344-111

โครงสร้างข้อมูล (STRUCTURES)

Outline

- Structure
  - แนวคิดของ struct
  - การประกาศ struct
  - การกำหนดค่าเริ่มต้นให้กับตัวแปร struct
  - การใช้งานตัวแปรที่มีโครงสร้าง struct
  - Nested structure
  - Self-referentail structures
- Union
  - Concept of union
  - Using union
- Union and struct
- Concept of enum

Question

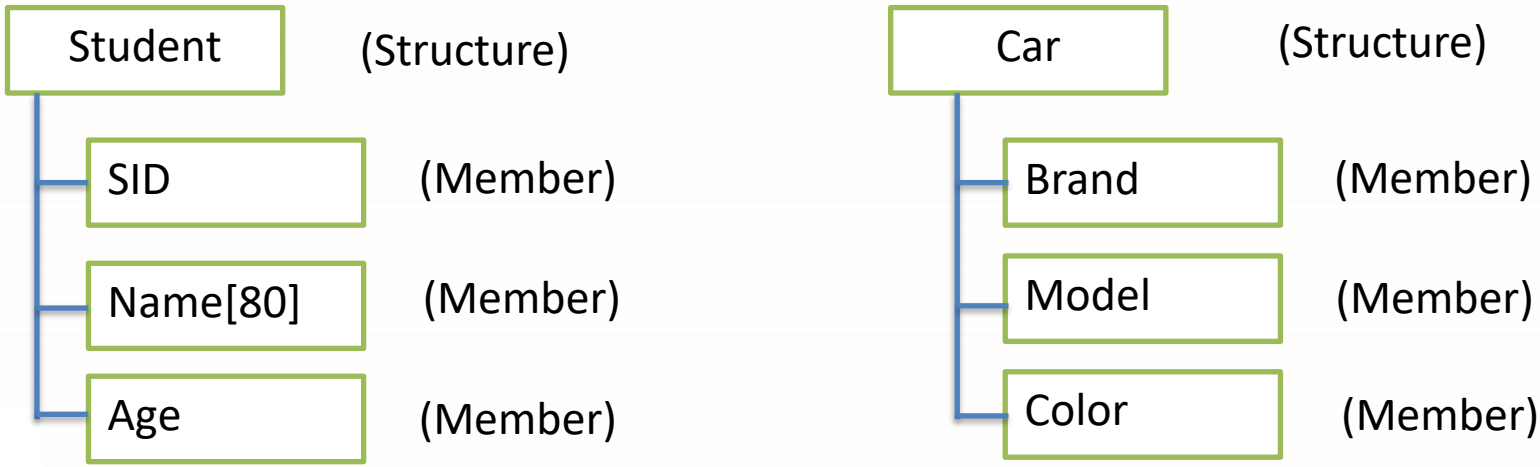
- ประกาศตัวแปรสำหรับเก็บข้อมูลของนักเรียนจำนวน 3 คน มีรายละเอียดดังนี้

	Name	Surname	Age
คนที่ 1	Amber	Grace	20
คนที่ 2	Amy	Freya	19
คนที่ 3	Ava	Holly	18

- Discuss with your friends

แนวคิด Struct

- ตัวแปรธรรมดาเก็บค่าได้ 1 ค่า
- ตัวแปรแบบ **Array** เก็บค่าได้หลายค่า **แต่**ต้องเป็นข้อมูลประเภทเดียวกัน
- **Struct** เป็นการการสร้างชนิดข้อมูลขึ้นใหม่ที่สามารถ
  - จัดกลุ่มข้อมูลให้รวมเป็นหนึ่งได้
  - ภายในโครงสร้างมีตัวแปรได้หลายประเภทและหลายตัว
- เช่น
  - ข้อมูลนักเรียน: รหัส ชื่อ-นามสกุล อายุ น้ำหนัก ความสูง
  - ข้อมูลรถยนต์: ยี่ห้อ รุ่น สี ทะเบียนรถ



Structure definition

```
struct ชื่อโครงสร้าง {
    คุณสมบัติ 1;
    คุณสมบัติ 2;
    ...
};
```

Example1

```
struct student{
    int id;
    char name[20];
    int age;
    float height;
    float weight;
};
```

แบบจำลองของโครงสร้าง

id	name[0]	...	name[19]	age	height	weight

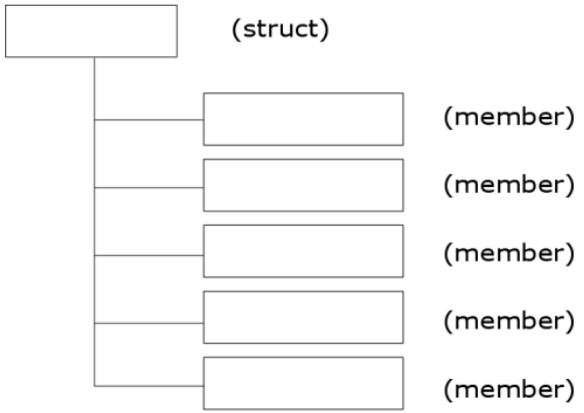
Example2

```
struct car{
    char brand[20];
    char model[20];
    char color[10];
};
```

Structure definition

```
struct employee {
    char firstName[20];
    char lastName[20];
    unsigned int age;
    char gender;
    double salary;
};
```

ลองเขียนแผนภาพจแสดงโครงสร้างที่ประกาศขึ้น



- นิยามกลุ่มข้อมูลที่สร้างใหม่ ชื่อว่า **employee**
- **ประกอบด้วยสมาชิก (Fields) ดังนี้**
  - ...

Structure definition

- จงนิยาม structure ที่มีชื่อว่า date เพื่อเก็บข้อมูลวันที่ โดยประกอบด้วยสมาชิก 3 ตัว ชื่อว่า d, m และ y ซึ่งเป็นจำนวนเต็ม
- จงนิยาม struct ชื่อว่า person ซึ่งมีสมาชิก 2 ตัวคือ name ใช้สำหรับเก็บข้อความที่ยาวไม่เกิน 50 ตัวอักษร และ age ซึ่งเป็นจำนวนเต็ม
- What is your structure?

Structure declaration

แบบที่1	**นิยามรูปแบบโครงสร้างหนึ่งขึ้นมาและตั้งชื่อว่า student พร้อมกับการประกาศตัวแปรสำหรับโครงสร้างนี้**
แบบที่ 2	นิยามรูปแบบโครงสร้างหนึ่งขึ้นมาและตั้งชื่อว่า student หลังจากนั้น ประกาศตัวแปรที่แยกออกจาก struct โดยกำหนดให้ตัวแปรมี type เป็น struct ที่สร้างขึ้น

```
struct student{
    int id;
    char name[20];
    int age;
    float height;
    float weight;
}student1, students[10],*studentptr;
```

```
struct student{
    int id;
    char name[20];
    int age;
    float height;
    float weight;
};

struct student student1, students[10],*studentptr;
```

กำหนดค่าเริ่มต้นให้กับตัวแปรแบบ struct

```
34  struct student student1, students[10],*studentptr;
35
36  struct student{
37      int id;
38      char name[20];
39      int age;
40      float height;
41      float weight;
42  };
43  struct student student1 = {1,"Wararat",36,163.0,50.0};
44  struct student *studentptr;
45  studentptr = &student1;
46
47  struct student students[] ={1,"Wararat",36,163.0,50.0,
48                              2,"Janya",36,153.0,45.0,
49                              3,"Niwan",37,160.0,48.0};
```

บรรทัดที่ 43 ประกาศตัวแปรชื่อ **student1** มี **type** คือ **student** เป็น **struct** โดยมีการกำหนดค่าเริ่มต้นให้กับตัวแปร **student1** คือ

- **id** มีค่าเป็น 1
- **name** มีค่าเป็น **Wararat**
- **age** มีค่าเป็น 36
- **height** มีค่าเป็น 163.0
- **weight** มีค่าเป็น 50.0

บรรทัดที่ 44 ประกาศตัวแปรชื่อ **studentptr** เพื่อเก็บตำแหน่งที่อยู่ตัวแปรที่มี **type** เป็น **student**

กำหนดค่าเริ่มต้นให้กับตัวแปรแบบ struct

```
34  struct student student1, students[10],*studentptr;
35
36  struct student{
37      int id;
38      char name[20];
39      int age;
40      float height;
41      float weight;
42  };
43  struct student student1 = {1,"Wararat",36,163.0,50.0};
44  struct student *studentptr;
45  studentptr = &student1;
46
47  struct student students[] ={1,"Wararat",36,163.0,50.0,
48                              2,"Janya",36,153.0,45.0,
49                              3,"Niwan",37,160.0,48.0};
```

บรรทัดที่ 47 กำหนดค่าเริ่มต้นให้กับตัวแปรแบบ array ของ struct สามารถเขียนได้อีกแบบคือ

```
struct student students[] ={
                            {1,"Wararat",36,163.0,50.0},
                            {2,"Janya",36,153.0,45.0},
                            {3,"Niwan",37,160.0,48.0}
                            };
```

Access struct variable

- การอ้างถึงสมาชิกหรือคุณสมบัติของตัวแปร struct ที่ประกาศขึ้นพิจารณาดังนี้
  - **ตัวแปรธรรมดา**
    - ใช้ชื่อตัวแปรตามด้วย . แล้วตามด้วยคุณสมบัติที่ต้องการ
    - เช่น student1.name;
  - **ตัวแปร array**
    - ใช้ชื่อตัวแปรตามด้วย index แล้วตามด้วย . แล้วตามด้วยคุณสมบัติที่ต้องการ
    - เช่น students[0].name
  - **ตัวแปรพอยน์เตอร์** ทำได้ 2 วิธี ดังตัวอย่าง
    - วิธีที่ 1 คือ (*studentptr).name
    - วิธีที่ 2 คือ studentptr->name

Example

```c
#include<stdio.h>
void main()
{
    struct student{
        int id;
        char name[20];
        int age;
        float height;
        float weight;
    };
    struct student student1 = {1,"Wararat",36,163.0,50.0};
    struct student *studentptr;
    studentptr = &student1;

    struct student students[] ={1,"Wararat",36,163.0,50.0,
                                2,"Janya",36,153.0,45.0,
                                3,"Niwan",37,160.0,48.0};

    printf("Name of student1: %s\n\n", student1.name);

    printf("Name of students[1]: %s\n\n", students[1].name);

    printf("Name of *student: %s\n\n", (*studentptr).name);

    printf("Name of *student: %s\n\n", studentptr->name);

}
```

```
Name of student1: Wararat

Name of students[1]: Janya

Name of *student: Wararat

Name of *student: Wararat
```

Struct member allocated in memory

```
4   struct student{
5       int id;
6       char name[20];
7       int age;
8       float height;
9       float weight;
10  }student1, students[10],*studentptr;
11
12  printf("size of structure in bytes : %d\n", sizeof(student1));
13   printf("\nAddress of student1     = %u", &student1);
14  printf("\nAddress of id           = %u", &student1.id );
15  printf("\nAddress of name         = %u", &student1.name );
16  printf("\nAddress of age          = %u", &student1.age );
17  printf("\nAddress of height       = %u", &student1.height );
18  printf("\nAddress of weight       = %u",&student1.weight);
```

```
size of structure in bytes : 36

Address of student1     = 2686716
Address of id           = 2686716
Address of name         = 2686720
Address of age          = 2686740
Address of height       = 2686744
Address of weight       = 2686748
```

student1

Size = 36

id	name	age	height	weight
716-719	720-739	740-743	744-747	748-751

Struct member allocated in memory

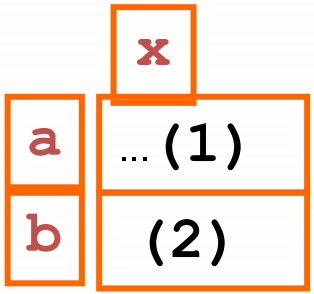
```
 4      struct student{
 5          int id;
 6          char name[20];
 7          int age;
 8          float height;
 9          float weight;
10      }student1, students[10],*studentptr;
11
12      printf("size of structure in bytes : %d\n", sizeof(students));
13      printf("\nAddress of students       = %u\n", students);
14      printf("\nAddress of students[0]    = %u\n", &students[0]);
15      printf("\nAddress of students[1]    = %u\n", &students[1]);
16
17      printf("\nsize of structure in bytes : %d\n", sizeof(studentptr));
18      printf("\nAddress of student1    = %u\n", &student1);
19      studentptr = &student1;
20      printf("value of structureptr  : %u\n", studentptr);
```

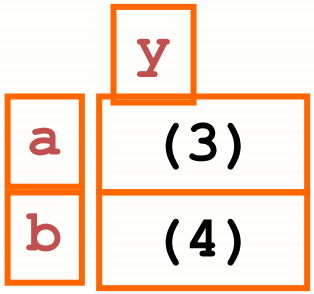
```
size of structure in bytes : 360

Address of students       = 2686352

Address of students[0]    = 2686352

Address of students[1]    = 2686388

size of structure in bytes : 4

Address of student1    = 2686712
value of structureptr  : 2686712
```

Exercise 1

- จงเขียนผลลัพธ์ของโปรแกรมต่อไปนี้

```
#include <stdio.h>
int main()
{
   struct test{
     double a;
     double b;
   } x,y;
    x.a= 44;
    x.b=0.5;
    y.a = x.a+1;
    y.b = x.b +0.25;
   printf("y= %.2f + %.2fi\n",y.a,y.b);
   return 0;
```





ผลลัพธ์
(5)

Exercise 2

- งานกลุ่ม: ตั้งชื่อไฟล์ว่า group x-ex2-L8.pdf
- โปรแกรมเพื่อรับค่าจุดบนแกนโคออดิเนตของรูปสี่เหลี่ยม และทำการคำนวณหาพื้นที่ของรูปสี่เหลี่ยมนี้

```
1  #include <stdio.h>
2  struct point{
3      .....(1).....
4      .....(2).....
5  };
6  void main() {
7      .....(3).....  pt1,pt2 ;
8      int   area;
9      /* Input Data */
10     printf("\nEnter rectangle data\n");
11     printf("\tStart point x : ");
12     ..........(4).............
13     printf("\tStart point y : ");
14     ..........(5).............
15     printf("\tEnd point x : ");
16     ..........(6).............
17     printf("\tEnd point y : ");
18     ..........(7).............
19     /*Compute area*/
20     ..........(8).............
21     printf("\nArea is %d\n\n", area);
22 }
```

```
Enter rectangle data
        Start point x : 1
        Start point y : 2
        End point x : 4
        End point y : 6

Area is 12
```

Exercise 3

- เขียนโปรแกรมเพื่อรับข้อมูลของผู้ใช้จำนวน 5 คน ซึ่งประกอบด้วย
  - ชื่อ(name) นามสกุล(surname) ความสูง(height) และน้ำหนัก(weight)
- แล้วให้คำนวณค่าดัชนีของน้ำหนัก (Body Mass Index : BMI) ซึ่งสามารถคิดได้จากสูตร BMI = w / h^2 โดยที่
  - w แทนน้ำหนักตัวมีหน่วยเป็นกิโลกรัม
  - h แทนความสูงมีหน่วยเป็นเมตร
- หากค่า BMI อยู่ในช่วง
  - 20-25 ให้ขึ้นข้อความว่า “Normal BMI.”
  - อยู่นอกช่วงดังกล่าวให้ขึ้นข้อความว่า “Dangerous BMI.”
- ให้ใช้โครงสร้างทำหน้าที่เก็บข้อมูลของผู้ใช้ ค่า BMI และผลลัพธ์ที่ได้ และให้แสดงข้อมูลทั้งหมด

- ในการกำหนดค่าให้กับ string สามารถใช้

  strcpy(ตัวแปรแบบ array, “ค่าที่จะกำหนดให้กับตัวแปร");

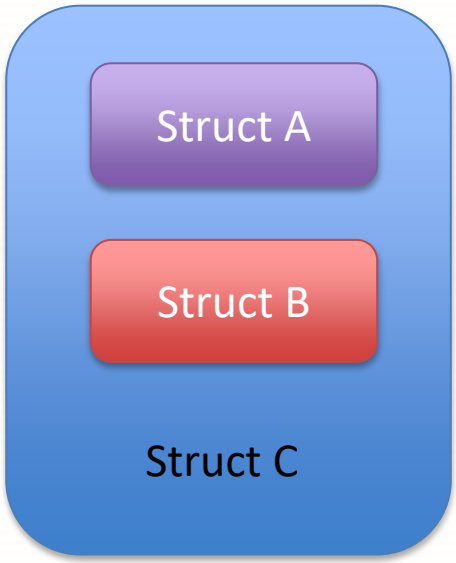
```c
#include <stdio.h>
#include <string.h>
struct  student{
    .......(1)......
    .......(2)......
    .......(3)......
    .......(4)......
    .......(5)......
    .......(6)......
};
void main() {
    .......(7)......
    int i;
    for(i=0;i<3;i++)
    {
        // Input data
        printf("Enter student data\n");
        printf("\tName         : ");
         .......(8)......
         printf("\tSurname      : ");
         .......(9)......
         printf("\tHeight (m)   : ");
         .......(10)......
         printf("\tWeight (Kg) : ");
         .......(11)......
         // Compute BMI
        .......(12)......
        // Find BMI information
        if (.......(13)......)
            .......(14)......
        else
            .......(15)......
     }
    printf("\n\nBMI result");
    for(i=0;i<3;i++)
    {
        printf("\n%s %s weight %.2f kg. height %.2f", .......(16)......);
        printf("\n\tBody mass index %.2f is %s\n\n", .......(17)......);
    }
}
```

ผลการทำงานของโปรแกรม

```
Enter student data
        Name         : Wararat
        Surname      : Jakawat
        Height (m)   : 1.63
        Weight (Kg) : 50.0
Enter student data
        Name         : Janya
        Surname      : Sainui
        Height (m)   : 1.50
        Weight (Kg) : 40.5
Enter student data
        Name         : Titima
        Surname      : Khongkaew
        Height (m)   : 1.60
        Weight (Kg) : 62.5


BMI result
Wararat Jakawat weight 50.00 kg. height 1.63
        Body mass index 18.82 is Dangerous BMI


Janya Sainui weight 40.50 kg. height 1.50
        Body mass index 18.00 is Dangerous BMI


Titima Khongkaew weight 62.50 kg. height 1.60
        Body mass index 24.41 is Normal BMI
```

Nested structures

- Members of a structure can be of any other type including **structure**




```
struct C{
    struct A {...
    }data_a;

    struct B {...
    }data_b;
...
};
```

```
struct A{ ...
};

struct B{
... };

struct C{
   struct A data_a;
   struct B data_b;
... };
```

Example: Nested structures

```c
main()
{
    struct student{
    char name[30];
    int age;
    struct address{
        int house_no;
        char district[20];
        }add;
    }std;

    printf("enter name\n");
    scanf("%s",std.name);
    printf("enter age\n");
    scanf("%d",&std.age);

    printf("enter address :\n");
    printf("enter house no : \n");
    scanf("%d",&std.add.house_no);
    printf("enter district : \n");
    scanf("%s",std.add.district);
}
```

```c
main()
{
    struct address{
        int house_no;
        char district[20];
        }add;

    struct student{
        char name[30];
        int age;
        struct address add1;
    }std;

    printf("enter name\n");
    scanf("%s",std.name);
    printf("enter age\n");
    scanf("%d",&std.age);

    printf("enter address :\n");
    printf("enter house no : \n");
    scanf("%d",&std.add1.house_no);
    printf("enter district : \n");
    scanf("%s",std.add1.district);
}
```

Initializing nested Structures

- Define structures as the following

```
5   struct address{
6       int house_no;
7       char district[20];
8       }add;
9
10  struct student{
11      char name[30];
12      int age;
13      struct address add1;
14  }std;
```

- Write the statements to initialize a student. The student is Jane and 18 years old. The house number is 3 at Songkla.

- What is your answer?

Question

- What are the good things of structures?

- Why do we use nested structures?

Self-referentail structures

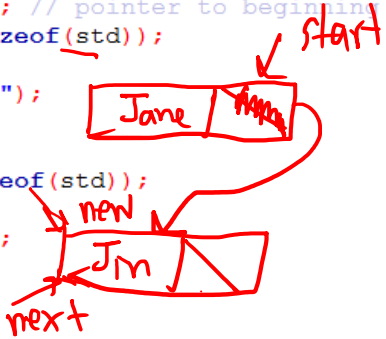
- A member of the structure is a pointer to the parent structure type
- Efficient to insert or remove items at any point in the list
- Example

```
struct tag {
    member 1;
    member 2;
    .......
    struct tag *name
}
```

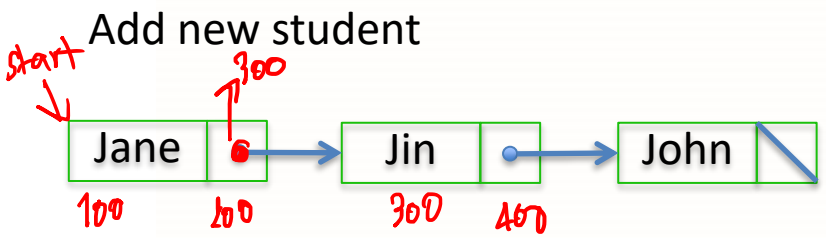
  - Name refer to the name of a pointer variable.
  - The structure of type tag contain a member that points to another structure of type tag

Self-referentail structures

```
4   main()
5   {
6       struct student{
7       char name[10];            // name for a student
8       struct student *nextstd; // pointer to the next student
9       };
10
11      typedef struct student std; // structure type declaration
12
13      std *start, *next, *neww; // pointer to beginning of list
14      start = (std *)malloc(sizeof(std));
15      //create student 1
16      strcpy(start->name,"Jane");
17      start->nextstd = NULL;
18
19      neww = (std *)malloc(sizeof(std));
20      //create student 2
21      strcpy(neww->name,"Jin");
22      neww->nextstd = NULL;
23
24      next = neww;
25
26      start->nextstd = next;
27
28      neww = (std *)malloc(sizeof(std));
29
30      strcpy(neww->name,"John");
31      neww->nextstd = NULL;
32
33      next->nextstd = neww;
34
35      int i=0;
36      for(i=0;i<3;i++)
37      {
38          printf("%s \t",start->name);
39          start = start->nextstd;
40      }
41  }
```



linklist

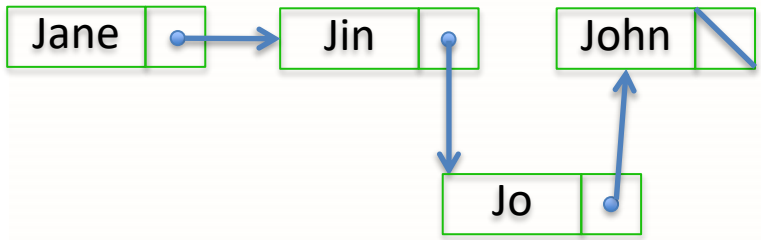



```
Jane    Jin    John
```

Self-referentail structures

```
43      //insert : ex insert Jo between Jin and John
44
45      next = start;
46      neww = (std *)malloc(sizeof(std));
47      strcpy(neww->name,"Jo");
48      neww->nextstd = NULL;
49
50      while(next != NULL)
51      {
52          if(strcmp(next->name,"Jin") == 0)
53          {
54              printf("%s \t",next->name);
55              neww->nextstd = next->nextstd;
56              next->nextstd = neww;
57              break;
58          }
59           next = next->nextstd;
60      }
61
62      printf("\n\nAfter Insert \n\n");
63      next = start;
64      while(next != NULL)
65      {
66          printf("%s \t",next->name);
67          next = next->nextstd;
68      }
```

Insert

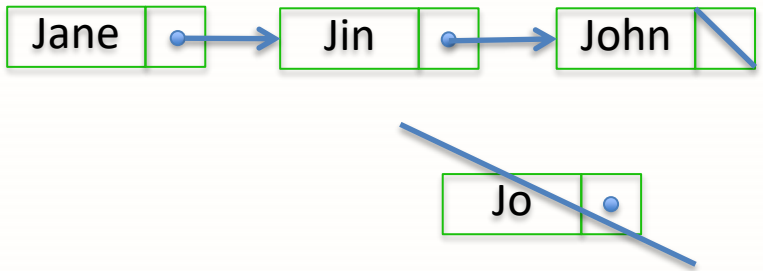



```
Jane    Jin     John

After Insert

Jane    Jin     Jo      John
```

Self-referentail structures

```
70      //remove Jo
71      std *temp,*pre;
72      next = start;
73      while(next != NULL)
74      {
75          if(strcmp(next->name,"Jo") == 0)
76          {
77              temp = next->nextstd;
78              free(next);
79              pre->nextstd = temp;
80              break;
81          }
82           pre = next;
83           next = next->nextstd;
84      }
85
86      printf("\n\nAfter Remove \n\n");
87      next = start;
88      while(next != NULL)
89      {
90          printf("%s \t",next->name);
91          next = next->nextstd;
92      }
```

Delete




```
Jane    Jin     John
After Insert
Jane    Jin     Jo      John
After Remove
Jane    Jin     John
```

Union

- เป็นชนิดข้อมูลที่คล้ายกับชนิดข้อมูลโครงสร้าง
  - เป็นชนิดข้อมูลที่ประกอบด้วยสมาชิกที่มีชนิดข้อมูลหลาย ๆ ประเภท
  - การใช้งานและการอ้างถึงสมาชิกทำได้เหมือนข้อมูลโครงสร้าง
- สิ่งที่แตกต่าง
  - การจัดเก็บข้อมูลของแต่ละสมาชิกที่มีชนิดข้อมูลต่างกันสามารถจะใช้พื้นที่หน่วยความจำร่วมกัน
  - พื้นที่หน่วยความจำของ union จะมีขนาดเท่ากับสมาชิกตัวที่**ใหญ่ที่สุด**
  - เก็บข้อมูลสมาชิกได้ทีละตัว
- รูปแบบการประกาศใช้ union แทน struct

```
union number {
                int x;
                double y;
            };
```

Example

```c
#include<stdio.h>
void main()
{
    union uPoint{
        int  x;
        float  y;
    };
    typedef union uPoint up;
    up p1;

    printf("Size of p1: %d\n\n",sizeof(p1));

    struct sPoint{
        int  x;
        float  y;
    };
    typedef struct sPoint sp;
    sp p2;

    printf("Size of p2: %d\n\n",sizeof(p2));

}
```

```
Size of p1: 4

Size of p2: 8
```

ตัวอย่างการใช้ union

```c
#include<stdio.h>
void main()
{
    union uPoint{
        int  x;
        float  y;
    };
    typedef union uPoint up;
    up p1;

    p1.x = 10;
    printf ("int : %6d,  float : %10.4f\n:", p1.x, p1.y );

    p1.y = 20;
    printf ("int : %6d,  float : %10.4f\n:", p1.x, p1.y );
}
```

```
int :     10,  float :     0.0000
:int : 1101004800,  float :    20.0000
```

- ผลลัพธ์ของการทำงานขึ้นอยู่กับการทำงานแต่ละครั้ง
- การใช้ printf ครั้งแรกแสดง ค่าของ x ได้ถูกต้อง แต่ ค่าของ y เป็นค่าที่คาดเดาไม่ได้
- การใช้ printf ครั้งที่สองแสดง ค่าของ x เป็นค่าที่ไม่สามารถคาดเดาได้ แต่ ค่าของ y เป็นค่าที่ถูกต้อง

Union and Struct

- Using structure when "thing" should be a group of other things
- Using union when "thing" can be one of many different things but *only one at a time*

- การใช้งานยูเนียนใช้ในลักษณะที่มีตัวควบคุมการทำงานขณะนั้นว่ากำลังใช้งานที่สมาชิกตัวใดของยูเนียน
- เพื่อให้สามารถเรียกใช้ข้อมูลได้ถูกต้อง ใช้ยูเนียนร่วมกับข้อมูลโครงสร้าง

Example

- เช่น ต้องการเก็บข้อมูลผลคะแนนของนักเรียน แบ่งการคิดเกรดเป็น
  - นักเรียนประเภท A ให้เก็บผลการเรียนเป็นลักษณะร้อยละ
  - นักเรียนประเภท B ให้เก็บผลการเรียนในลักษณะเกรด A - F

```
typedef   enum  { A, B }  Type;
typedef   union  {
      float   percent;
      char  grade;
} Result;
```

```
typedef  struct {
    char    name[16];
    float    score;
    Type    type;
    Result  result;
} Student ;
```

- percent and grade will be overlap in memory

Enumuration constant

- เป็นการชนิดข้อมูลขึ้นมาใหม่ที่มีสมาชิกได้หลายตัว
- สมาชิกแต่ละตัวจะเป็นค่าคงที่ตัวเลขจำนวนเต็มบวก เรียงต่อกัน เพิ่มขึ้นที่ละ 1
- สำหรับกำหนดให้ข้อความเป็นค่าคงที่ที่เป็นตัวเลขได้ เพื่อสะดวกในการใช้งาน
- รูปแบบการประกาศ
  - enum variable_name {enum1, enum2, ...};

- ตัวอย่าง 1 กรณีไม่กำหนดค่าให้กับสมาชิก ตัวแรกเริ่มด้วยค่าคงที่ 0
  - ตัวอย่างนี้จะได้ Jan=0, Feb =1,...Dec=11;

```
enum months {Jan,Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec}
```

Enumuration constant

- ตัวอย่าง 2 กรณีกำหนดค่าให้กับสมาชิกตัวใดตัวหนึ่ง ตัวถัดไปจะเพิ่มค่าทีละ 1
  - ตัวอย่างนี้จะได้ Jan =1, Feb=2, Mar =10, Apr=11,..

```
enum months {Jan=1,Feb, Mar=10, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec}
```

- การประกาศตัวแปร m ให้มีชนิดเป็น enum months ที่สร้างขึ้น ดังนี้
  - Enum months m;
  - m = Feb // m = 2

- May = ???

Enumuration constant

- ตัวอย่าง 3 สมาชิกของ enum สามารถมีค่าที่เหมือนกันได้

```
enum State {Work = 1, Fail = 0, Freeze = 0};

int main()
{
   printf("%d, %d, %d", Work, Fail, Freeze);
   return 0;
}
```

```
1, 0, 0
```

- ตัวอย่าง 4 สมาชิกของ enum สามารถเป็นตัวเลขจำนวนเต็มลบได้

```
enum State {Work = -1, Fail , Freeze };

int main()
{
   printf("%d, %d, %d", Work, Fail, Freeze);
   return 0;
}
```

```
-1, 0, 1
```

Enumuration constant

- ตัวอย่าง 5 สมาชิกของ enum ต้องไม่เหมือนกัน

```
enum State {Work , Fail };
enum Result {Work, None};

int main()
{
    return 0;
}
```

Line	Message
	=== Build file: "no target" in "no project" (compiler: unknown) ===
3	error: redeclaration of enumerator 'Work'
2	note: previous definition of 'Work' was here
	=== Build failed: 1 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ===

Demonstraing ENUM

```
1   #include  <stdio.h>
2   enum months{
3   Jan=1, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec};
4
5   void main()
6   {
7      enum months month;
8
9       char *monthName[] = { "", "January", "February", "March", "April", "May", "June", "July", "August",
10  "September", "October", "November", "December" };
11
12      for ( month = Jan; month <= Dec; ++month )
13      printf("%2d%11s\n", month, monthName[month]);
14  }
```

```
 1    January
 2   February
 3      March
 4      April
 5        May
 6       June
 7       July
 8     August
 9  September
10    October
11   November
12   December
```

- Line 2-3: ประกาศข้อมูลชนิดใหม่ ชื่อ months ชนิด enum มีสมาชิก 12 ตัว
- Line 7: ประกาศตัวแปรชื่อ month ให้มีชนิดข้อมูลเป็น enum months
- Line 9: ประกาศตัวแปรชื่อ monthName เก็บลิสต์ข้อความ พร้อมกำหนดค่าเริ่มต้นข้อความ
- Line 12-13: วนลูปแสดงชื่อเดือนโดยมีการใช้ index ผ่านตัวแปร month ซึ่งมีชนิดเป็น enum months

Typedef

- เป็นการกำหนดชนิดข้อมูลขึ้นมาใช้เอง มีรูปแบบดังนี้

```
typedef type nameofnewtype;
```

- **ตัวอย่าง 1:** ต้องการกำหนดชนิดข้อมูลชื่อว่า age ขึ้นมาใช้ โดยกำหนดให้เป็นชนิด int

```
typedef int age;
age aa;
```

```
int aa;
```

Typedef and struct

- เป็นคำสั่งที่ใช้กำหนดชนิดข้อมูลใหม่ มักจะมีการนำมาใช้กับโครงสร้างข้อมูล
- การใช้มี 2 ลักษณะ

```
struct point {
    int x;
    int y;
};
typedef struct point Point;
struct point p1;
Point p2;
```

```
typedef struct {
int x;
int y;
} Point ;

Point p1, p2;
```

- บรรทัดที่ 7 เป็นการประกาศตัวแปรชื่อ p2 มีชนิดข้อมูลเป็นโครงสร้าง Point

Structure assignment

- การกำหนดค่าให้กับสมาชิกของ structure สามารถทำได้เช่นเดียวกับตัวแปรปกติ

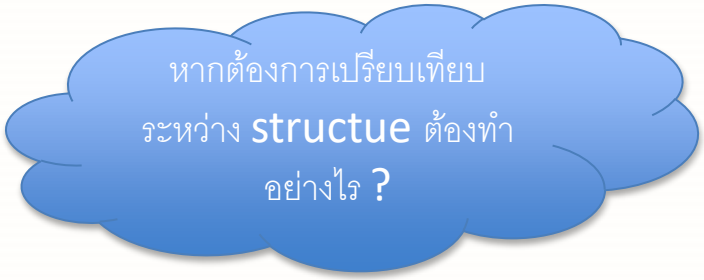
- ตัวอย่างเช่น
  - student1.id = 1;
  - student1.id = students[0].id;
  - student1 = students[0];

```
3   void main()
4   {
5       struct student{
6           int id;
7           char name[20];
8           int age;
9           float height;
10          float weight;
11      } student1;
12
13      struct student students[] ={1,"Jane",36,163.0,50.0,
14                                  2,"John",36,153.0,45.0,
15                                  3,"Jin",37,160.0,48.0};
16
17      student1 = students[0];
18
19      printf("%d %s %d %.2f %.2f\n\n",
20             student1.id,student1.name,student1.age,student1.height,student1.weight);
21  }
```

```
1 Jane 36 163.00 50.00
```

Structure assignment

- แต่**ไม่**สามารถดำเนินการเปรียบเทียบโดยตรงระหว่าง structure
  - student1 == students[0]
  - student1 > students[0]
  - student1 < student[0]




```
23      if(student1 == students[0])
24          printf("yes\n\n");
25      else
26          printf("no\n\n");
27  }
28
```

Logs & others

Code::Blocks | Search results | Cccc | Build log | Build messages | CppCheck | CppCheck

File	Line	Message
		=== Build file: "no target" in "no project" (compiler: unknown) ===
L:\344-241\1-2...		In function 'main':
L:\344-241\1-2...	23	error: invalid operands to binary == (have 'struct student' and 'struct student')
		=== Build failed: 1 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ===