Subject Title: Programming in ‘C’

Subject Code: CAM203-1C

Unit-2 Structures and Union

Prepared by Ankit Rami(AR)

Any Query Contact – 8460467193

Email – ankitramiblog@gmail.com

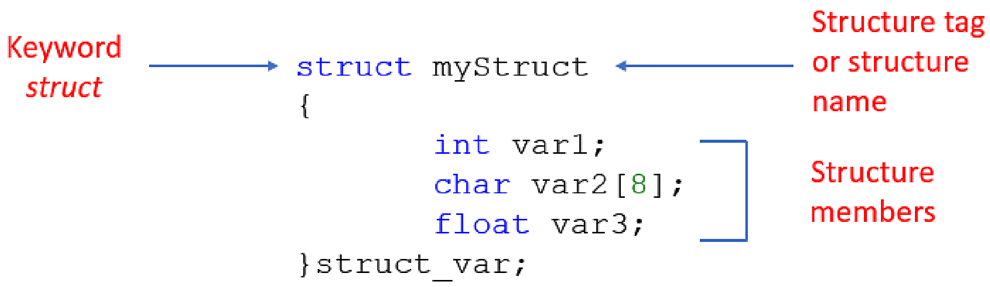
Introduction for Structures(Struct)

- ✓ During programming, there is a need to store multiple logically related elements under one roof.
- ✓ Ex- Employee’s details like name, employee number, and designation need to be stored together. In such cases, the C language provides structures to do the job for us.
- ✓ Structures (also called struct) are a way to group several related variables into one place. Each variable in the structure is known as a **member** of the structure.
- ✓ The structure in C is a user-defined data type that can be used to group items of possibly different types into a single type.
- ✓ **“Structure in C is a user-defined data type that enables us to store the collection of different data types.”**

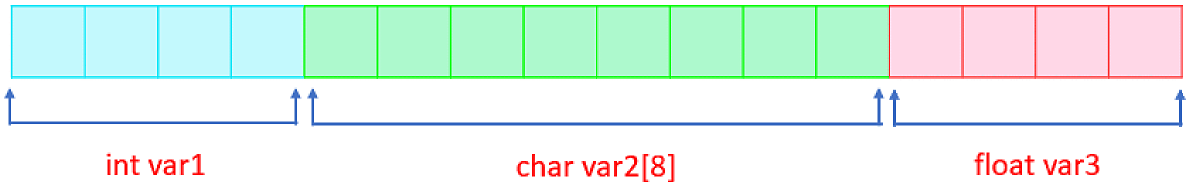
Introduction for Structures(Struct)

- ***Structure Syntax***

✓ A structure can be defined as a single entity holding variables of different data types that are logically related to each other. All the data members inside a structure are accessible to the functions defined outside the structure. To access the data members in the main function, you need to create a structure variable.



```
struct myStruct
{
        int var1;
        char var2[8];
        float var3;
}struct_var;
```

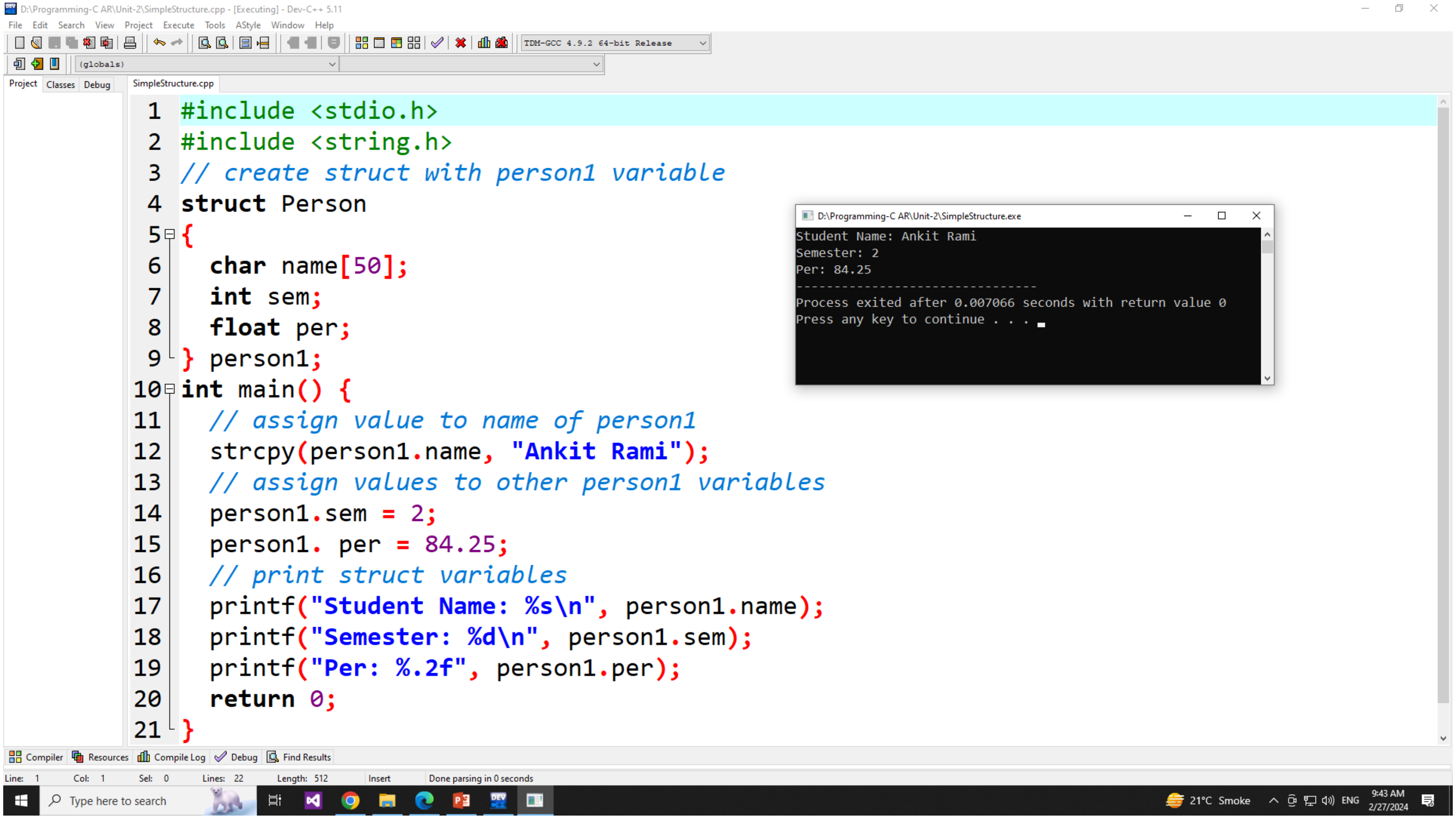


Memory representation for struct_var

Why Do We Use Structures in C?

- ✓ Structure in C programming is very helpful in cases where we need to store similar data of multiple entities. Let us understand the need for structures with a real-life example. Suppose you need to manage the record of books in a library. Now a book can have properties like book_name, author_name, and genre. So, for any book you need three variables to store its records. Now, there are two ways to achieve this goal.
- ✓ The first and the naive one is to create separate variables for each book. But creating so many variables and assigning values to each of them is impractical. So what would be the optimized and ideal approach? Here, comes the structure in the picture.
- ✓ We can define a structure where we can declare the data members of different data types according to our needs. In this case, a structure named BOOK can be created having three members book_name, author_name, and genre. Multiple variables of the type BOOK can be created such as book1, book2, and so on (each will have its own copy of the three members book_name, author_name, and genre).

Simple Structures Example


D:\Programming-C AR\Unit-2\SimpleStructure.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
SimpleStructure.cpp
D:\Programming-C AR\Unit-2\SimpleStructure.exe
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 1 Sel: 0 Lines: 22 Length: 512 Insert Done parsing in 0 seconds
Type here to search
21°C Smoke
ENG
9:43 AM
2/27/2024

```c
#include <stdio.h>
#include <string.h>
// create struct with person1 variable
struct Person
{
  char name[50];
  int sem;
  float per;
} person1;
int main() {
  // assign value to name of person1
  strcpy(person1.name, "Ankit Rami");
  // assign values to other person1 variables
  person1.sem = 2;
  person1. per = 84.25;
  // print struct variables
  printf("Student Name: %s\n", person1.name);
  printf("Semester: %d\n", person1.sem);
  printf("Per: %.2f", person1.per);
  return 0;
}
```

```
Student Name: Ankit Rami
Semester: 2
Per: 84.25
--------------------------------
Process exited after 0.007066 seconds with return value 0
Press any key to continue . . .
```

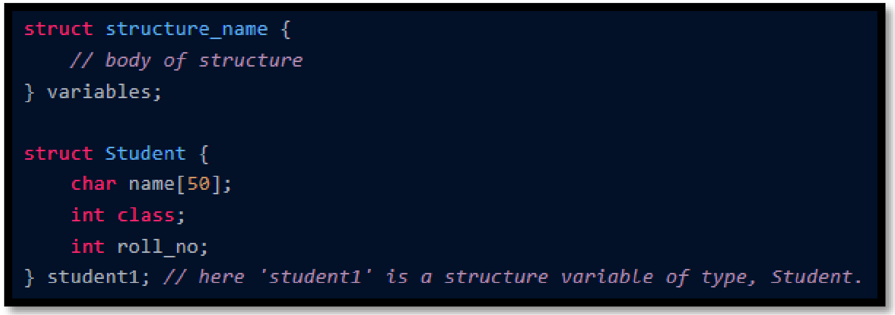
Prepared by Ankit Rami(AR) | Any Query Contact – 8460467193 | Email – ankitramiblog@gmail.com

Declaration and Initialization of Structure

- ***C Structure Definition***

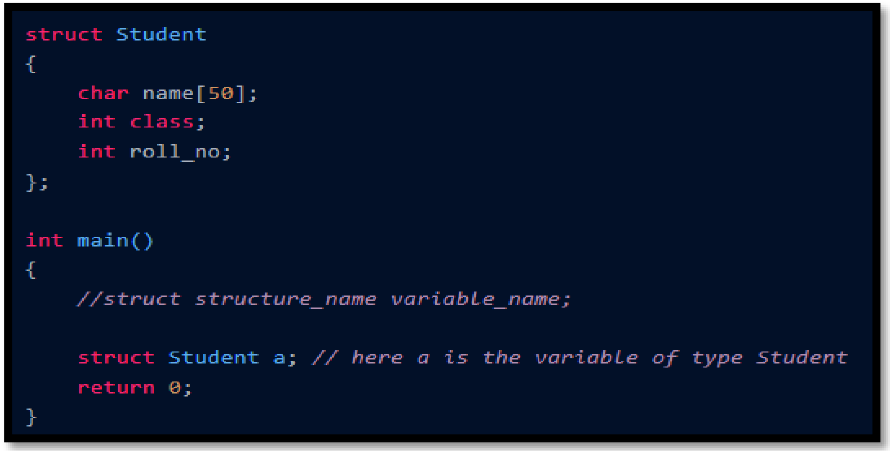
✓ To use structure in our program, we have to define its instance. We can do that by creating variables of the structure type. We can define structure variables using two methods:

1. Structure Variable Declaration with Structure Template



```c
struct structure_name {
    // body of structure
} variables;

struct Student {
    char name[50];
    int class;
    int roll_no;
} student1; // here 'student1' is a structure variable of type, Student.
```

2. Structure Variable Declaration after Structure Template

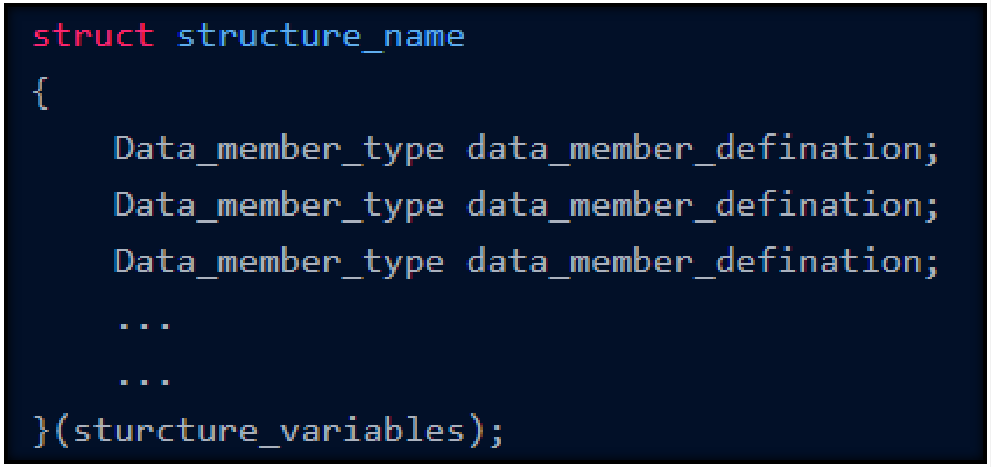


```c
struct Student
{
    char name[50];
    int class;
    int roll_no;
};

int main()
{
    //struct structure_name variable_name;

    struct Student a; // here a is the variable of type Student
    return 0;
}
```

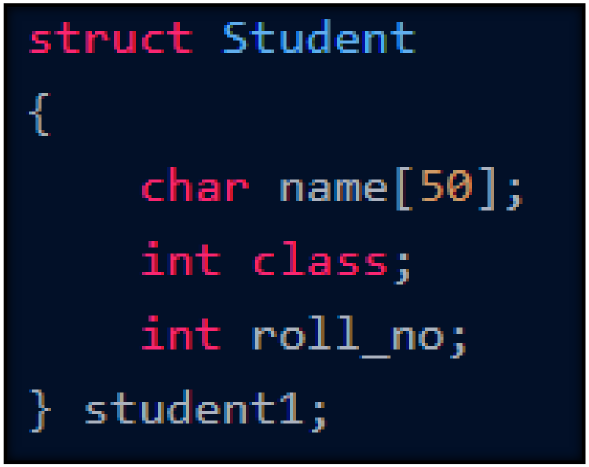
Declaration and Initialization of Structure

- ***C Structure Declaration***

✓ We have to declare structure in C before using it in our program. In structure declaration, we specify its member variables along with their **Data Type**. We can use the **Struct** keyword to declare the structure in C using the following syntax



```
struct structure_name
{
    Data_member_type data_member_defination;
    Data_member_type data_member_defination;
    Data_member_type data_member_defination;
    ...
    ...
}(sturcture_variables);
```



```
struct Student
{
    char name[50];
    int class;
    int roll_no;
} student1;
```

Declaration and Initialization of Structure

- ***Accessing Members of the Structure***

✓ To access structure members and create multiple structure variables with different values using just one structure definition.

✓ There are two ways to access structure members:

1. By . (member or dot operator)

Ex- p1.name;

2. By -> (structure pointer operator)

Ex- p1->name;

Declaration and Initialization of Structure

- ***Initialize Structure Members***
- We can initialize structure members in 3 ways which are as follows:

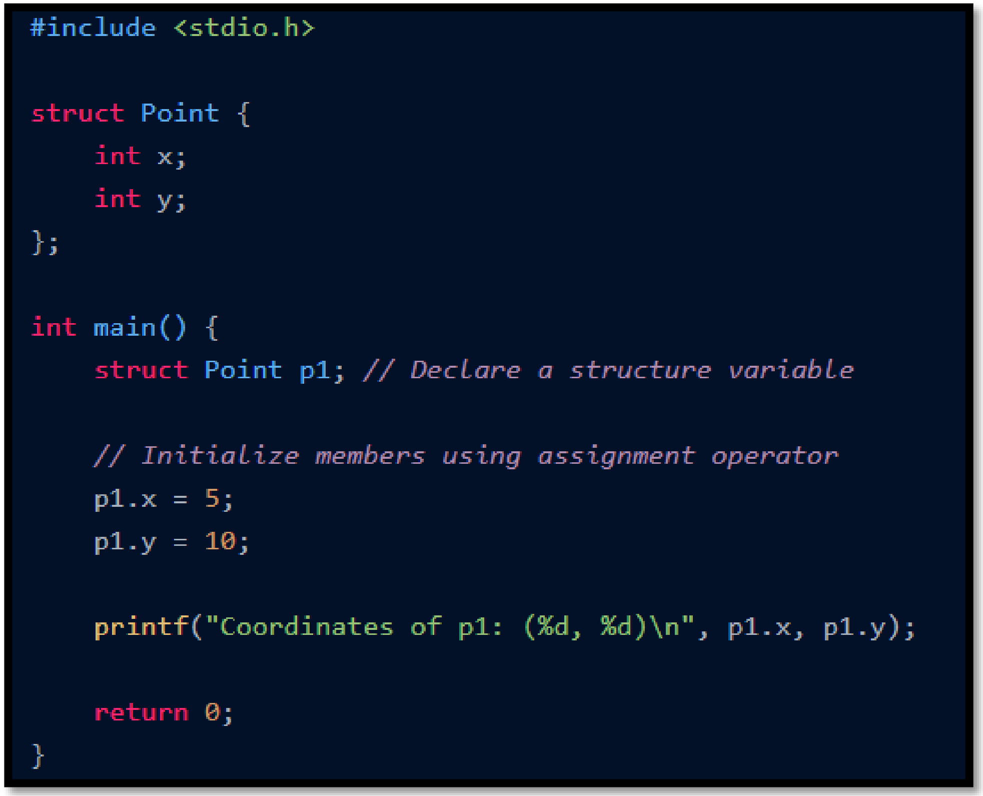
1. **Using Assignment Operator.**
2. **Using Initializer List.**
3. **Using Designated Initializer List.**

Declaration and Initialization of Structure

- ***Initialize Structure Members***

✓ **Using Assignment Operator.**

✓ You can initialize structure members individually using the assignment operator (=) after declaring the structure variable.



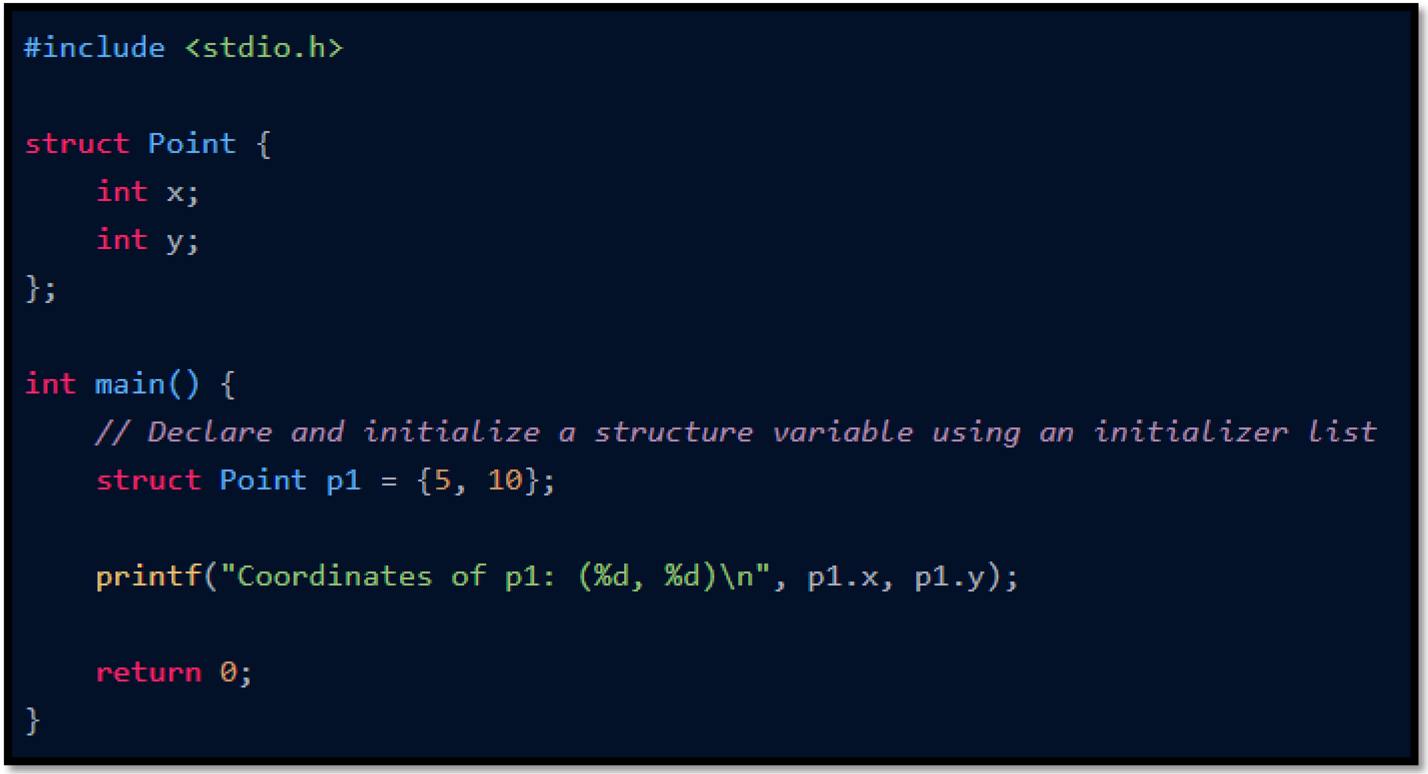
```c
#include <stdio.h>

struct Point {
    int x;
    int y;
};

int main() {
    struct Point p1; // Declare a structure variable

    // Initialize members using assignment operator
    p1.x = 5;
    p1.y = 10;

    printf("Coordinates of p1: (%d, %d)\n", p1.x, p1.y);

    return 0;
}
```

Declaration and Initialization of Structure

- ***Initialize Structure Members***

✓ **Initialization using Initializer List**

✓ You can also initialize structure members using an initializer list when declaring the structure variable.



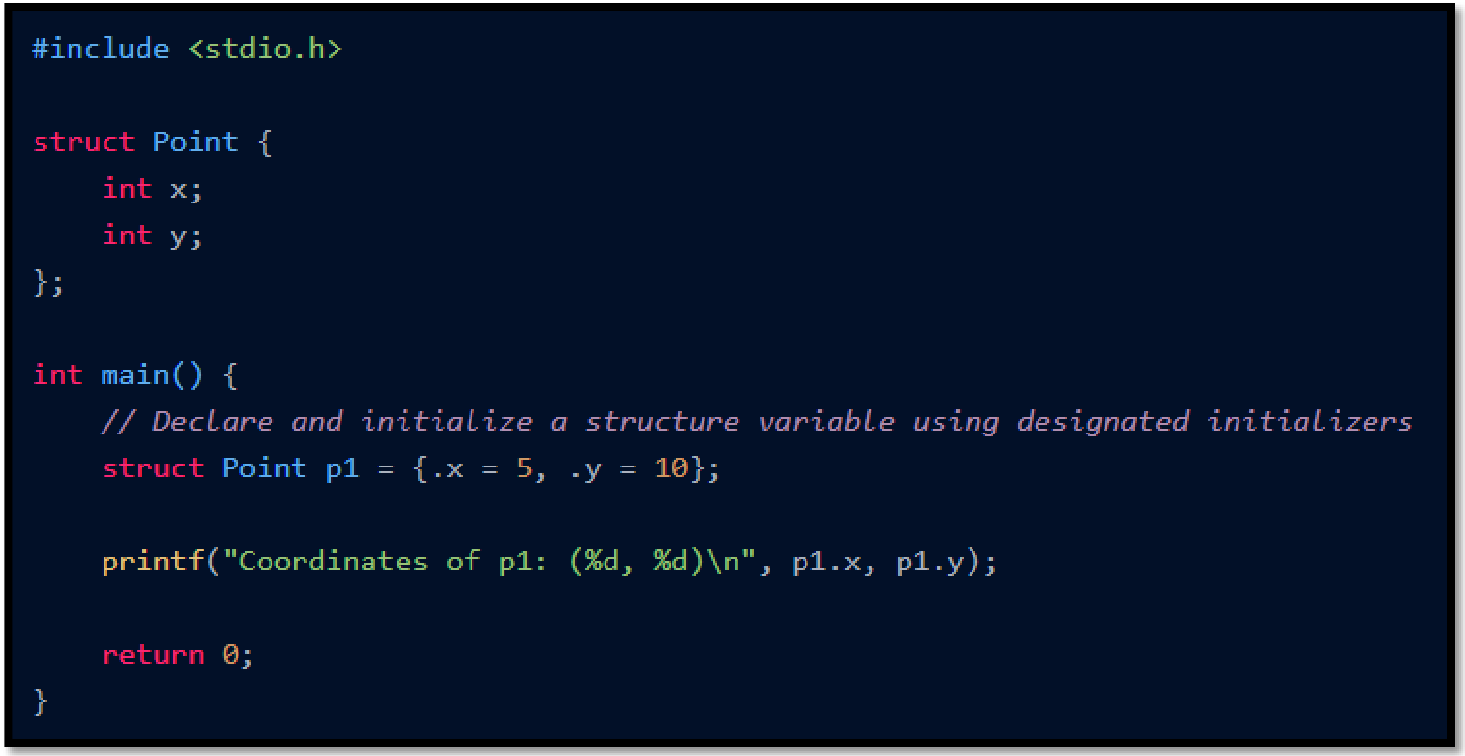
```c
#include <stdio.h>

struct Point {
    int x;
    int y;
};

int main() {
    // Declare and initialize a structure variable using an initializer list
    struct Point p1 = {5, 10};

    printf("Coordinates of p1: (%d, %d)\n", p1.x, p1.y);

    return 0;
}
```

Declaration and Initialization of Structure

- ***Initialize Structure Members***

✓ **Initialization using Designated Initializer List**

✓ Designated initializers allow you to specify the member you want to initialize explicitly. This is especially useful when you want to initialize only specific members of a structure.



```c
#include <stdio.h>

struct Point {
    int x;
    int y;
};

int main() {
    // Declare and initialize a structure variable using designated initializers
    struct Point p1 = {.x = 5, .y = 10};

    printf("Coordinates of p1: (%d, %d)\n", p1.x, p1.y);

    return 0;
}
```

Comparison of Structures and Arrays

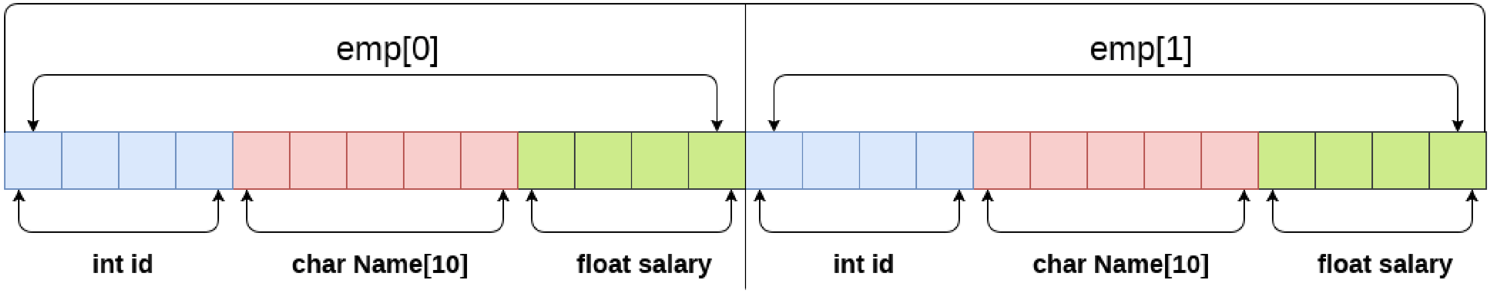
Array	Structure
Array is a collection of similar type of data(Homogenous Data)	Structure is a collection of Different type of data(Heterogeneous Data)
Array Size is Fixed	Structure size is not fixed
Array elements are access using index number Ex- ar[5]	Structure elements are access using (.) and (->) operator
Array allocate static memory	Structure allocate dynamic memory
Array elements takes less time than structure	Structure elements takes more time than Array.
In Array no any keyword to represent Array	In Structure 'struct' keyword to represent Structure

Ex- int arr[5] = { 10, 20, 30, 40, 50 };	Ex- struct Person {
	char name[50];
	int citNo;
	float salary;
	} person1;

Array of Structures in C

- ✓ An array of Structures in C can be defined as the collection of multiple structures variables where each variable contains information about different entities. The array of structures in C are used to store information about multiple entities of different data types. The array of structures is also known as the collection of structures.

Array of structures

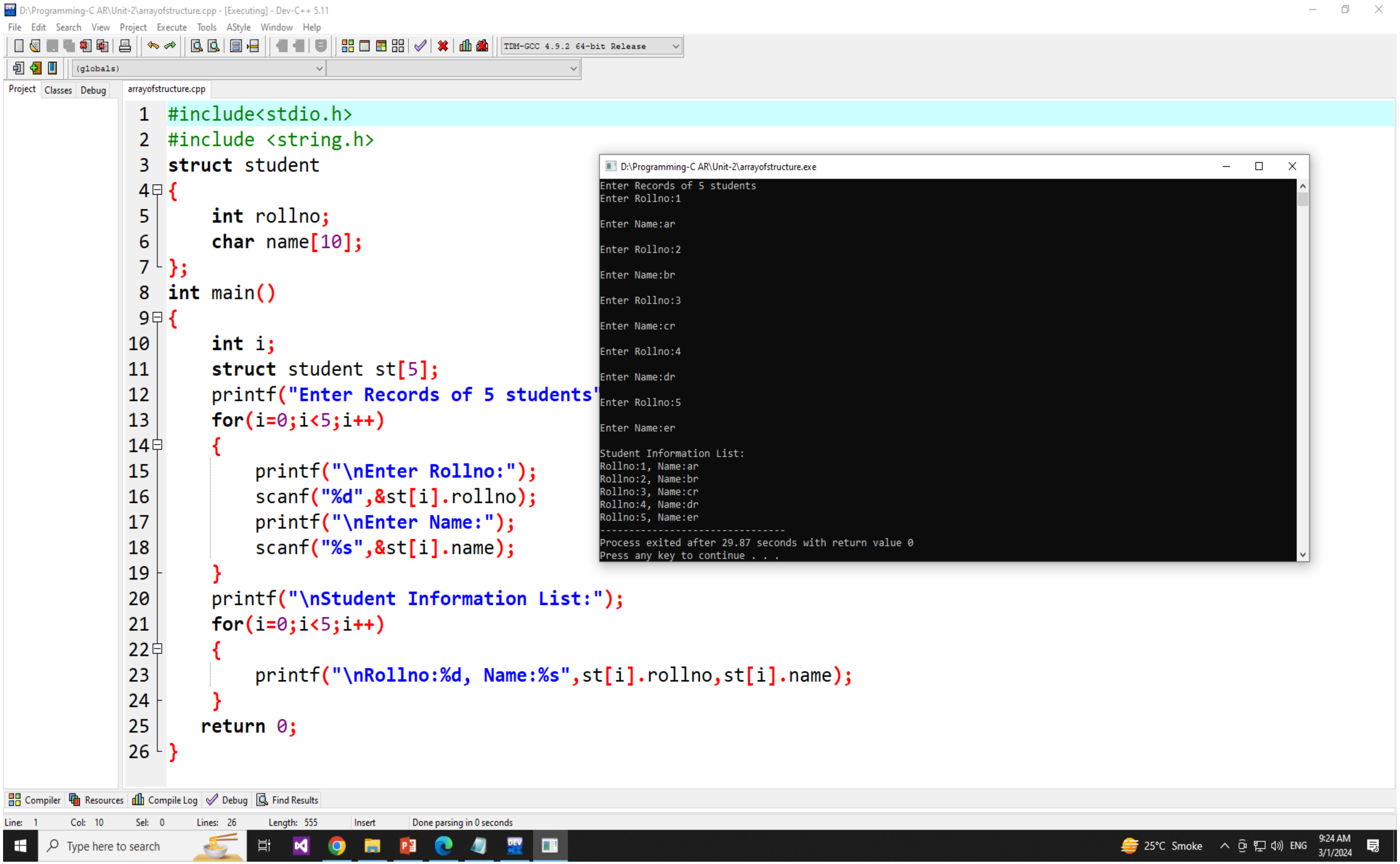


```
struct employee
{
    int id;
    char name[5];
    float salary;
};
struct employee emp[2];
```

sizeof (emp) = 4 + 5 + 4 = 13 bytes

sizeof (emp[2]) = 26 bytes

Array of Structures in C

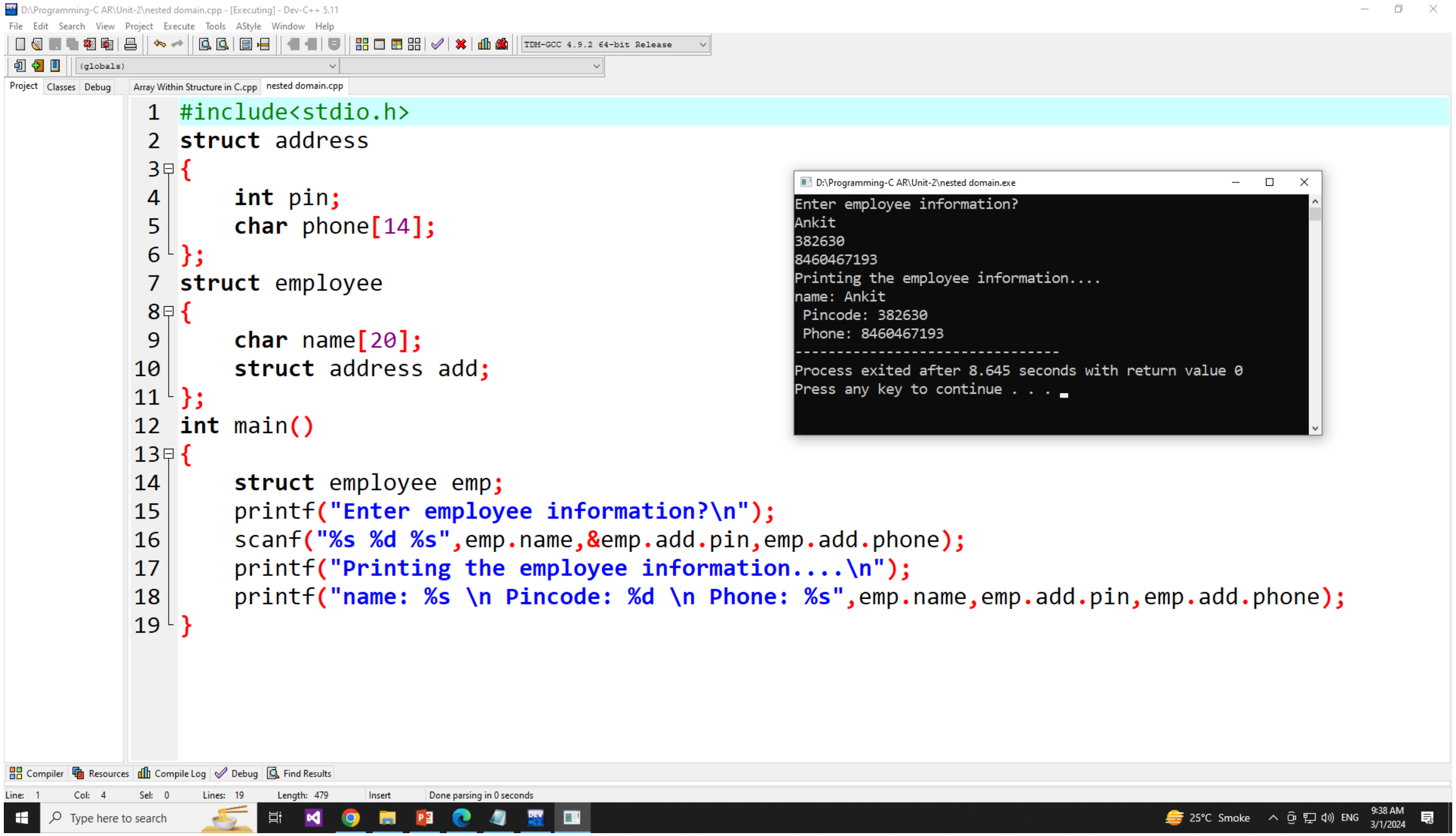

D:\Programming-C AR\Unit-2\arrayofstructure.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
arrayofstructure.cpp
D:\Programming-C AR\Unit-2\arrayofstructure.exe
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 10 Sel: 0 Lines: 26 Length: 555 Insert Done parsing in 0 seconds
Type here to search
25°C Smoke ENG 9:24 AM 3/1/2024

```c
#include<stdio.h>
#include <string.h>
struct student
{
    int rollno;
    char name[10];
};
int main()
{
    int i;
    struct student st[5];
    printf("Enter Records of 5 students'
    for(i=0;i<5;i++)
    {
        printf("\nEnter Rollno:");
        scanf("%d",&st[i].rollno);
        printf("\nEnter Name:");
        scanf("%s",&st[i].name);
    }
    printf("\nStudent Information List:");
    for(i=0;i<5;i++)
    {
        printf("\nRollno:%d, Name:%s",st[i].rollno,st[i].name);
    }
   return 0;
}
```

```
Enter Records of 5 students
Enter Rollno:1

Enter Name:ar

Enter Rollno:2

Enter Name:br

Enter Rollno:3

Enter Name:cr

Enter Rollno:4

Enter Name:dr

Enter Rollno:5

Enter Name:er

Student Information List:
Rollno:1, Name:ar
Rollno:2, Name:br
Rollno:3, Name:cr
Rollno:4, Name:dr
Rollno:5, Name:er
--------------------------------
Process exited after 29.87 seconds with return value 0
Press any key to continue . . .
```

Concept of Structures within Structures

- ✓ Concept of structures within structures is also called Nested Structure in C.
- ✓ C provides us the feature of nesting one structure within another structure by using which, complex data types are created. For example, we may need to store the address of an entity employee in a structure. The attribute address may also have the subparts as street number, city, state, and pin code. Hence, to store the address of the employee, we need to store the address of the employee into a separate structure and nest the structure address into the structure employee.

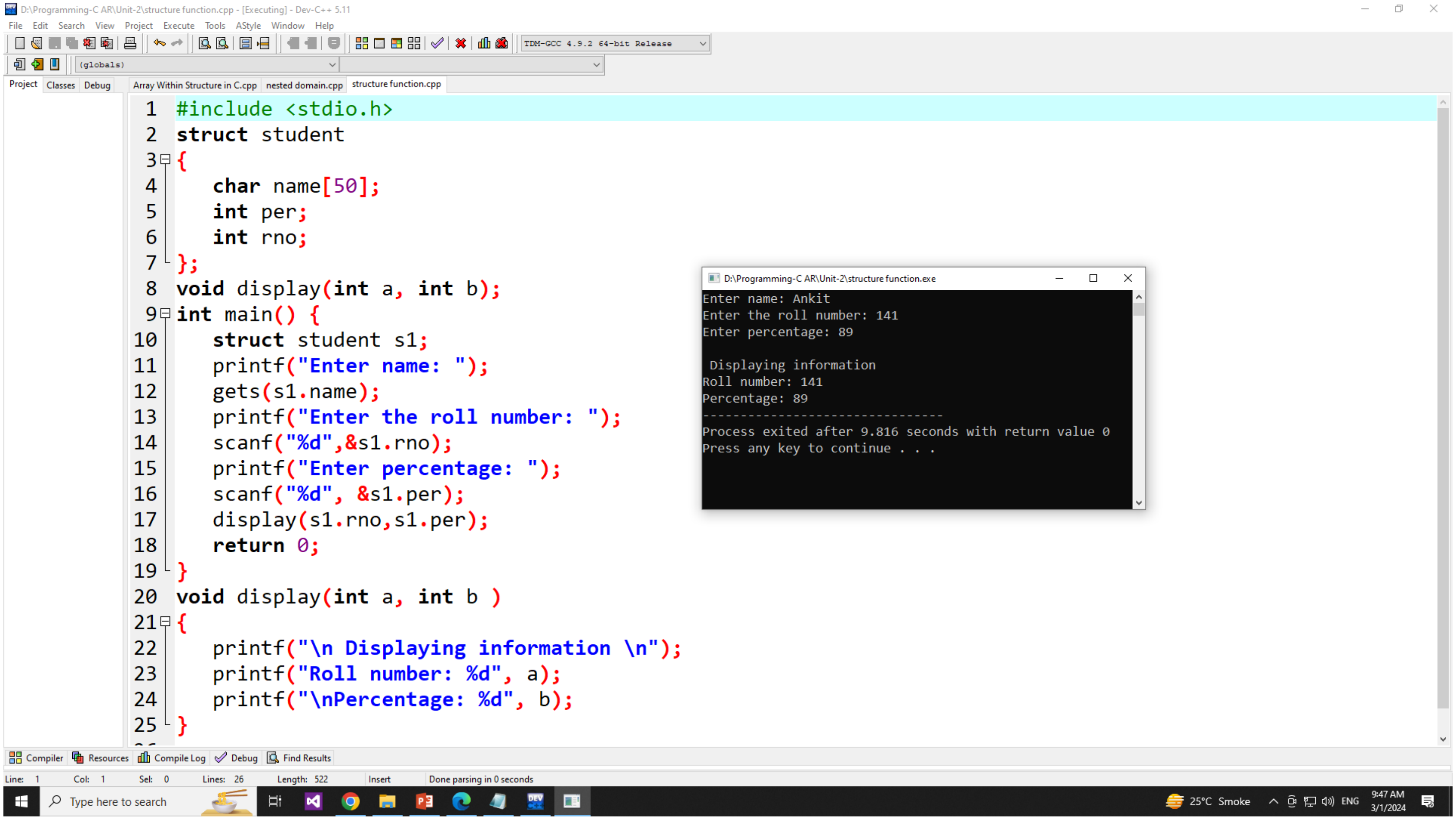

D:\Programming-C AR\Unit-2\nested domain.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
Array Within Structure in C.cpp
nested domain.cpp
D:\Programming-C AR\Unit-2\nested domain.exe
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 4 Sel: 0 Lines: 19 Length: 479 Insert Done parsing in 0 seconds
Type here to search
25°C Smoke
ENG
9:38 AM
3/1/2024

```c
#include<stdio.h>
struct address
{
    int pin;
    char phone[14];
};
struct employee
{
    char name[20];
    struct address add;
};
int main()
{
    struct employee emp;
    printf("Enter employee information?\n");
    scanf("%s %d %s",emp.name,&emp.add.pin,emp.add.phone);
    printf("Printing the employee information....\n");
    printf("name: %s \n Pincode: %d \n Phone: %s",emp.name,emp.add.pin,emp.add.phone);
}
```

```
Enter employee information?
Ankit
382630
8460467193
Printing the employee information....
name: Ankit
 Pincode: 382630
 Phone: 8460467193
--------------------------------
Process exited after 8.645 seconds with return value 0
Press any key to continue . . .
```

Functions in Structures in C

- ✓ In C, you can define functions inside a structure as member functions. These functions have access to the member variables of the structure and can be used to manipulate them.
- ✓ Structure functions in C make the code efficient. A code that consumes less memory and takes less time to execute is good.


D:\Programming-C AR\Unit-2\structure function.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
Array Within Structure in C.cpp
nested domain.cpp
structure function.cpp
D:\Programming-C AR\Unit-2\structure function.exe
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 1 Sel: 0 Lines: 26 Length: 522 Insert Done parsing in 0 seconds
Type here to search
25°C Smoke
ENG
9:47 AM
3/1/2024

```c
#include <stdio.h>
struct student
{
    char name[50];
    int per;
    int rno;
};
void display(int a, int b);
int main() {
    struct student s1;
    printf("Enter name: ");
    gets(s1.name);
    printf("Enter the roll number: ");
    scanf("%d",&s1.rno);
    printf("Enter percentage: ");
    scanf("%d", &s1.per);
    display(s1.rno,s1.per);
    return 0;
}
void display(int a, int b )
{
    printf("\n Displaying information \n");
    printf("Roll number: %d", a);
    printf("\nPercentage: %d", b);
}
```

```
Enter name: Ankit
Enter the roll number: 141
Enter percentage: 89

 Displaying information
Roll number: 141
Percentage: 89
--------------------------------
Process exited after 9.816 seconds with return value 0
Press any key to continue . . .
```

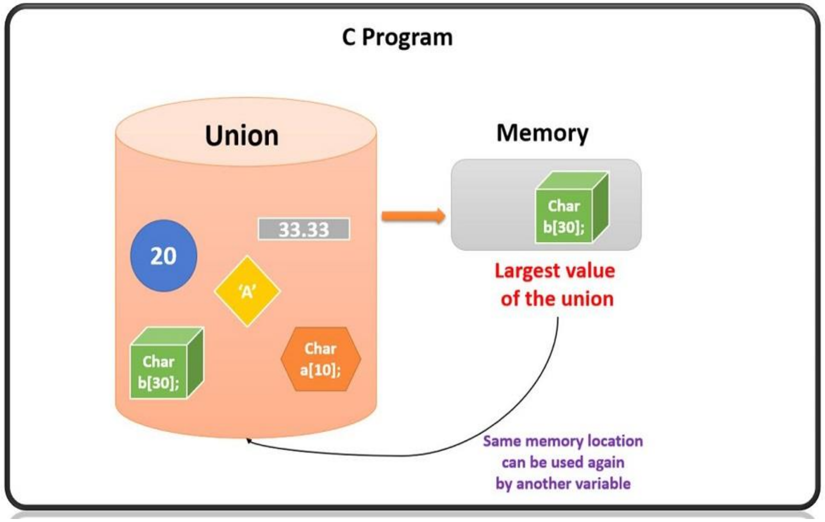
Reference Link for Structure in C

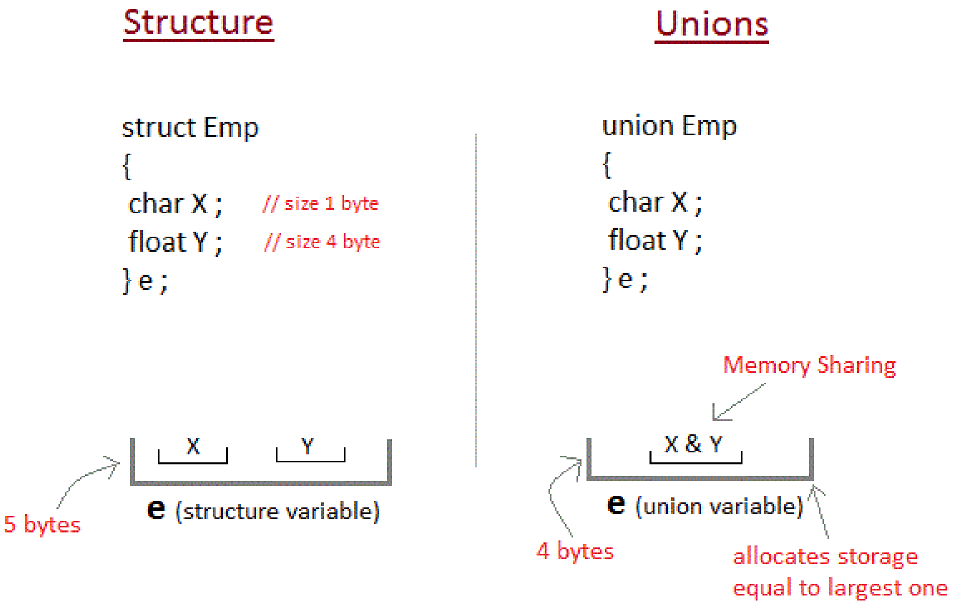
- *https://www.scaler.com/topics/c/structures-c/*
- *https://www.geeksforgeeks.org/structures-c/*
- *https://www.w3schools.com/c/c_structs.php*
- *https://www.javatpoint.com/structure-in-c*
- *https://www.scaler.com/topics/c/structure-and-functions-in-c/*
- *https://www.javatpoint.com/nested-structure-in-c*

PROGRAMMING

Introduction for Union in C

- ✓ **“Union is a data type in C programming that allows different data types to be stored in the same memory locations”**
- ✓ Union provides an efficient way of reusing the memory location, as only one of its members can be accessed at a time.
- ✓ A union is used almost in the same way you would declare and use a structure.


C Program
Union
20
33.33
'A'
Char b[30];
Char a[10];
Memory
Char b[30];
Largest value of the union
Same memory location can be used again by another variable



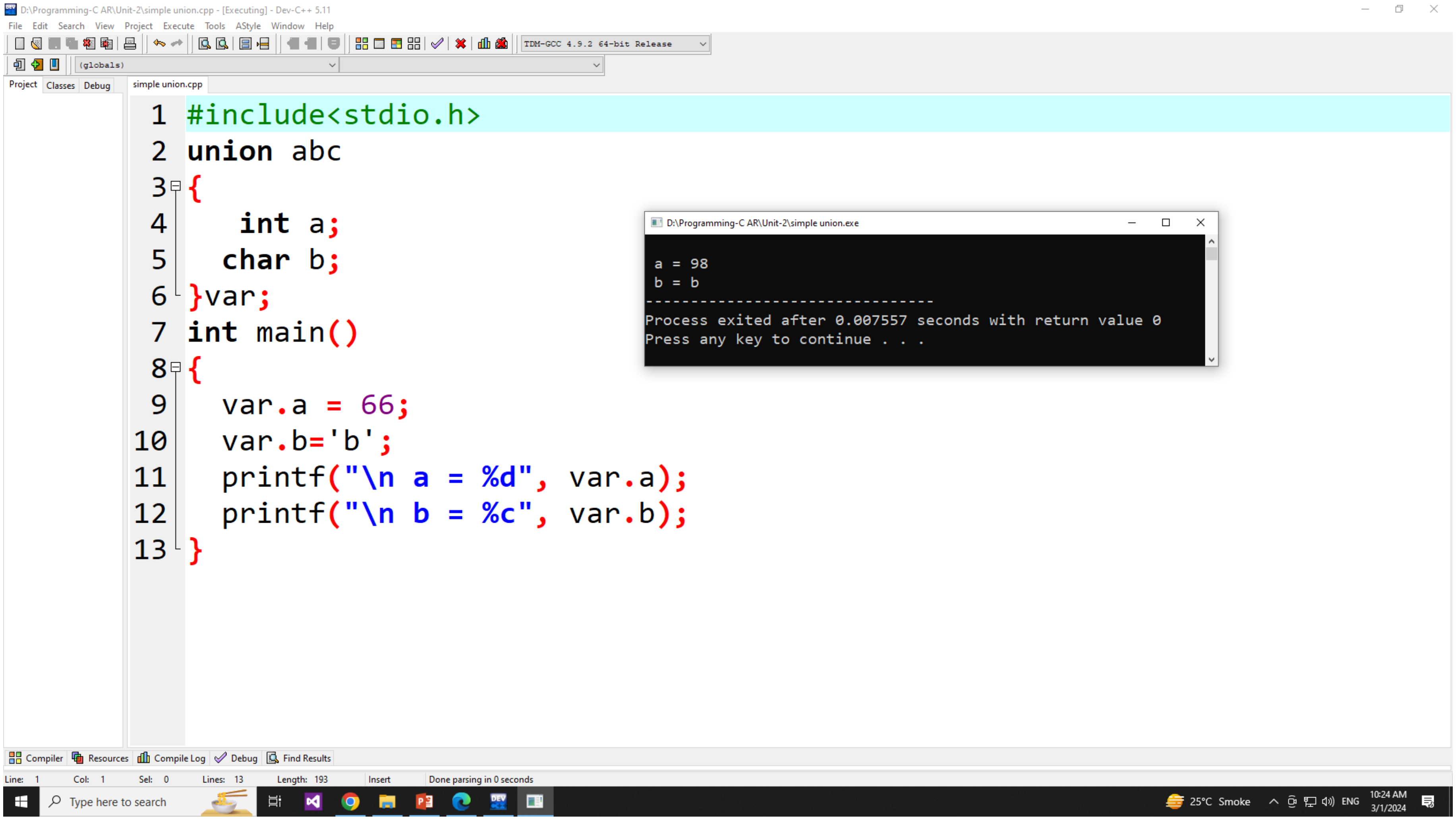
Structure

```
struct Emp
{
 char X ;    // size 1 byte
 float Y ;   // size 4 byte
} e ;
```

Unions

```
union Emp
{
 char X ;
 float Y ;
} e ;
```

Simple Program for Union in C


D:\Programming-C AR\Unit-2\simple union.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
simple union.cpp
D:\Programming-C AR\Unit-2\simple union.exe
Compiler Resources Compile Log Debug Find Results
Type here to search
25°C Smoke
ENG
10:24 AM
3/1/2024

```c
#include<stdio.h>
union abc
{
    int a;
   char b;
}var;
int main()
{
   var.a = 66;
   var.b='b';
   printf("\n a = %d", var.a);
   printf("\n b = %c", var.b);
}
```

```

 a = 98
 b = b
--------------------------------
Process exited after 0.007557 seconds with return value 0
Press any key to continue . . .
```

Line: 1 Col: 1 Sel: 0 Lines: 13 Length: 193 Insert Done parsing in 0 seconds

Declaration and Initialization of Union

- **C Union Declaration**

✓ A union is defined using the union keyword. For instance:

```
union circle
{
    char name[30];
    int radius;
} circle1, circle2;
```

✓ The declaration above includes a union named "circle" with two members, name and radius. At the same time, two union variables circle1 and circle2 are created.

Declaration and Initialization of Union

- **Define a Union Variable**

✓ Defining Union Variable with Declaration

```
union student {
    char name[50];
    int roll_no;
} stud1, stud2;
```

✓ Defining Union Variable after Declaration

```
union student {
    char name[50];
    int roll_no;
};
union student stud1, stud2;
```

Declaration and Initialization of Union

- **Accessing Union Members**

✓ To access the members of a union, you use the dot (.) operator. If you have a pointer to a union, then you'd use the arrow (->) operator.

```
strcpy(stud1.name, “Ankit Rami");
stud2.roll_no = 25;
```

- **Initialization of Union in C**

✓ When initializing a union, only the first member can be set using an initializer:

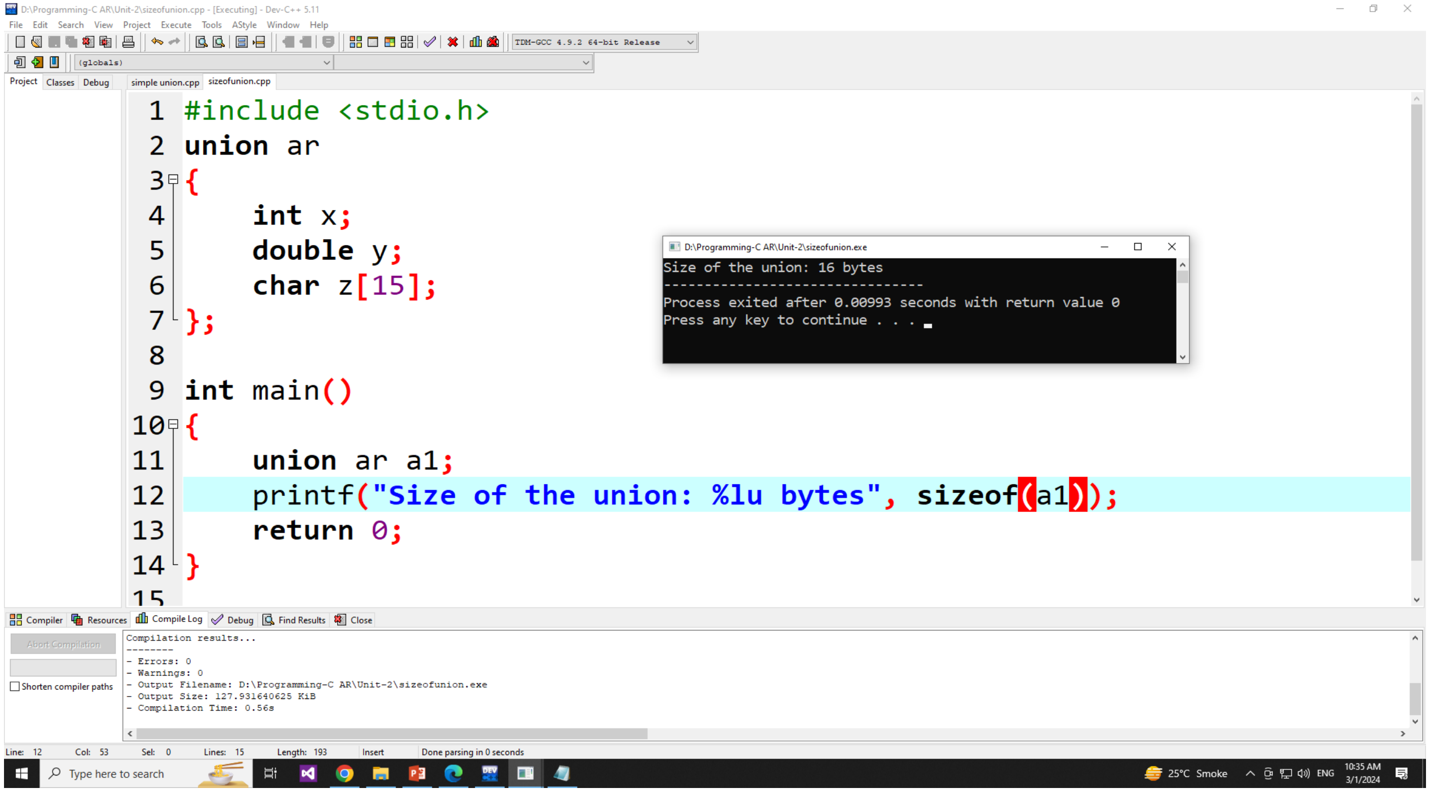
```
union student stud3 = {“Ankit"};
```

Reference Link - https://www.scaler.com/topics/c/unions-in-c/

Declaration and Initialization of Union

• Size of Union

✓ In C, the size of a union is determined by its largest member due to the fundamental nature of unions. Unions enable multiple data members to share the same memory location, unlike structures where each member has its distinct memory allocation. As a result, when you declare a union with members of varying data types, the compiler calculates the size of the union based on the largest member's size.


D:\Programming-C AR\Unit-2\sizeofunion.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
simple union.cpp
sizeofunion.cpp
D:\Programming-C AR\Unit-2\sizeofunion.exe
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Shorten compiler paths
Line: 12 Col: 53 Sel: 0 Lines: 15 Length: 193 Insert Done parsing in 0 seconds
Type here to search
25°C Smoke ENG 10:35 AM 3/1/2024

```c
#include <stdio.h>
union ar
{
	int x;
	double y;
	char z[15];
};

int main()
{
	union ar a1;
	printf("Size of the union: %lu bytes", sizeof(a1));
	return 0;
}
```

```
Size of the union: 16 bytes
--------------------------------
Process exited after 0.00993 seconds with return value 0
Press any key to continue . . .
```

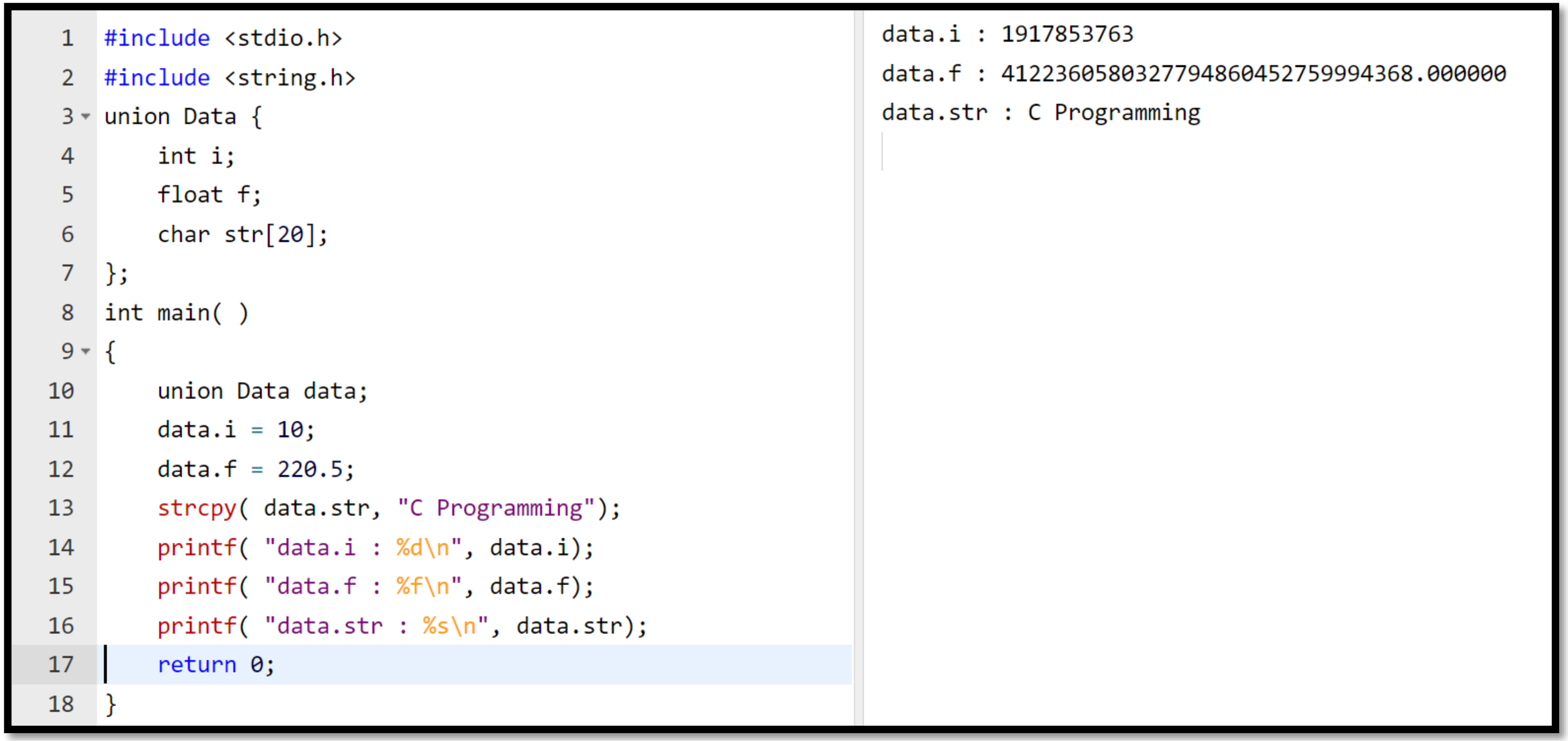
```
Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: D:\Programming-C AR\Unit-2\sizeofunion.exe
- Output Size: 127.931640625 KiB
- Compilation Time: 0.56s
```

Comparison of Structures and Union

Structure	Union
Sum of sizes of all members store memory	Size of the largest member store in memory
All members can be accessed simultaneously	Only one member can be accessed at a time
All or specific members can be initialized	Only the first member can be initialized
struct keyword used to define a Structure	union keyword used to define a Union
Structure occupies higher memory space	Union occupies lower memory space

Ex- struct Person {
char name[50];
int citNo;
float salary;
} person1;
	Ex- union Person {
	char name[50];
	int citNo;
	float salary;
	} person1;

Simple Example of Union

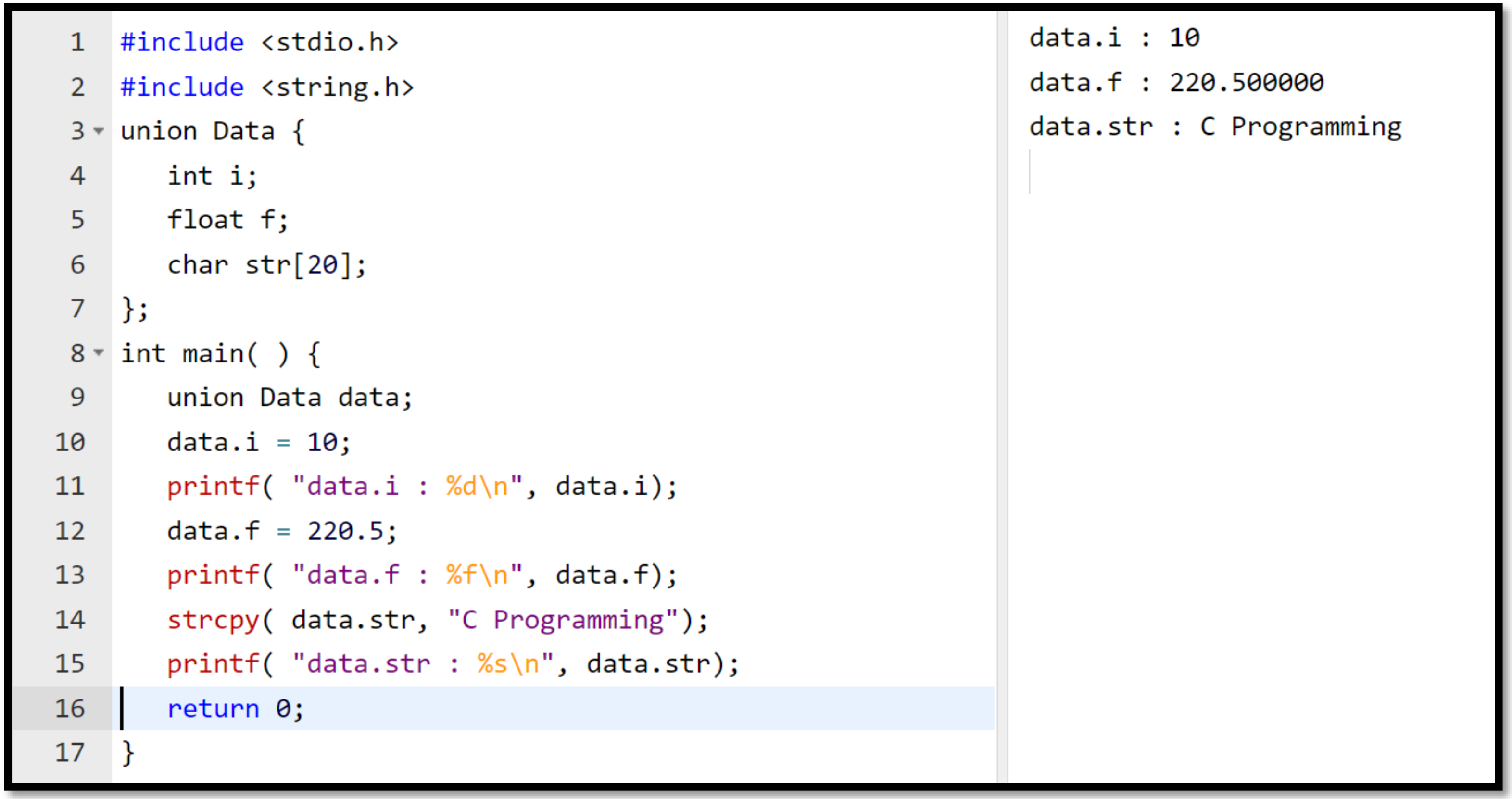


```c
#include <stdio.h>
#include <string.h>
union Data {
    int i;
    float f;
    char str[20];
};
int main( )
{
    union Data data;
    data.i = 10;
    data.f = 220.5;
    strcpy( data.str, "C Programming");
    printf( "data.i : %d\n", data.i);
    printf( "data.f : %f\n", data.f);
    printf( "data.str : %s\n", data.str);
    return 0;
}
```

```
data.i : 1917853763
data.f : 4122360580327794860452759994368.000000
data.str : C Programming
```

- Here, we can see that the values of **i** and **f** members of union got corrupted because the final value assigned to the variable has occupied the memory location and this is the reason that the value of **str** member is getting printed very well.

Simple Example of Union



```c
#include <stdio.h>
#include <string.h>
union Data {
    int i;
    float f;
    char str[20];
};
int main( ) {
    union Data data;
    data.i = 10;
    printf( "data.i : %d\n", data.i);
    data.f = 220.5;
    printf( "data.f : %f\n", data.f);
    strcpy( data.str, "C Programming");
    printf( "data.str : %s\n", data.str);
    return 0;
}
```

```
data.i : 10
data.f : 220.500000
data.str : C Programming
```

- Here, all the members are getting printed very well because one member is being used at a time.

Any Query Contact Us

Faculty Name- Ankit Rami

Email – ankitramiblog@gmail.com

Contact No – +91 8460467193

Website - amit.arinfoway.com

Like Comment Share

Subscribe

Subscribe Our YouTube Channel

https://www.youtube.com/channel/UCWbJh2iQ8w-8nrU0Xpjpw7g