



**NANYANG
TECHNOLOGICAL
UNIVERSITY**

CE1007/ CZ1007 DATA STRUCTURES

Lesson 9.3 Nested Structures

Assoc Prof Hui Siu Cheung

College of Engineering
School of Computer Science and Engineering

OVERVIEW

The following are the coverage for Structures:

- Structure Declaration, Initialisation and Operations
- Arrays of Structures
- **Nested Structures**
- Pointers to Structures
- Functions and Structures
- The typedef Construct

Content Copyright Nanyang Technological University 2

The following are the coverage for Structures: this video focusses on Nested structures

LEARNING OBJECTIVES

At this lesson, you should be able to:

Content Copyright Nanyang Technological University 3

The slide features a red header with the title 'LEARNING OBJECTIVES'. Below the header is a large, light gray rectangular box with rounded corners, intended for the learning objectives. The text 'At this lesson, you should be able to:' is positioned at the top left of this box. The slide also includes a footer with the text 'Content Copyright Nanyang Technological University' and the page number '3'.

LEARNING OBJECTIVES: At this lesson, you should be able to:

LEARNING OBJECTIVES

At this lesson, you should be able to:

- Write a program using nested structures

Content Copyright Nanyang Technological University 4

Write a program using Nested structures

NESTED STRUCTURES

- A structure can also be **included** in other structures.

Content Copyright Nanyang Technological University 5

Nested Structures

A structure can also be included in other structures.

NESTED STRUCTURES

- A structure can also be **included** in other structures.
- For example, to keep track of the course history of a student, one can use a structure (**without** any nested structures) such as:

Content Copyright Nanyang Technological University 6

For example, to keep track of the course history of a student, one can use a structure (without any nested structures) as follows:

NESTED STRUCTURES

- A structure can also be **included** in other structures.
- For example, to keep track of the course history of a student, one can use a structure (**without** any nested structures) such as:

```

struct studentTag { // without any nested structures
    char name[40];
    char id[20];
    char tel[20];
    int  SC101Yr; /* the year when SC101 is taken */
    int  SC101Sr; /* the semester when SC101 is taken */
    char SC101Grade; /* the grade obtained for SC101 */
    int  SC102Yr; /* the year when SC102 is taken */
    int  SC102Sr; /* the semester when SC102 is taken */
    char SC102Grade; /* the grade obtained for SC102 */
};
struct studentTag student[1000];
  
```

(1) Student information

(2) Course: SC101

(3) Course: SC102

// student – a complete database of student records

Content Copyright Nanyang Technological University

In the structure template definition **struct studentTag**, the members are the student information including **name**, **identity** and **telephone**

NESTED STRUCTURES

- A structure can also be **included** in other structures.
- For example, to keep track of the course history of a student, one can use a structure (**without** any nested structures) such as:

```

struct studentTag { // without any nested structures
    char name[40];
    char id[20];
    char tel[20];
    int  SC101Yr; /* the year when SC101 is taken */
    int  SC101Sr; /* the semester when SC101 is taken */
    char SC101Grade; /* the grade obtained for SC101 */
    int  SC102Yr; /* the year when SC102 is taken */
    int  SC102Sr; /* the semester when SC102 is taken */
    char SC102Grade; /* the grade obtained for SC102 */
};
struct studentTag student[1000];
  
```

(1) Student information

(2) Course: SC101

(3) Course: SC102

// student – a complete database of student records

Content Copyright Nanyang Technological University

In addition, it also includes the courses that are taken by the student.

NESTED STRUCTURES

- A structure can also be **included** in other structures.
- For example, to keep track of the course history of a student, one can use a structure (**without** any nested structures) such as:

```

struct studentTag { // without any nested structures
    char name[40];
    char id[20];
    char tel[20];
    int  SC101Yr; /* the year when SC101 is taken */
    int  SC101Sr; /* the semester when SC101 is taken */
    char SC101Grade; /* the grade obtained for SC101 */
    int  SC102Yr; /* the year when SC102 is taken */
    int  SC102Sr; /* the semester when SC102 is taken */
    char SC102Grade; /* the grade obtained for SC102 */
};
struct studentTag student[1000];

```

(1) Student information

(2) Course: SC101

(3) Course: SC102

// student – a complete database of student records

Content Copyright Nanyang Technological University 9

Once the **studentTag** is defined, an array variable **student** of 1000 elements is created.

NESTED STRUCTURES (CONT'D)

- Alternatively, **student** can be defined in a more elegant manner (using **nested structures**) as:

```
struct personTag {  
    char name[40];  
    char id[20];  
    char tel[20];  
};
```

Content Copyright Nanyang Technological University

10

Nested Structures

Alternatively, the variable **student** can be defined in a more elegant manner using nested structures.

NESTED STRUCTURES (CONT'D)

```
struct personTag {  
    char name[40];  
    char id[20];  
    char tel[20];  
};
```

```
struct courseTag {  
    int year, semester;  
    char grade;  
};
```

Content Copyright Nanyang Technological University 11

Nested Structures

We also create a structure template called **courseTag** to contain the course information as follows:

The structure **courseTag** has three members, namely **year** and **semester** of type **int**, and **grade** of type **char**.

NESTED STRUCTURES (CONT'D)

```

struct personTag {
    char name[40];
    char id[20];
    char tel[20];
};

struct courseTag {
    int year, semester;
    char grade;
};

struct studentTag {
    struct personTag      studentInfo;
    struct courseTag     SC101,SC102;
};
  
```

Nested structure

Content Copyright Nanyang Technological University 12

We define the nested structure **studentTag** as shown here.

Note that the structure definition of **personTag** and **courseTag** must appear before the definition of structure **studentTag**.

NESTED STRUCTURES (CONT'D)

```

struct personTag {
    char name[40];
    char id[20];
    char tel[20];
};

struct courseTag {
    int year, semester;
    char grade;
};

struct studentTag {
    struct personTag      studentInfo;
    struct courseTag     SC101,SC102;
};

struct studentTag student[1000]; // student – complete database
  
```

Nested structure

Content Copyright Nanyang Technological University

13

Finally, we define the variable **student** as shown here.

NESTED STRUCTURES: INITIALISATION

```
/* Array variable initialisation */
struct studentTag student[1000] = {
    { "John", "CE000011", "123-4567",
      { 2002, 1, 'B' },
      { 2002, 1, 'A' } },
    { "Mary", "CE000022", "234-5678",
      { 2002, 1, 'C' },
      { 2002, 1, 'A' } },
    { "Peter", "CE000033", "345-6789",
      { 2002, 1, 'B' },
      { 2002, 1, 'A' } }
    ...
};
```

Content Copyright Nanyang Technological University14

Nested Structures: Initialization

In this program, after defining the nested structure **studentTag** and the array of structures variable **student**,

NESTED STRUCTURES: INITIALISATION

```
/* Array variable initialisation */
struct studentTag student[1000] = {
    { "John", "CE000011", "123-4567"},
    { 2002, 1, 'B' },
    { 2002, 1, 'A' } },
    { "Mary", "CE000022", "234-5678"},
    { 2002, 1, 'C' },
    { 2002, 1, 'A' } },
    { "Peter", "CE000033", "345-6789"},
    { 2002, 1, 'B' },
    { 2002, 1, 'A' } }
    ...
};
```

Content Copyright Nanyang Technological University

15

we initialize the variable **student** with initial data.

NESTED STRUCTURES: OPERATION

```

/* To print individual elements of the array*/
int i;
for (i=0; i<=2; i++) {
    printf("Name:%s, ID: %s, Tel: %s\n",
           student[i].studentInfo.name,
           student[i].studentInfo.id,
           student[i].studentInfo.tel);

    printf("SC101 in year %d semester %d : %c\n",
           student[i].SC101.year,
           student[i].SC101.semester,
           student[i].SC101.grade);
    printf("SC102 in year %d semester %d : %c\n",
           student[i].SC102.year,
           student[i].SC102.semester,
           student[i].SC102.grade);
}

```

- Using dot (member operator) to access members of structures

Content Copyright Nanyang Technological University 16

Nested Structures: Example

To access each array element, we use a **for** loop to traverse the array.

NESTED STRUCTURES: OPERATION

```

/* To print individual elements of the array*/
int i;
for (i=0; i<=2; i++) {
    printf("Name:%s, ID: %s, Tel: %s\n",
        student[i].studentInfo.name,
        student[i].studentInfo.id,
        student[i].studentInfo.tel);

    printf("SC101 in year %d semester %d : %c\n",
        student[i].SC101.year,
        student[i].SC101.semester,
        student[i].SC101.grade);

    printf("SC102 in year %d semester %d : %c\n",
        student[i].SC102.year,
        student[i].SC102.semester,
        student[i].SC102.grade);
}

```

- Using dot (member access operator) to access members of structures

Content Copyright Nanyang Technological University 17

The array notation and member operator are used for accessing each array element and structure member

NESTED STRUCTURES: OPERATION

```
/* To print individual elements of the array*/
int i;
for (i=0; i<=2; i++) {
    printf("Name:%s, ID: %s, Tel: %s\n",
        student[i].studentInfo.name,
        student[i].studentInfo.id,
        student[i].studentInfo.tel);

    printf("SC101 in year %d semester %d : %c\n",
        student[i].SC101.year,
        student[i].SC101.semester,
        student[i].SC101.grade);
    printf("SC102 in year %d semester %d : %c\n",
        student[i].SC102.year,
        student[i].SC102.semester,
        student[i].SC102.grade);
}
```

- Using dot (member access operator) to access members of structures

Content Copyright Nanyang Technological University

18

The data can then be processed and printed on the screen.

NESTED STRUCTURES: NOTATIONS

- **student[i]** denotes the $i+1^{th}$ array record. It consists of three members: studentInfo, SC101, SC102.

Content Copyright Nanyang Technological University 19

Nested Structures: Notations

This statement denotes the *i plus oneth* array record.

NESTED STRUCTURES: NOTATIONS

- **student[i]** denotes the $i+1^{th}$ array record. It consists of three members: studentInfo, SC101, SC102.
- **student[i].studentInfo** denotes the personal information in the $i+1^{th}$ record. It consists of three members: name, id, tel.

Content Copyright Nanyang Technological University 20

- This statement denotes the personal information in the $i+1^{th}$ record; It consists of three members: name, id, tel.

NESTED STRUCTURES: NOTATIONS

- **student[i]** denotes the $i+1^{th}$ array record. It consists of three members: studentInfo, SC101, SC102.
- **student[i].studentInfo** denotes the personal information in the $i+1^{th}$ record. It consists of three members: name, id, tel.
- **student[i].studentInfo.name** denotes the student name in this record.

Content Copyright Nanyang Technological University 21

- **This statement** denotes the student name in this record

NESTED STRUCTURES: NOTATIONS

- **student[i]** denotes the $i+1^{th}$ array record. It consists of three members: studentInfo, SC101, SC102.
- **student[i].studentInfo** denotes the personal information in the $i+1^{th}$ record. It consists of three members: name, id, tel.
- **student[i].studentInfo.name** denotes the student name in this record.
- **student[i].studentInfo.name[j]** denotes a single character value.

Content Copyright Nanyang Technological University 22

Nested Structures: Notations

This statement denotes a single character value.

NESTED STRUCTURES: NOTATIONS

- **student[i]** denotes the $i+1^{th}$ array record. It consists of three members: studentInfo, SC101, SC102.
- **student[i].studentInfo** denotes the personal information in the $i+1^{th}$ record. It consists of three members: name, id, tel.
- **student[i].studentInfo.name** denotes the student name in this record.
- **student[i].studentInfo.name[j]** denotes a single character value.
- **student[i].SC101, student[i].SC102** denotes the course information in the $i+1^{th}$ record. Each consists of three members: year, semester, grade.

Content Copyright Nanyang Technological University

This statement denotes the course information in the $i+1$ record.

SUMMARY

At this lesson, you should be able to:

- Write a program using nested structures

Content Copyright Nanyang Technological University 24

After watching this video lesson, you should be able to do the listed.