Programming Lab Report

Vivid Padungkiatsakul 6601023620026[1]

[1]Robotic Engineering and Automation System, Faculty of Engineering, King Mongkut's University of Technology North Bangkok

[010243107] C Programming 2023

Lab Week 11

Enumerator & Structure

Topic

- Enumerator
- Structure
- Structure of function (Typedef)

Experiment I

1 Problem description

- Declare enumerator day of a week (Sunday,..., Saturday).
- Declare a variable by using the enumerator and set the value to Wednesday.
- Display the value of the variable.

2 Program design

- Enumerator or enum is the group of constants or unchangeable values.
- From the problem, We have to create enum that contain all day in the week. So, We create enum called "Week" and contain all days in the content.
- After that we have to declare the value by enum and set value to "Wednesday". So first, we declare the variable of the enum is "Day". Then we set "Day" equal to "Wednesday".
- Finally, We display the value of the variable or in this case is "Day", when we print the value of "Day" that in the "Week" enum doesn't set the value of that "state", we will get "3", because when we don't set the value of that "state", it will start at "0". So, "Wednesday" is in the fourth index, the value of "Wednesday" is 3.

3 Program text

```c
#include <stdio.h>
enum Week {
    Sunday,
    Monday,
    Tuesday,
    Wednesday,
    Thursday,
    Friday,
    Saturday
};
enum Week Day;

int main(){
    Day = Wednesday;
    printf("%d",Day);
    return 0;
}
```

4 Terminal output

```
3
```

Experiment II

---

1 Problem description

- Declare a structure to store information of a book.
  - Title: C programming
  - Author: John Smith
  - Price: 250 Baht
  - Weight: 0.5 kg
- Then, declare a variable to store information as initial value and display it on screen.

2 Program design

- Structure or struct is the group of related variables.
- From the problem, We have to create struct that store book information. So, We create struct called "Book" and contain all related variables.
- After that, we create the variable of the struct called "Cpro" that contain the value of the variable that in the "Book".
- Finally, we display the value of members of a structure by call the variable of the struct ("Cpro") then we use dot syntax (".") and the name of the member. For excample, we want get value of "Title", we can access by "Cpro.Title" to get value of "Title" that we set value in "Cpro".

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>
struct Book{
    char Title[50];
    char Author[50];
    int Price;
    float Weight;
};

int main(){
    struct Book Cpro = {
        "C programming",
        "John Smith",
        250,
        0.5
    };

    printf("Title: %s\n", Cpro.Title);
    printf("Author: %s\n", Cpro.Author);
    printf("Price: %d Baht\n", Cpro.Price);
    printf("Weight: %.1f kg\n", Cpro.Weight);

    return 0;
}
```

4 Terminal output

```
Title: C programming
Author: John Smith
Price: 250 Baht
Weight: 0.5 kg
```

Experiment III

---

1 Problem description

- Declare **array of structure** to get information of 5 students from user using scanf.
  - Name
  - ID
  - Age
  - Department
- Display the data on screen.

2 Program design

- From the problem, We have to create struct that store student information. So, We create struct called "Student" and contain all related variables.
- After that, We have to use array of structure to store the user input. We can declare the array of struct by use bracket syntax "[]". In this problem, we have to store 5 students so, we can declare "Std[5]" to store 5 students (Std0-Std4).
- For geting user input and store in array of struct. We use getUserInput() function to get user input value.
- getUserInput(), The function getting the pointer and point to variable in the struct by using arrow operator to set the pointer point.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>

struct Student{
    char Name[50];
    long long int ID;
    int Age;
    char Department[100];
};

void getUserInput(struct Student *s) {
    printf("Enter student name: ");
    scanf("%s", &s->Name);
    printf("Enter student ID: ");
    scanf("%lld", &s->ID);
    printf("Enter student age: ");
    scanf("%d", &s->Age);
    printf("Enter student department: ");
    scanf("%s", &s->Department);
    printf("\n");
}

int main(){
    struct Student Std[5];
    for (int i = 0; i < 5; i++) {
        getUserInput(&Std[i]);
    }

    printf("Student Information:\n");
    for (int i = 0; i < 5; i++) {
        printf("Name: %s, ID: %lld, Age: %d, Department: %s\n", Std[i].Name, Std[i].
            ID, Std[i].Age, Std[i].Department);
    }
    return 0;
}
```

Part 4: Terminal output is on the next page.

4 Terminal output

```
Enter student name: STD1
Enter student ID: 1111111111
Enter student age: 18
Enter student department: PRE

Enter student name: STD2
Enter student ID: 2222222222
Enter student age: 19
Enter student department: EEE

Enter student name: STD3
Enter student ID: 3333333333
Enter student age: 20
Enter student department: CE

Enter student name: STD4
Enter student ID: 4444444444
Enter student age: 21
Enter student department: IE

Enter student name: STD5
Enter student ID: 5555555555
Enter student age: 22
Enter student department: CHE

Student Information:
Name: STD1, ID: 1111111111, Age: 18, Department: PRE
Name: STD2, ID: 2222222222, Age: 19, Department: EEE
Name: STD3, ID: 3333333333, Age: 20, Department: CE
Name: STD4, ID: 4444444444, Age: 21, Department: IE
Name: STD5, ID: 5555555555, Age: 22, Department: CHE
```

Experiment IV

---

1 Problem description

- Given structure of fraction number.

```c
typedef struct {
    int n; //numerator
    int d; //denumerator
} fraction;
```

- Write a program that sum up two fraction numbers by using the given structure above. For example,

$$\frac{x}{y} = \frac{a}{b} + \frac{c}{d}$$
$$x = (a \times d) + (b \times c)$$
$$y = b \times d$$

2 Program design

- From the problem, we have to use **typedef** to define a new name for the structure type. Then sum the fraction by using struct from above.
- So first, we declare **typedef** struct called "fraction". In the struct has n and d variable. n is for numerator value and d is for denumerator Value.
- After that we create function that useed "fraction" type to do the sum fraction called "SumFraction()".
- The SumFraction() function by getting the two fraction struct and sum the fraction like the formula above. Then return other fraction struct that store the result of the sum fraction.

Part 3: Program text is on the next page.

3 Program text

```c
#include <stdio.h>

typedef struct {
    int n;
    int d;
} fraction;

fraction SumFraction(fraction f1, fraction f2){
    fraction result;
    result.n = (f1.n * f2.d) + (f1.d * f2.n);
    result.d = f1.d * f2.d;
    return result;
}

int main(){
    fraction f1 = {1,2};
    fraction f2 = {1,3};
    fraction Sum = SumFraction(f1, f2);
    printf("Result of sum: %d/%d\n", Sum.n, Sum.d);
    return 0;
}
```

4 Terminal output

```
Result of sum: 5/6
```

End of Lab Experiment Week 11