

Міністерство освіти і науки України
Національний університет “Львівська політехніка”



Лабораторна робота №18
з дисципліни
«Програмування частина 2»

Виконав:
Студент групи АП-11
Братейко Вадим

Прийняв:
Чайковський І.Б.

Львів 2024

«Структури та об'єднання даних»

Мета роботи: ознайомитися з поняттями структури та об'єднання даних , навчитися їх використовувати у процесі програмування

Теоретичні відомості

Структура – це сукупність змінних, об'єднаних під одним ім'ям. За допомогою структур зручно розміщувати в суміжних полях пов'язані між собою елементи інформації.

Перед будь-яким використанням структур треба оголосити структурний тип. Оголошення структурного типу має такий вигляд:

```
struct ім'я _структурного_типу {  
    тип_поля ім'я_поля ;  
    ...  
    тип_поля ім'я_поля ;  
};
```

Елементами структури вважаються змінні, декларовані в списку, що обмежується фігурними дужками. Оголошення структури створює шаблон, який можна використовувати для створення її об'єктів (тобто примірників цієї структури). Змінні, з яких складається структура, називаються членами (члени структури ще називаються елементами або полями.) Як правило, члени структури пов'язані один з одним за змістом. Наприклад, елемент списку розсилки, що складається з імені та адреси логічно представити у вигляді структури. У нижченаведеному фрагменті коду показано, як оголосити структуру, в якій визначені поля імені і адреси. Ключове слово `struct` повідомляє компілятору, що оголошується (ще кажуть, "декларується") структура.

```
struct addr  
{  
    char name[30];  
    char street[40];  
    char city[20];  
    char state[3];  
    unsigned long int zip;  
};
```

Приклад 1

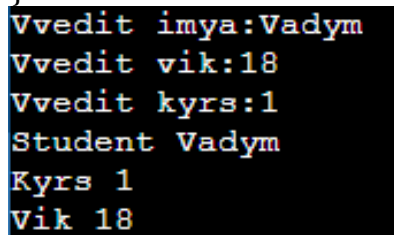
```
#include <stdio.h>  
int main(void)  
{  
    struct {  
        int a;  
        int b;  
    }  
    x, y;  
    x.a = 10;  
    y = x; /* присвоювання одної структури другій */  
    printf("%d", y.a);
```

```
return 0;
}
```

10

Приклад 2

```
#include <stdio.h>
/* визначення структури*/
struct student
{
    char name[30];
    int kurs;
    int age;
};
int main()
{
    /* оголошення змінної stud1 типу struct student*/
    struct student stud1;
    printf("Vvedit imya:");
    gets(stud1.name);
    printf("Vvedit vik:");
    scanf("%d", &stud1.age);
    printf("Vvedit kyrs:");
    scanf("%d", &stud1.kurs);
    printf("Student %s\n", stud1.name);
    printf("Kyrs %d\n", stud1.kurs);
    printf("Vik %d\n", stud1.age);
}
```

A screenshot of a terminal window showing the output of the program. The text is displayed in a monospaced font with color coding: 'Vvedit imya:' is red, 'Vvedit vik:' is green, 'Vvedit kyrs:' is blue, 'Student Vadym' is red, 'Kyrs 1' is green, and 'Vik 18' is blue. The input 'Vadym' for the name and '1' for the course are visible in the previous lines.

```
Vvedit imya:Vadym
Vvedit vik:18
Vvedit kyrs:1
Student Vadym
Kyrs 1
Vik 18
```

Приклад 3

```
#include <stdio.h>
/* визначення структури*/
struct student
{
    char name[30];
    int kurs;
    int age;
};
int main()
{
    /* оголошення масиву на 10 структур */
    struct student stud[10];
    int i, n;
```

```

printf("Kilkict studentiv:");
scanf("%d", &n);
for(i=0;i<n;i++)
{
printf("Vvedit imya:");
scanf("%s", stud[i].name);
printf("Vvedit vik:");
scanf("%d", &stud[i].age);
printf("Vvedit kurs:");
scanf("%d", &stud[i].kurs);
}
/* Виведення */
for(i=0;i<n;i++)
{
printf("Student %s\n", stud[i].name);
printf("Kurs %d\n", stud[i].kurs);
printf("Vik %d\n", stud[i].age);
}
}

```

```

Kilkict studentiv:3
Vvedit imya:Вадим
Vvedit vik:18
Vvedit kurs:1
Vvedit imya:Василь
Vvedit vik:20
Vvedit kurs:1
Vvedit imya:Остап
Vvedit vik:18
Vvedit kurs:1
Student Вадим
Kurs 1
Vik 18
Student Василь
Kurs 1
Vik 20
Student Остап
Kurs 1
Vik 18

```

Приклад 4

```
#include <stdio.h>
```

```

// Визначення структури для зберігання інформації про працівника
struct Employee {
    char name[50];
    float weight;
    float height;
    int age;
}; // Додано крапку з комою

```

```

int main()
{
    // Оголошення змінної типу структури Employee
    struct Employee emp;

    // Зчитування інформації з клавіатури
    printf("Ім'я працівника: ");
    scanf("%s", emp.name);

    printf("Вага працівника: ");
    scanf("%f", &emp.weight);

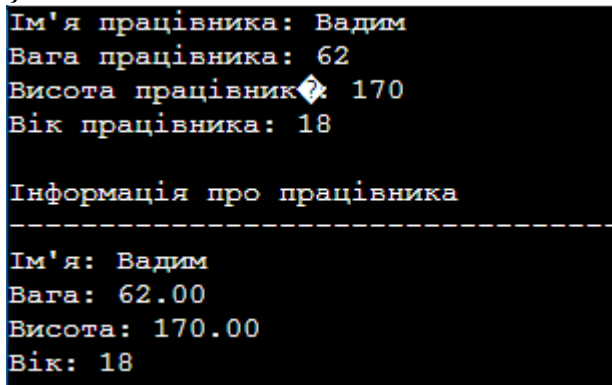
    printf("Висота працівника: ");
    scanf("%f", &emp.height);

    printf("Вік працівника: ");
    scanf("%d", &emp.age);

    // Виведення інформації про працівника
    printf("\nІнформація про працівника\n");
    printf("-----\n");
    printf("Ім'я: %s\n", emp.name);
    printf("Вага: %.2f\n", emp.weight);
    printf("Висота: %.2f\n", emp.height);
    printf("Вік: %d\n", emp.age);

    return 0;
}

```



```

Ім'я працівника: Вадим
Вага працівника: 62
Висота працівника: 170
Вік працівника: 18

Інформація про працівника
-----
Ім'я: Вадим
Вага: 62.00
Висота: 170.00
Вік: 18

```

Відповіді на контрольні запитання

1) Дайте визначення поняття "структура":

Структура в мові програмування - це складений тип даних, який дозволяє об'єднати різноманітні типи даних під одним ім'ям. Вона дозволяє створювати нові типи даних, які складаються з різноманітних полів, що представляють різні аспекти даних.

2) Яким чином здійснюється оголошення структури?

Оголошення структури здійснюється за допомогою ключового слова `struct`, за

яким слідує ім'я структури та опис її полів всередині фігурних дужок:

```
struct <ім'я_структури> {  
    тип_даних поле1;  
    тип_даних поле2;  
    // Інші поля...  
};
```

3) Охарактеризуйте синтаксис об'єднання даних:

Об'єднання в мові програмування - це спеціальний тип даних, який дозволяє різним полям використовувати один і той же блок пам'яті. Синтаксис оголошення об'єднання подібний до синтаксису оголошення структури:

```
union <ім'я_об'єднання> {  
    тип_даних поле1;  
    тип_даних поле2;  
    // Інші поля...  
};
```

4) Які операції не можна застосовувати до структур?

До структур не можна застосовувати операції порівняння за допомогою операторів порівняння ==, !=, <, <=, >, >=. Для порівняння структур потрібно порівнювати їх поля одне за одним.

Висновок: на цій лабораторній роботі я ознайомився з поняттями структури та об'єднання даних, навчився їх використовувати у процесі програмування.