Structure

Structured Programming Language (CSE-1271)

Course Instructor : Mohammed Mamun Hossain
Assistant Professor, Dept. of CSE, BAUST

Outline

1. Structure
2. Union
3. Enumeration

Structure

- A collection of one or more different variables with the same handle (same name).

```
struct point
{
      char name[30];
      int x;
      int y;
      double temperature;
};
```

```
struct point pt;
```

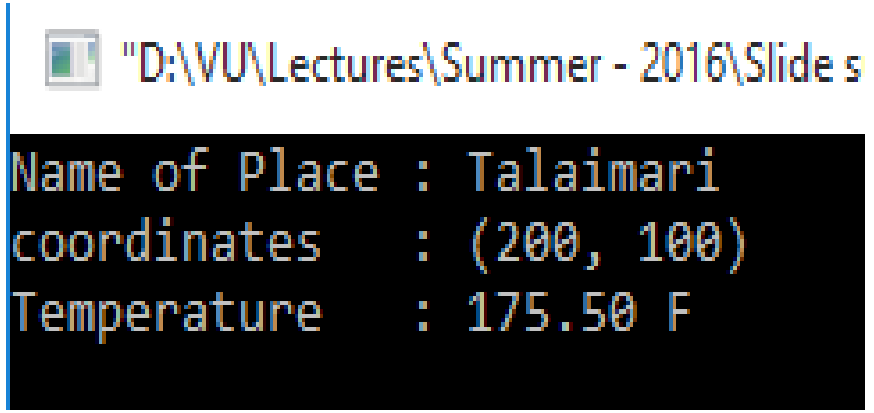
Structure

- Access an element: `structure-name.member`
- Example: `pt.x, pt.y`

```
struct point
{
    char name[30];
    int x;
    int y;
    double temperature;
};
struct point pt = {"Talaimari", 200, 100, 175.50};

int main()
{
    printf("Name of Place : %s\n", pt.name);
    printf("coordinates   : (%d, %d)\n", pt.x,pt.y);
    printf("Temperature   : %.2lf F\n", pt.temperature);

    return 0;
}
```



Structure

- Structs can also contain other structs:

```
struct rectangle {
      struct point ul;
      struct point lr;
};

struct rectangle rect;
```

- To access:

```
rect.ul.x;
```

Structure | Arrays

Is structure array?

NO

- In array each element is of the same type.
- Each member of a structure can have its own data type, which may differ from the types of the other members.

But,

- Array of Structures act like any other array.

```
struct point pt[3];
```

```
pt[1].name="A";
pt[1].x = 0;
pt[1].y = 1;
```

```
pt[2].name="B";
pt[2].x = 4;
pt[2].y = 1;
```

```
pt[0].name = "mid";
pt[0].x = (pt[2].x + pt[1].x)/2;
pt[0].y = (pt[2].y + pt[1].y)/2;
```

Structure | Pointers

- Pointers are an easier way to manipulate structure members by reference
- The entire structure is not passed by value, only the address of the first member
- Use arrow operator for accessing the struct element

```
struct point MyPoint, *PointPtr;
PointPtr = &MyPoint;
PointPtr->x = 250;
PointPtr->y = 50;
```

Unions

- A union is a special data type available in C that allows to store different data types in the same memory location. You can define a union with many members, but only one member can contain a value at any given time.
- A union is a memory location that is shared by two or more different types of variables.

```
union u_tag
{
      int ival;
      float fval;
      char cval;
} u;
```

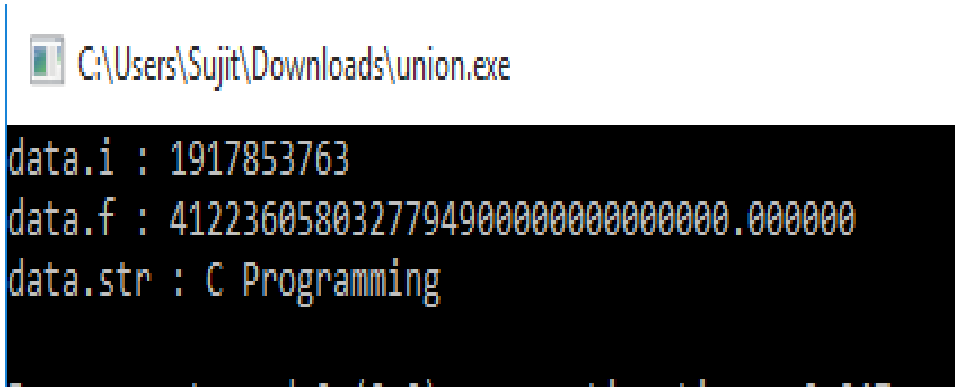
- Each of ival, fval, cval have the same location in memory.
- Usage is similar to that of structs:

```
u.ival or u.cval
```

Unions

Example:

```
4   union Data
5   {
6       int i;
7       float f;
8       char str[20];
9   };
10
11  int main( )
12  {
13      union Data data;
14
15      data.i = 10;
16      data.f = 220.5;
17      strcpy( data.str, "C Programming");
18
19      printf( "data.i : %d\n", data.i);
20      printf( "data.f : %f\n", data.f);
21      printf( "data.str : %s\n", data.str);
22
23      return 0;
24  }
```



Enumeration

- enum is another user-defined type consisting of a set of named constants called enumerators.
- Using a keyword enum, it is a set of integer constants represented by identifiers.

- General format:

```
enum [tag]
{
enum-list
}
[declarator];
```

Enumeration

- The values in an enum start with 0, unless specified otherwise, and are incremented by 1. For example, the following enumeration,

```
enum days {Mon, Tue, Wed, Thu, Fri, Sat, Sun};
```

- Creates a new data type, enum days, in which the identifiers are set automatically to the integers 0 to 6.
- To number the days 1 to 7, use the following enumeration,

```
enum days {Mon = 1, Tue, Wed, Thu, Fri, Sat, Sun};
```

- Or we can re-arrange the order,

```
enum days {Mon, Tue, Wed, Thu = 7, Fri, Sat, Sun};
```

Enumeration

- Example:

ere | structures.c | enum.c

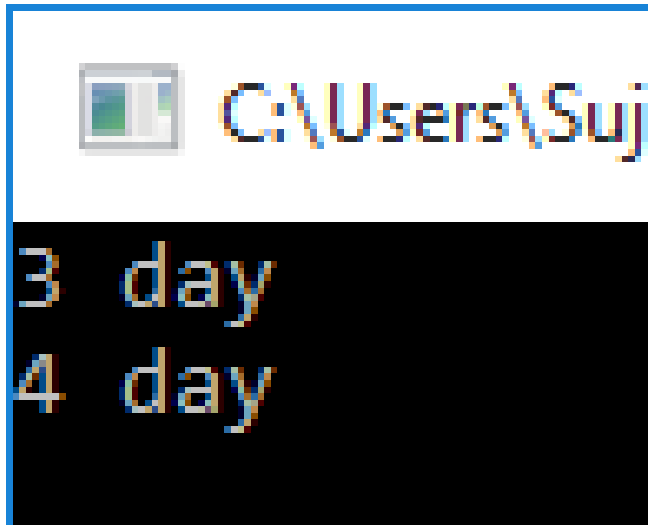
```c
#include <stdio.h>

enum week { sunday, monday, tuesday, wednesday, thursday, friday, saturday};
enum week1 { sun = 1, mon, tues, wednes, thurs, fri, satur};

int main()
{
    enum week today;
    today=wednesday;
    printf("%d day\n",today);

    enum week1 today1;
    today1=wednes;
    printf("%d day\n",today1);

    return 0;
}
```



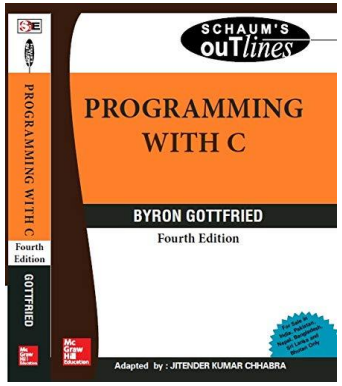
Thank You.

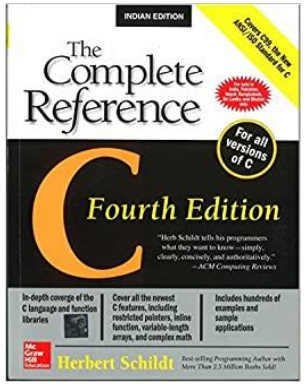
Questions and Answer

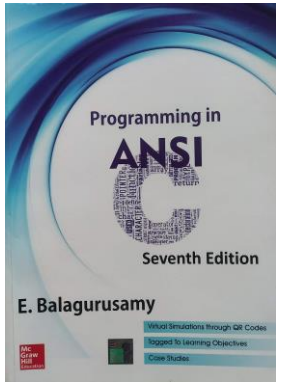
References

Books:

1. Programming With C. *By Byron Gottfried*
2. The Complete Reference C. *By Herbert Shield*
3. Programming in ANSI C *By E. Balagurusamy*
4. Teach yourself C. *By Herbert Shield*







Web:

1. www.wikbooks.org

and other slide, books and web search.