Structured Programming Language
(**0613-1107**)

Structures

Basic Idea

- To represent one item, we can declare one variable
- To represent several items of the same type, we can declare an array
- To represent several items of different types, we can declare a structure.

Example: Represent Your Birthday

- A birthday consists of 3 parts: year, month and day.
- They are all integers.

```
int year = 2007;
int month = 11;
int day =13;
```

- These 3 variables are logically related
- It would be better if you could group them together.
- The structure in C allows you to do so.

Structure

```
// define a structure called date
struct date {
    int year;                // 1st member variable
    int month;               // 2nd member variable
    int day;                 // 3rd  member variable
};     // don't forget the semicolon !
```

- This newly defined structure "date" is a new data type which contains 3 integer members

What is a Structure?

- A Structure is a collection of related data items, possibly of different types.
- A structure type in C is called struct.
- A struct can be composed of data of different types.
- In contrast, array can contain only data of the same type.

Examples

- Structures hold data that belong **together**.
- Examples:
  - Student:
    - student id, name, major, gender, start year, …
  - Bank account:
    - account number, name, currency, balance, …
  - Address book:
    - name, address, telephone number, …
- In database applications, structures are called records.

Structure Declaration 1

```
// define the type
struct PersonalData  {
   char name[namesize];
   char address[addresssize];
   int YearOfBirth;
   int MonthOfBirth;
   int DayOfBirth;
   };
struct PersonalData x;   // declare the variable
```

Structure Declaration 2

```
struct PersonalData  {
   char name[namesize];
   char address[addresssize];
   int YearOfBirth;
   int MonthOfBirth;
   int DayOfBirth;
   }  x;          /* variable identifier follows type */
```

Structure Declaration 3

```
typedef struct {
   char name[namesize];
   char address[addresssize];
   int YearOfBirth;
   int MonthOfBirth;
   int DayOfBirth;
   }  PersonalData;

PersonalData x;
```

Structure

Given the structure date defined in previous slides, we can declare variables:

```
struct date birthday;
struct date purchaseDate;
```

- birthday and purchaseDate are two variables of type struct date
- Each has 3 members: day, month, year

Accessing the Members of a Structure

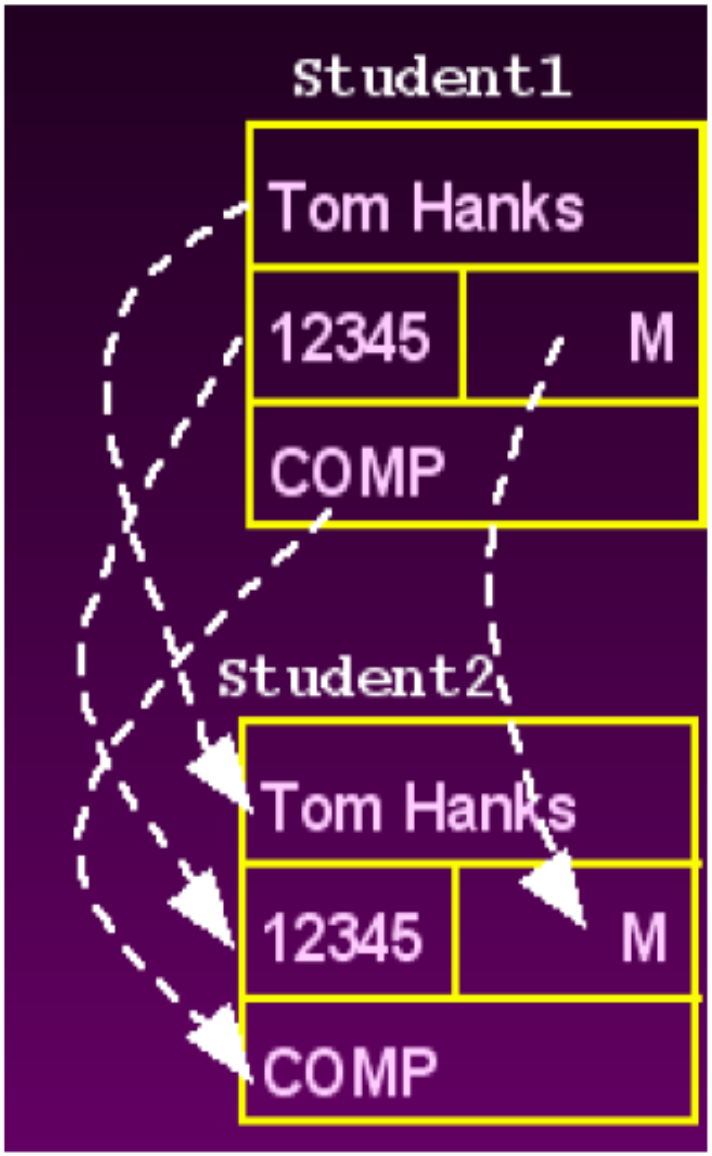
A member of a structure is accessed by specifying the variable name, followed by a period, and then the member name

```
struct date today;


today.month = 1;  // assigning values to members
today.day = 1;
if(today.month == 1 && today.day == 1)
    printf("Happy new year!");
```

Struct-to-Struct Assignment

```
// The values contained in one
// struct type variable can be
// assigned to another variable
// of the same struct type.
struct StudentRecord{
    char Name[15];
    int Id;
    char Dept[5];
    char Gender;
};
StudentRecord Student1, Student2;
strcpy(Student1.Name, "Tom Hanks");
Student1.Id = 12345;
strcpy(Student1.Dept, "COMP");
Student1.gender = 'M';
Student2 = Student1;
```



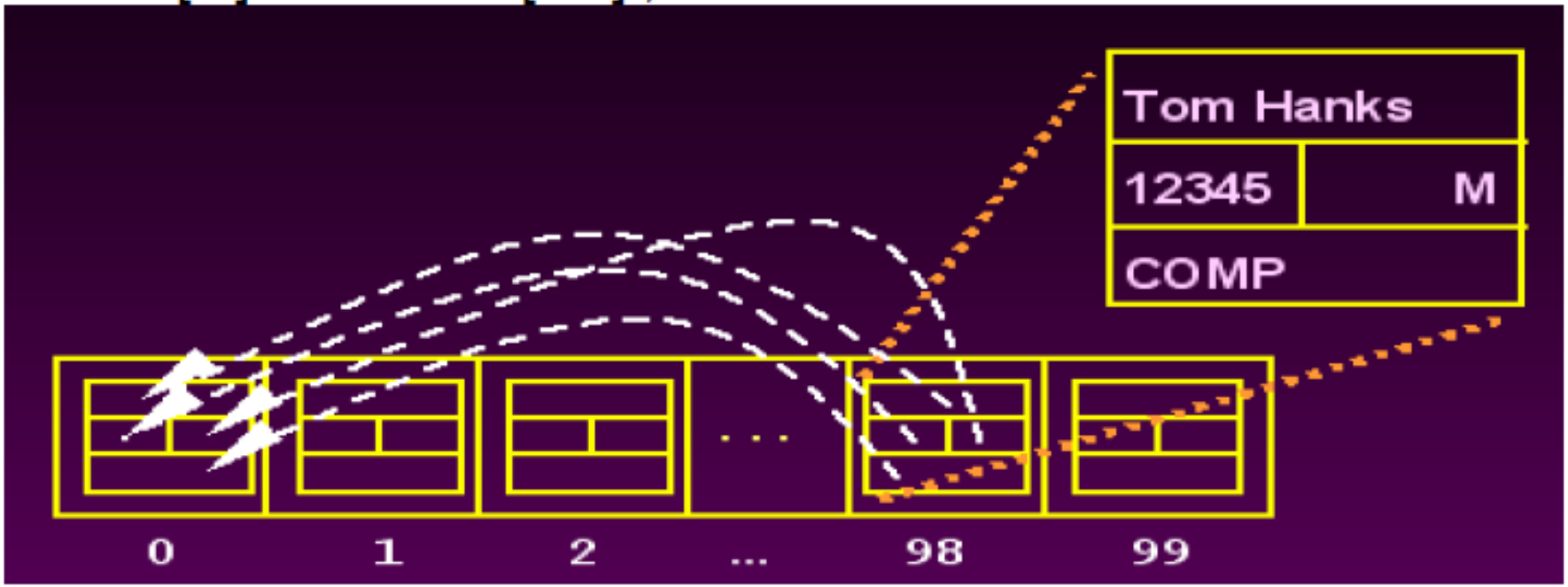
Function that Returns a Structure

```
struct date dateUpdate(struct date d) {
    struct date tomorrow;
    // code to calculate the date of tomorrow
    ......
    return tomorrow;
}
```

Arrays of Structures

- Example:

```
StudentRecord Class[100];
strcpy(Class[98].Name, "Tom Hanks");
Class[98].Id = 12345;
strcpy(Class[98].Dept, "COMP");
Class[98].gender = 'M';
Class[0] = Class[98];
```



- Define a structure type, **struct personal** that would contain person name, date of joining and salary. Using this structure, write a program to read this information for one person from the keyboard and print the same on the screen.

DEFINING AND ASSIGNING VALUES TO STRUCTURE MEMBERS

```
struct  personal
{
   char  name[20];
   int   day;
   char  month[10];
   int   year;
   float salary;
};
main()
{
   struct personal person;
   printf("Input Values\n");
   scanf("%s %d %s %d %f",
          person.name,
         &person.day,
          person.month,
         &person.year,
         &person.salary);
   printf("%s %d %s %d %f\n",
          person.name,
          person.day,
          person.month,
          person.year,
          person.salary);
}
```

Output

```
Input Values
M.L.Goel 10 January 1945 4500
M.L.Goel 10 January 1945 4500.00
```

Example

Write a program to illustrate the comparison of structure variables.

Program

```
struct class
{
    int  number;
    char name[20];
    float marks;
};
main()
{
    int  x;
    struct class student1 = {111,"Rao",72.50};
    struct class student2 = {222,"Reddy", 67.00};
    struct class student3;

    student3 = student2;

    x = ((student3.number ==  student2.number) &&
        (student3.marks  ==  student2.marks)) ? 1 : 0;

    if(x == 1)
    {
            printf("\nstudent2 and student3 are
    same\n\n");
       printf("%d %s %f\n", student3.number,
                        student3.name,
                        student3.marks);
    }
    else
       printf("\nstudent2 and student3 are
    different\n\n");
```

Output

student2 and student3 are same

222 Reddy 67.000000

write a program to calculate the subject-wise and student-wise totals and store them as a part of the structure.

```
struct marks
  {
     int  sub1;
     int  sub2;
     int  sub3;
     int  total;
  };
  main()
  {
     int  i;
     struct marks student[3] =  {{45,67,81,0},
                                 {75,53,69,0},
                                 {57,36,71,0}};
     struct marks total;
     for(i = 0; i <= 2; i++)
     {
        student[i].total = student[i].sub1 +
                           student[i].sub2 +
                           student[i].sub3;
        total.sub1 = total.sub1 + student[i].sub1;
        total.sub2 = total.sub2 + student[i].sub2;
        total.sub3 = total.sub3 + student[i].sub3;
        total.total = total.total + student[i].total;
     }
```

```
printf(" STUDENT          TOTAL\n\n");
      for(i = 0; i <= 2; i++)
printf("Student[%d]  %d\n", i+1,student[i].total);

printf("\n SUBJECT          TOTAL\n\n");
printf("%s       %d\n%s       %d\n%s       %d\n",
           "Subject 1   ", total.sub1,
           "Subject 2   ", total.sub2,
           "Subject 3   ", total.sub3);

     printf("\nGrand Total = %d\n", total.total);
  }
```

Output

```
STUDENT          TOTAL
Student[1]       193
Student[2]       197
Student[3]       164

SUBJECT          TOTAL
Subject 1        177
Subject 2        156
Subject 3        221

Grand Total  = 554
```

Functions in Structures in C

```c
struct fruit {
    char name[50];
    char color[50];
};
// function prototype
void display(struct fruit s);

int main() {
    struct fruit s1;

    printf("Enter name: ");
    // read string input from the user until \n is entered
    // \n is discarded
    scanf("%[^\n]%*c", s1.name);

    printf("Enter color: ");
    scanf("%[^\n]%*c", s1.color);

    display(s1); // passing struct as an argument
    return 0;
}
void display(struct fruit s) {
    printf("\nDisplaying information\n");
    printf("Name: %s", s.name);
    printf("\nColor: %s", s.color);
}

```


THANK
YOU