

Структури



Лекція №1

Дисципліна «Програмування ч.2»

2-й семестр



Оголошення структури

Структура – це сукупність різнотипних елементів, яким присвоюється одне ім'я (може бути відсутнім). Вона займає одну ділянку пам'яті.

Елементи, що складають структуру, називаються **полями**.

Змінна типу «структура» повинна бути описана. Опис складається з двох кроків: **опису шаблону** (тобто складу) або типу структури та **опису змінних** структурного типу.

```
struct [<ім'я_структури>]
{
    <тип_1> ім'я_поля_1;
    <тип_2> ім'я_поля_2;
    ...
} st1, st2, ...;
```

де **struct** – ключове слово;
<ім'я_структури> – ім'я типу «структура» (**може бути відсутнім**);
<тип_1>, <тип_2> – імена стандартних або визначених типів;
ім'я_поля_1, ім'я_поля_2, ... – імена полів структури;
st1, st2 ... – імена змінних типу «структура».



Приклад оголошення структури

```
struct Book
{
    char    title[MAXTITL];
    char    author[MAXAUTL];
    float   value;
};
```

Це оголошення (шаблон) описує структуру, яка створюється з двох символьних масивів типу **char** і однієї змінної типу **float**. Воно не створює реальний об'єкт даних, але визначає з чого він буде складатися.

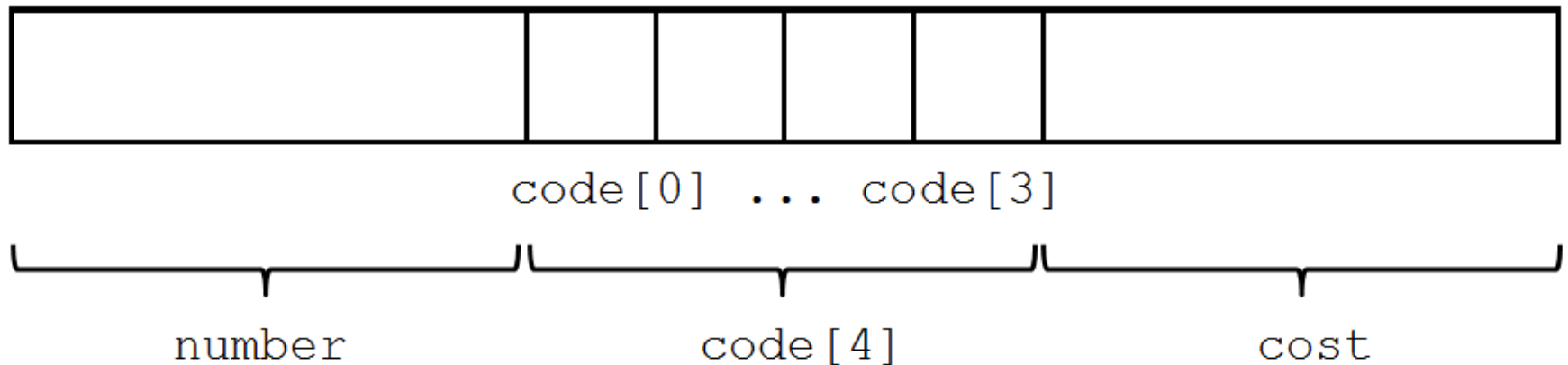
Необхідно оволодіти наступними трьома навичками:

- налаштування формату або схеми для структури;
- оголошення змінної, яка відповідає цій схемі;
- забезпечення доступу до індивідуальних компонентів змінної типу «структура».



Виділення пам'яті під структуру

```
struct Stuff
{
    int    number;
    char   code[4];
    float  cost;
};
```





Ініціалізація структури

```
struct Book
{
    char    title[MAXITL];
    char    author[MAXAUTL];
    float   value;
};

struct Book library =
{
    "Рекреації",
    "Андрухович Юрій Ігорович",
    1.95
};
```



Доступ до членів структури

Доступ до індивідуальних членів структури здійснюється за допомогою операції членства в структурі – **крапка** ('.').

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

char *s_gets(char *st, int n);

#define MAXTITL 41      // максимальна довжина назви + 1
#define MAXAUTL 31     // максимальна довжина імені автора + 1

struct Book             // шаблон структури Book
{
    char title[MAXTITL];
    char author[MAXAUTL];
    float value;
};
```



Доступ до членів структури

```
int main(void)
{
    struct Book library;          // Оголошення змінної типу Book

    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    printf("Введіть назву книги:\n");
    s_gets(library.title, MAXTITL);
    printf("\nВведіть прізвище, ім'я та по батькові автора:\n");
    s_gets(library.author, MAXAUTL);
    printf("\nВведіть ціну книги:\n");
    scanf("%f", &library.value);
    printf("\n\n%s, автор - %s: $%.2f\n", library.title,
           library.author, library.value);
    printf("%s: \"%s\" ($%.2f)\n", library.author,
           library.title, library.value);
    printf("\n");
    return 0;
}
```



Ініціалізатори для структур

Стандарти **c99** і **c11** надають можливість застосовувати призначені ініціалізатори для структур.

При ідентифікації конкретних членів використовують операцію «крапка» та імена членів.

Наприклад, щоб ініціалізувати лише член **value** структури **Book**, можна зробити це наступним чином:

```
struct Book surprise = { .value = 10.99 };
```

Призначені ініціалізатори можна вказувати в будь-якому порядку:

```
struct Book gift = {  
    .value    = 25.99,  
    .author   = "James Broadfool",  
    .title    = "Rue for the Toad"  
};
```




Масиви структур

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

char *s_gets(char *st, int n);

#define MAXTITL    40
#define MAXAUTL    40
#define MAXBOOKS 100           // Максимальна кількість книг

struct Book           // Встановлення шаблону Book
{
    char title[MAXTITL];
    char author[MAXAUTL];
    float value;
};
```



Масиви структур

```
int main(void)
{
    struct Book library[MAXBOOKS]; // Масив структур типу Book
    int count = 0;
    int index;

    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);

    printf("Натисніть [Enter] на початку рядка, "
           "щоб завершити роботу.\n");
    printf("=====\n");
    printf("Книга №%d\n", count+1);
    printf("=====\n");
    printf("Введіть назву книги: ");
    while(count < MAXBOOKS && s_gets(library[count].title,
        MAXTITL) != NULL && library[count].title[0] != '\0')
    {
        printf("Введіть ПІБ автора: ");
```



Масиви структур

```
s_gets(library[count].author, MAXAUTL);  
printf("Введіть ціну: ");  
scanf("%f", &library[count++].value);
```

```
while (getchar() != '\n')  
    continue;           // Очистити вхідний рядок
```

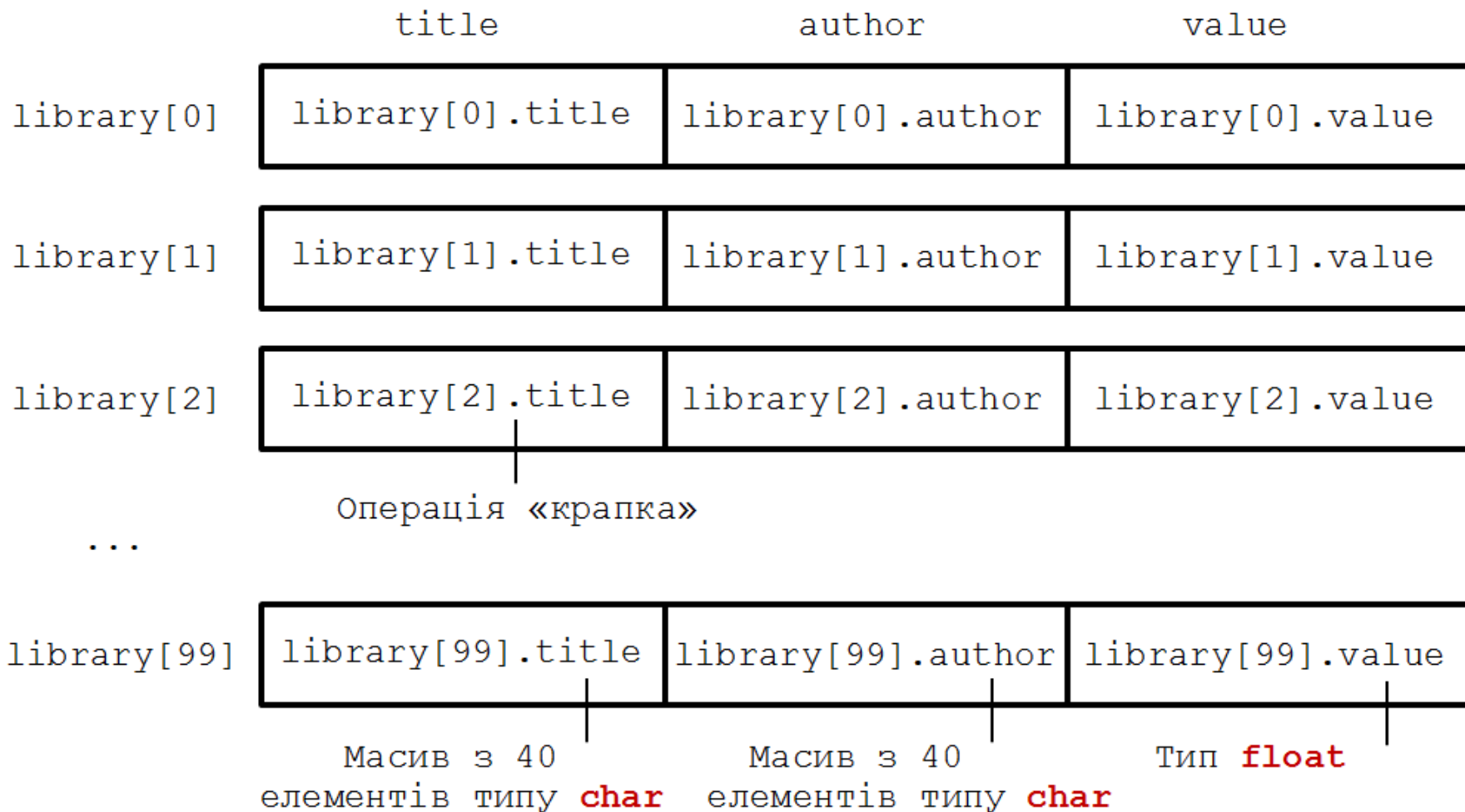
```
if (count < MAXBOOKS)  
{  
    printf("\nНатисніть [Enter] на початку рядка, "  
           "щоб завершити роботу.\n");  
    printf("=====\n");  
    printf("Книга №%d\n", count+1);  
    printf("=====\n");  
    printf("Введіть назву книги: ");  
}  
}
```



Масиви структур

```
if(count > 0)
{
    printf("\n\nКаталог ваших книг:\n\n");
    for(index = 0; index < count; index++)
        printf("%s, автор - %s: $%.2f\n", library[index].title,
            library[index].author, library[index].value);
}
else
    printf("Зовсім немає книг? Це дуже погано.\n");
return 0;
}
```

Схема розміщення масиву структур





Вкладені структури

```
#include <stdio.h>
#include <windows.h>
#define LEN 20

const char *msgs[6] =
{
    "    Дякую Вам за чудово проведений вечір, ",
    "Ви однозначно продемонстрували, що ",
    "являє собою особливий тип людини.",",
    "    Ми обов'язково повинні зустрітися з Вами ",
    "за чудовою вечерею з ",
    "і весело провести час"
};

struct Names // перша структура
{
    char first[LEN];
    char last[LEN];
};
```

Вкладені структури

```
struct Guy
```

```
{
```

```
    struct Names handle;
```

```
    char    favfood[LEN];
```

```
    char    job[LEN];
```

```
    float   income;
```

```
};
```

```
int main(void)
```

```
{
```

```
    struct Guy fellow =
```

```
{
```

```
    { "БІЛЛІ", "БОНС" },
```

```
    "запеченими омарами",
```

```
    "персональний тренер",
```

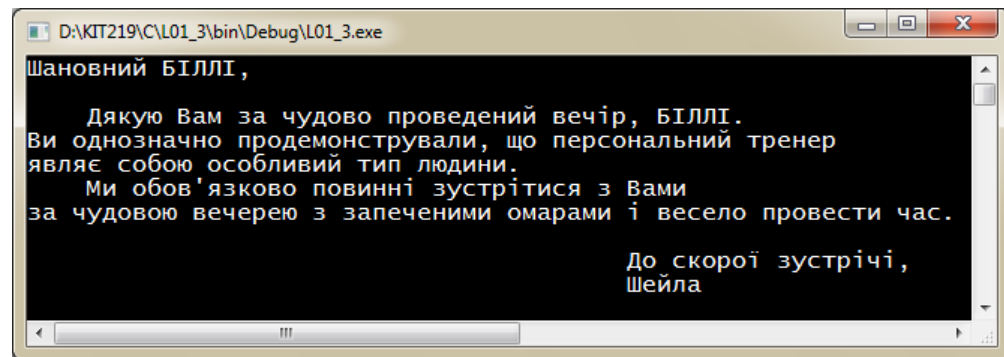
```
    68112.00
```

```
};
```

```
SetConsoleOutputCP(1251);
```

```
// друга структура
```

```
// вкладена структура
```



```
D:\KIT219\C\L01_3\bin\Debug\L01_3.exe
Шановний БІЛЛІ,
    Дякую Вам за чудово проведений вечір, БІЛЛІ.
    Ви однозначно продемонстрували, що персональний тренер
    являє собою особливий тип людини.
    Ми обов'язково повинні зустрітися з Вами
    за чудовою вечерею з запеченими омарами і весело провести час.
                                До скорої зустрічі,
                                Шейла
```



Вкладені структури

```
printf("Шановний %s, \n\n", fellow.handle.first);
printf("%s%s.\n", msgs[0], fellow.handle.first);
printf("%s%s\n", msgs[1], fellow.job);
printf("%s\n", msgs[2]);
printf("%s\n", msgs[3]);
printf("%s%s%s", msgs[4], fellow.favfood, msgs[5]);
if(fellow.income > 150000.0)
    puts("!!!");
else
{
    if(fellow.income > 75000.0)
        puts("!");
    else puts(".");
}
printf("\n%40s%s\n", " ", "До скорої зустрічі,");
printf("%40s%s\n", " ", "Шейла");
return 0;
}
```




Вказівники на структури

Причини використання вказівників на структури.

- 1) З вказівниками на структури працювати переважним чином набагато легше, ніж з самими структурами.
- 2) В деяких застарілих реалізаціях структура не може бути передана як аргумент функції, але вказівник на структуру – може.
- 3) Незважаючи на те, що структуру можна передавати як аргумент, передавання вказівника частіше є більш ефективним.
- 4) Багато інших видів представлення даних застосовують структури, що містять вказівники на інші структури.

Вказівники на структури

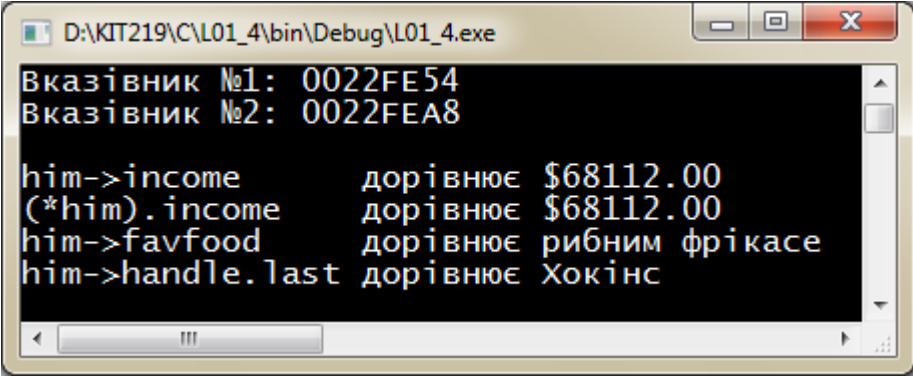
```
#include <stdio.h>
#include <windows.h>
#define LEN 20
```

```
struct Names
```

```
{
    char first[LEN];
    char last[LEN];
};
```

```
struct Guy
```

```
{
    struct Names handle;
    char favfood[LEN];
    char job[LEN];
    float income;
};
```



```
D:\KIT219\C\L01_4\bin\Debug\L01_4.exe
Вказівник №1: 0022FE54
Вказівник №2: 0022FEA8

him->income      дорівнює $68112.00
(*him).income    дорівнює $68112.00
him->favfood      дорівнює рибним фрікасе
him->handle.last  дорівнює Хокінс
```



Вказівники на структури

```
int main(void)
{
    struct Guy fellow[2] =
    {
        { { "Біллі", "Бонс" },
          "запеченими омарами",
          "персональний тренер",
          68112.00
        },

        { { "Джим", "Хокінс" },
          "рибним фрикасе",
          "редактор таблоїду",
          232400.00
        }
    };

    SetConsoleOutputCP(1251);
}
```



Вказівники на структури

```
struct Guy *him;      // вказівник на структуру

printf("Адреса №1:      %p\nАдреса №2:      %p\n",
        &fellow[0], &fellow[1]);

him = &fellow[0];      // повідомляє вказівник, на що
                        // вказувати

printf("\\nВказівник №1: %p\\nВказівник №2: %p\\n",
        him, him + 1);
printf("\\nhim->income      дорівнює $%.2f\\n"
        "(*him).income      дорівнює $%.2f\\n",
        him->income, (*him).income);
him++;                  // вказівник на наступну структуру
printf("him->favfood      дорівнює %s\\n"
        "him->handle.last дорівнює %s\\n",
        him->favfood, him->handle.last);
return 0;

}
```



Вказівники на структури

Оголошення та ініціалізація вказівника на структуру

```
struct Guy barney, *him;           // barney - структура типу Guy  
him = &barney;
```

```
struct Guy fellow[10], *him;       // fellow - це масив структур  
him = &fellow[0];
```

Доступ до членів структури за допомогою вказівника

Перший і найбільш поширений метод передбачає застосування операції '**->**'. Знак цієї операції формується з дефісу (**-**) і символу "більше" (**>**) :

```
him->income дорівнює barney.income, якщо him == &barney  
him->income дорівнює fellow[0].income, якщо him == &fellow[0]
```



Вказівники на структури

Доступ до членів структури за допомогою вказівника

Другий метод:

якщо `him == &fellow[0]`, то `*him == fellow[0]`, оскільки операції `'&'` і `'*'` є оберненими.

Після підстановки отримаємо такий вираз:

```
fellow[0].income == (*him).income
```

Круглі дужки в ньому є обов'язковими, оскільки операція `'.'` має більш високий пріоритет, ніж `'*'`.

Якщо `him` – це вказівник на структуру типу `Guy` на ім'я `barney`, то наступні вирази є еквівалентними:

```
// вважаючи, що him == &barney, маємо  
barney.income == (*him).income == him->income
```



Передавання членів структури

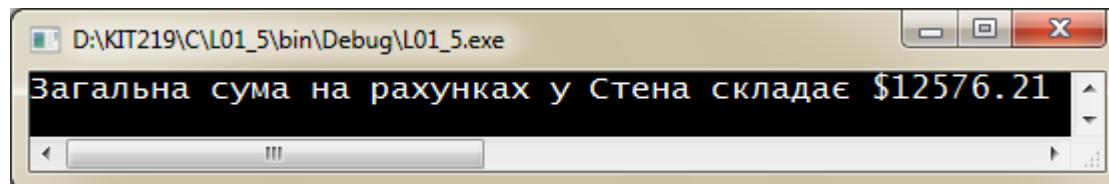
```
#include <stdio.h>
#include <windows.h>
#define FUNDLLEN 50

struct Funds
{
    char    bank[FUNDLLEN];
    double  bankfund;
    char    save[FUNDLLEN];
    double  savefund;
};

double sum(double, double);
int main(void)
{
    struct Funds stan =
    {
        "Garlic-Melon Bank", 4032.27,
        "Lucky1s Savings and Loan", 8543.94
    };
};
```

Передавання членів структури

```
SetConsoleOutputCP(1251);  
printf("Загальна сума на рахунках у Стена складає $%.2f\n",  
       sum(stan.bankfund, stan.savefund));  
return 0;  
}  
  
// Визначення суми двох чисел типу double  
double sum(double x, double y)  
{  
    return (x + y);  
}
```





Використання адреси структури

```
#include <stdio.h>
#include <windows.h>
#define FUNDLLEN 50
struct Funds
{
    char    bank[FUNDLLEN];
    double  bankfund;
    char    save[FUNDLLEN];
    double  savefund;
};
double sum(const struct Funds *); // аргумент є вказівником
int main(void)
{
    struct Funds stan =
    {
        "Garlic-Melon Bank", 4032.27,
        "Lucky1s Savings and Loan", 8543.94
    };
};
```



Використання адреси структури

```
SetConsoleOutputCP(1251);  
printf("Загальна сума на рахунках у Стена складає $%.2f\n",  
       sum(&stan));  
return 0;  
}  
  
double sum(const struct Funds *money)  
{  
    return(money->bankfund + money->savefund);  
}
```

