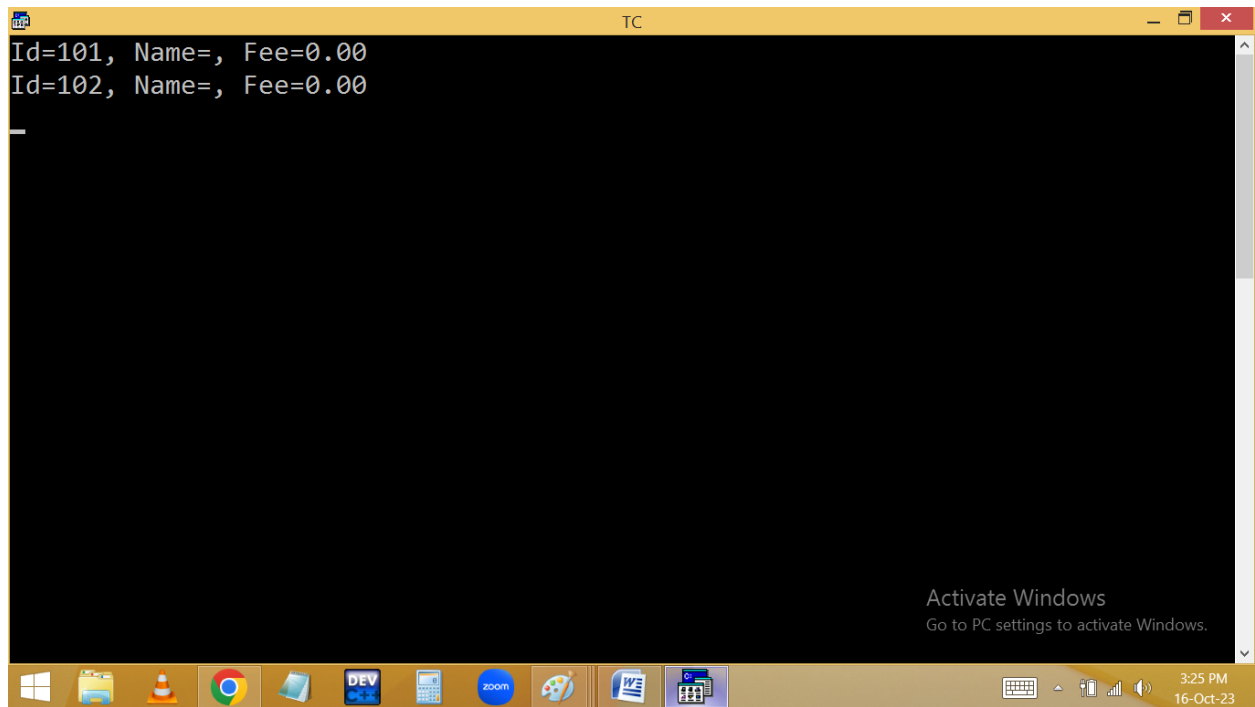
A screenshot of a Windows 10 desktop. A terminal window titled "TC" is open, displaying the output of a program. The output consists of two lines: "Id=101, Name=Krish, Fee=2000.00" and "Id=102, Name=Siri, Fee=4000.00". The taskbar at the bottom shows various application icons including File Explorer, Chrome, and several development tools. The system tray on the right indicates the time is 3:23 PM on 16-Oct-23. An "Activate Windows" watermark is visible in the bottom right corner of the terminal window.

```
Id=101, Name=Krish, Fee=2000.00
Id=102, Name=Siri, Fee=4000.00
```



A screenshot of a Windows 10 desktop. A code editor window titled "TC" is open, showing a C program. The code defines a structure "stu" with fields "id", "name", and "fee", and a "main" function that prints the details of two students. The taskbar and system tray are identical to the first screenshot. An "Activate Windows" watermark is visible in the bottom right corner of the code editor window.

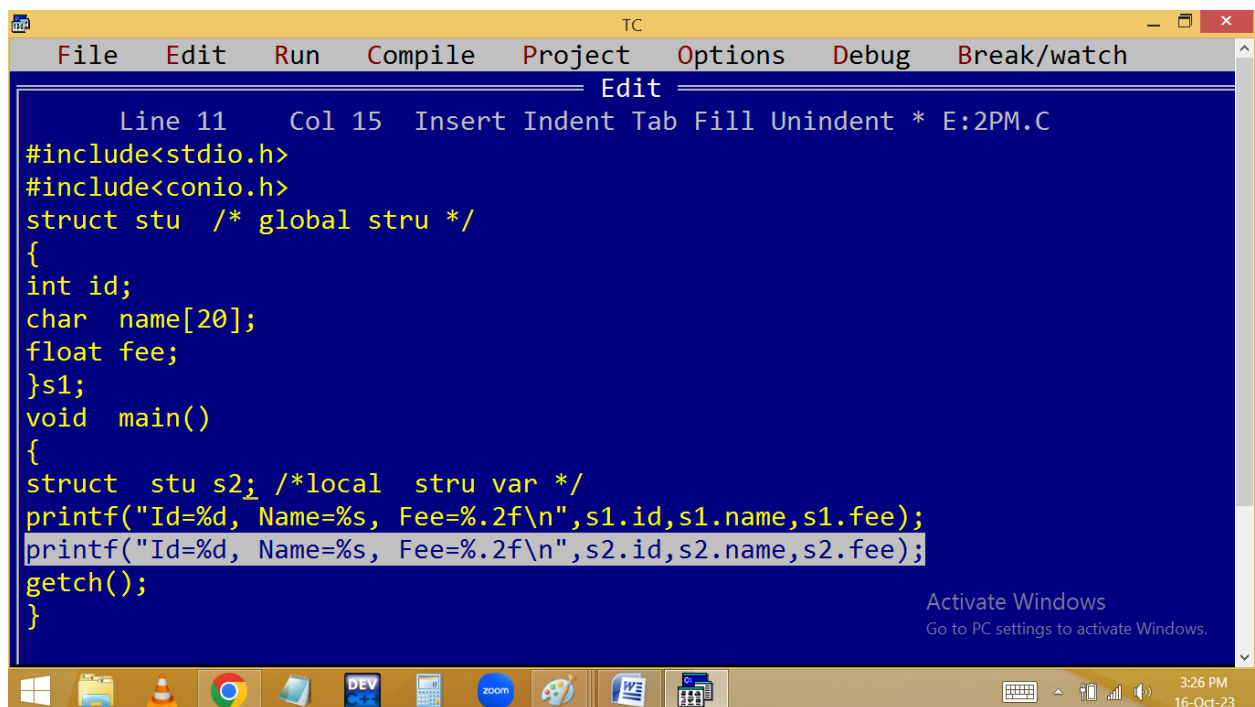
```
Line 11 Col 20 Insert Indent Tab Fill Unindent * E:2PM.C
#include<stdio.h>
#include<conio.h>
struct stu /* global stru */
{
int id;
char name[20];
float fee;
}s1 = {101};
void main()
{
struct stu s2={102}; /*local stru var */
printf("Id=%d, Name=%s, Fee=%.2f\n",s1.id,s1.name,s1.fee);
printf("Id=%d, Name=%s, Fee=%.2f\n",s2.id,s2.name,s2.fee);
getch();
}
```



The screenshot shows a Turbo C++ (TC) console window. The title bar is yellow and contains the text "TC". The main area is black with white text. The output of the program is displayed as follows:

```
Id=101, Name=, Fee=0.00
Id=102, Name=, Fee=0.00
```

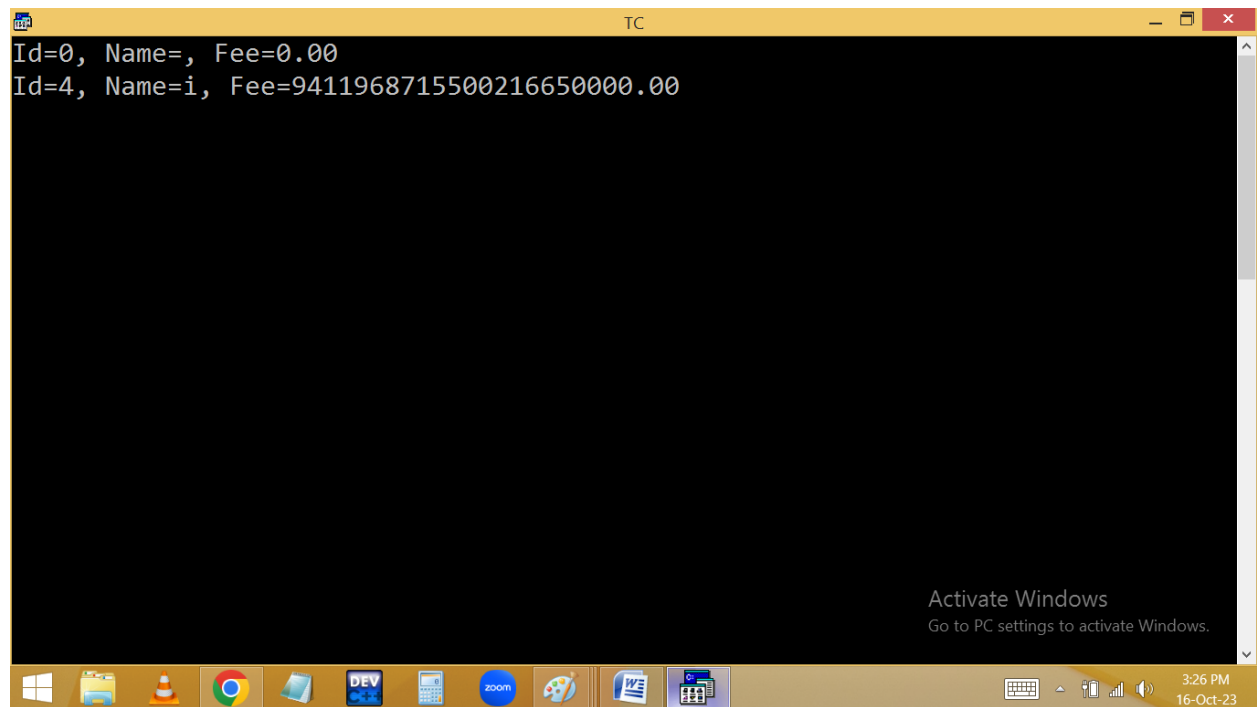
At the bottom right of the console window, there is a message: "Activate Windows Go to PC settings to activate Windows." The Windows taskbar is visible at the bottom, showing various application icons and the system clock indicating 3:25 PM on 16-Oct-23.



The screenshot shows a Turbo C++ (TC) editor window. The title bar is yellow and contains the text "TC". The menu bar includes "File", "Edit", "Run", "Compile", "Project", "Options", "Debug", and "Break/watch". The main area is blue with yellow text. The code being edited is as follows:

```
Line 11 Col 15 Insert Indent Tab Fill Unindent * E:2PM.C
#include<stdio.h>
#include<conio.h>
struct stu /* global stru */
{
int id;
char name[20];
float fee;
}s1;
void main()
{
struct stu s2; /*local stru var */
printf("Id=%d, Name=%s, Fee=%.2f\n",s1.id,s1.name,s1.fee);
printf("Id=%d, Name=%s, Fee=%.2f\n",s2.id,s2.name,s2.fee);
getch();
}
```

At the bottom right of the editor window, there is a message: "Activate Windows Go to PC settings to activate Windows." The Windows taskbar is visible at the bottom, showing various application icons and the system clock indicating 3:26 PM on 16-Oct-23.

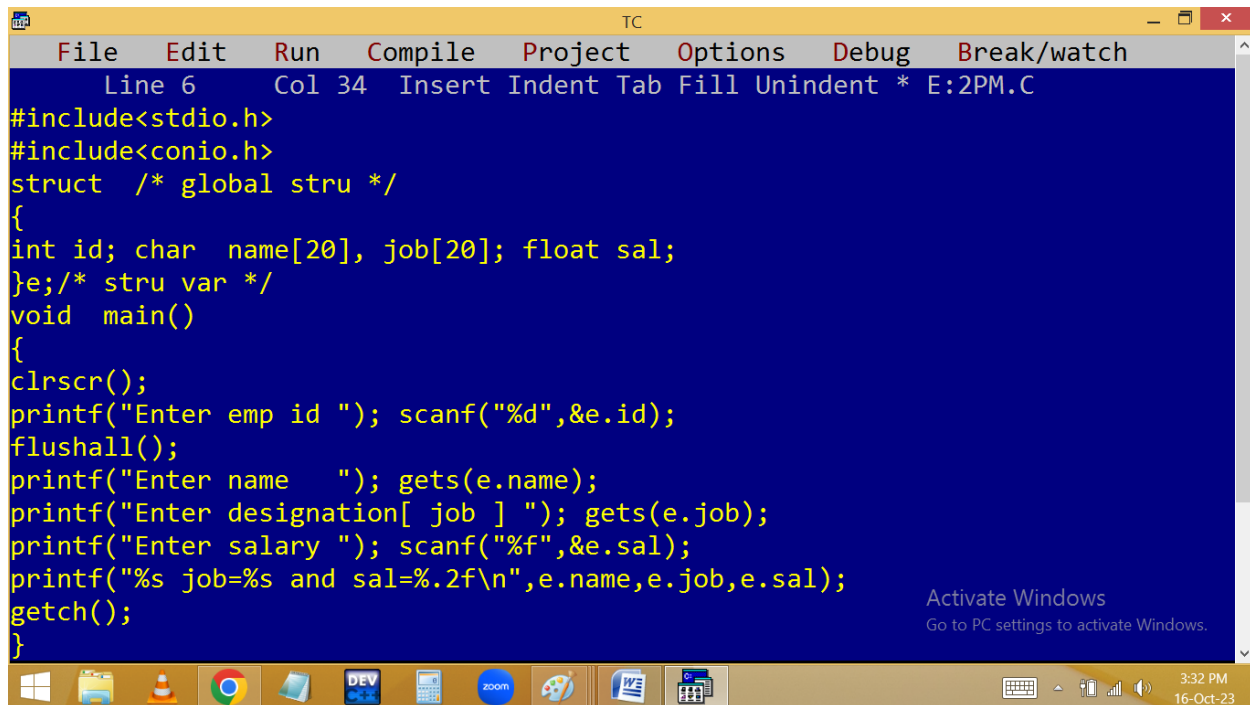


```
Id=0, Name=, Fee=0.00
Id=4, Name=i, Fee=9411968715500216650000.00
```

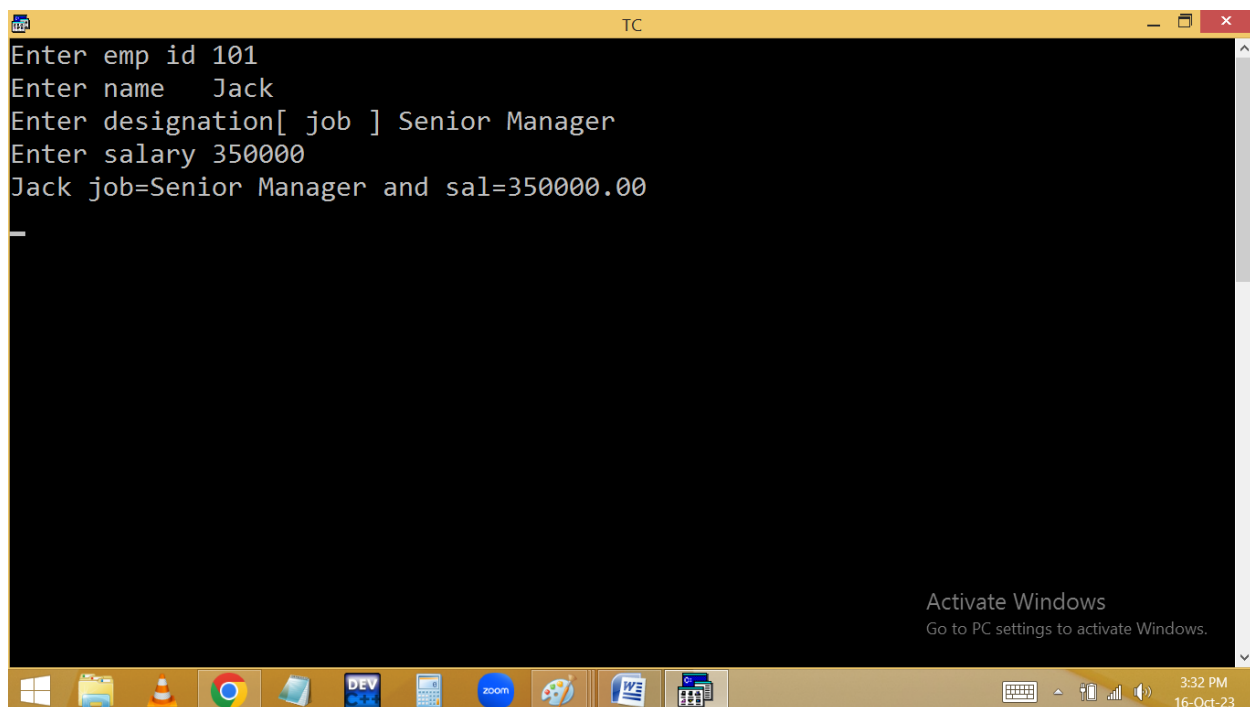
Activate Windows
Go to PC settings to activate Windows.

3:26 PM
16-Oct-23

Reading and printing structure data:



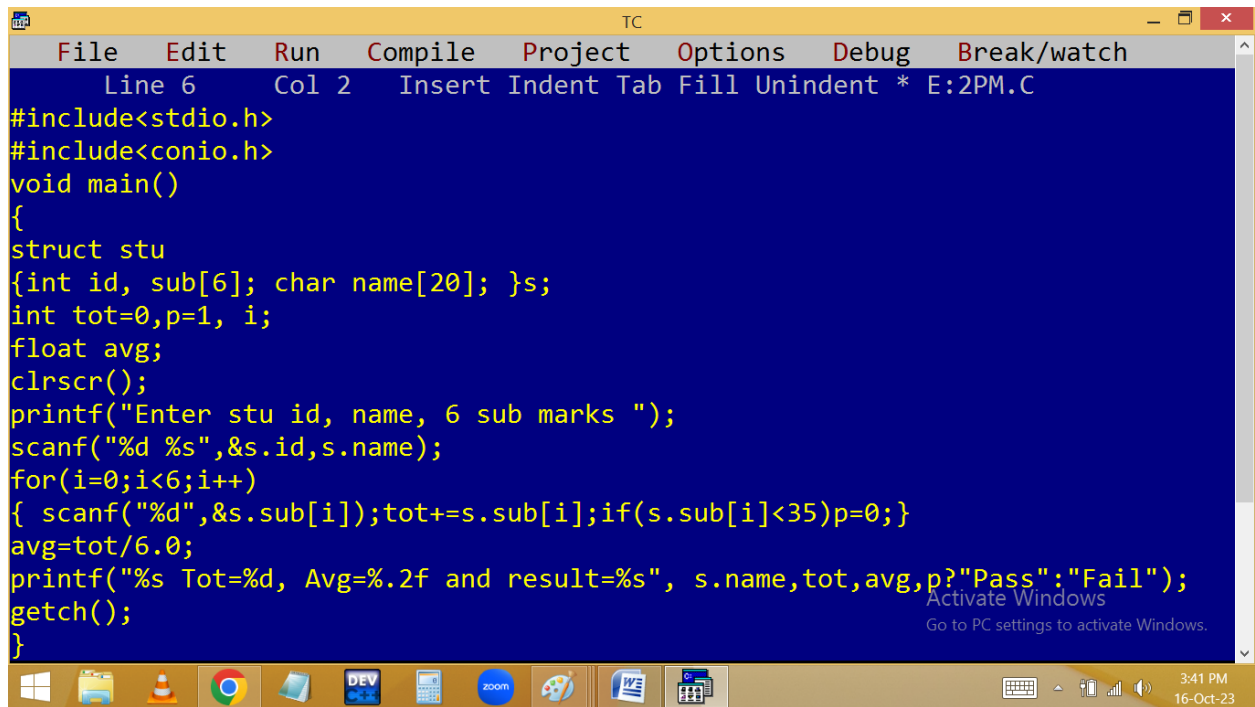
```
TC
File Edit Run Compile Project Options Debug Break/watch
Line 6 Col 34 Insert Indent Tab Fill Unindent * E:2PM.C
#include<stdio.h>
#include<conio.h>
struct /* global stru */
{
int id; char name[20], job[20]; float sal;
}e;/* stru var */
void main()
{
clrscr();
printf("Enter emp id "); scanf("%d",&e.id);
flushall();
printf("Enter name "); gets(e.name);
printf("Enter designation[ job ] "); gets(e.job);
printf("Enter salary "); scanf("%f",&e.sal);
printf("%s job=%s and sal=%.2f\n",e.name,e.job,e.sal);
getch();
}
```



```
TC
Enter emp id 101
Enter name Jack
Enter designation[ job ] Senior Manager
Enter salary 350000
Jack job=Senior Manager and sal=350000.00
-
Activate Windows
Go to PC settings to activate Windows.
```

Finding stu tot,avg and pass/fail using structure:

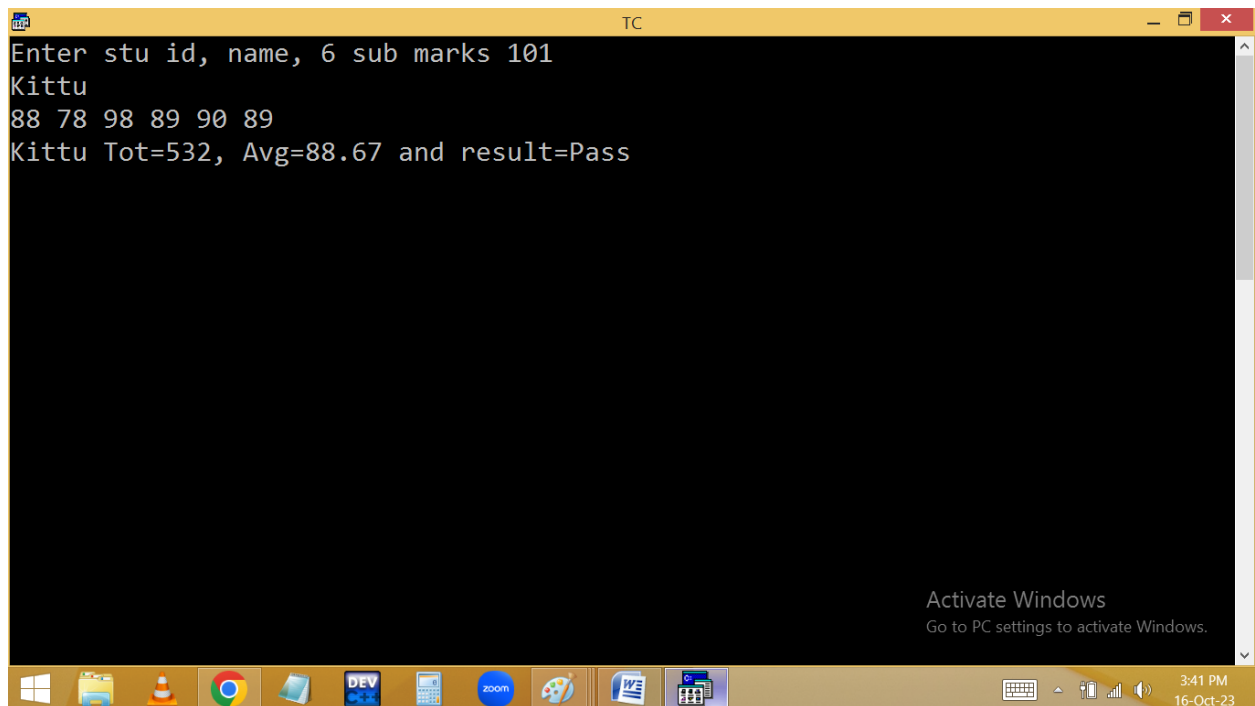
Using array of structure members:



```
TC
File Edit Run Compile Project Options Debug Break/watch
Line 6 Col 2 Insert Indent Tab Fill Unindent * E:2PM.C
#include<stdio.h>
#include<conio.h>
void main()
{
struct stu
{int id, sub[6]; char name[20]; }s;
int tot=0,p=1, i;
float avg;
clrscr();
printf("Enter stu id, name, 6 sub marks ");
scanf("%d %s",&s.id,s.name);
for(i=0;i<6;i++)
{ scanf("%d",&s.sub[i]);tot+=s.sub[i];if(s.sub[i]<35)p=0;}
avg=tot/6.0;
printf("%s Tot=%d, Avg=%.2f and result=%s", s.name,tot,avg,p?"Pass":"Fail");
getch();
}
```

Activate Windows
Go to PC settings to activate Windows.

3:41 PM
16-Oct-23



```
TC
Enter stu id, name, 6 sub marks 101
Kittu
88 78 98 89 90 89
Kittu Tot=532, Avg=88.67 and result=Pass

Activate Windows
Go to PC settings to activate Windows.

3:41 PM
16-Oct-23
```