

# Можливості структур



*Лекція №2*

*Дисципліна «Програмування»*

*2-й семестр*

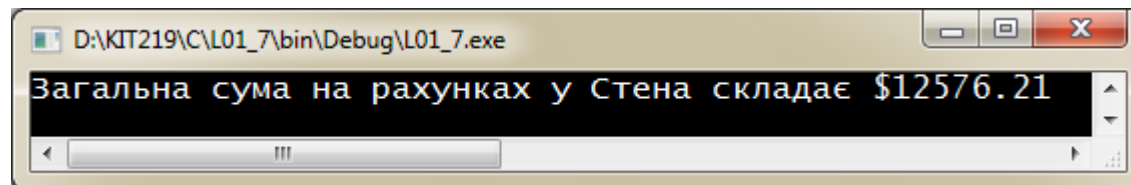


# Передавання структури в якості аргументу функції

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

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

```
double sum(struct Funds moolah); // аргумент є структурою
```





# Передавання структури в якості аргументу функції

```
int main(void)
{
    struct Funds stan =
    {
        "Garlic-Melon Bank",
        4032.27,
        "Lucky's Savings and Loan",
        8543.94
    };
    SetConsoleOutputCP(1251);
    printf("Загальна сума на рахунках у Стена складає $%.2f\n",
        sum(stan));
    return 0;
}

double sum(struct Funds moolah)
{
    return(moolah.bankfund + moolah.savefund);
}
```



# Додаткові можливості структур

1) Можна присвоювати одну структуру іншій: якщо **new\_data** і **old\_data** – структури одного типу, то можна записати:

```
old_data = new_data;    // присвоювання однієї структури іншій
```

Це працює, навіть якщо член структури є масивом.

2) Структуру можна ініціалізувати іншою структурою того ж самого типу:

```
struct Names right_field = { "Джеймс", "Бонд" };  
// ініціалізація структури іншою структурою  
struct Names captain = right_field;
```

3) Структури не тільки можна передавати функції в якості аргументів, але також і повертати їх з функції.



# Додаткові можливості структур

```
#include <stdio.h>
#include <string.h>
#include <windows.h>
#define NLEN 30
```

```
struct Namect
{
    char  fname[NLEN];
    char  lname[NLEN];
    int   letters;
};
```

```
void getinfo(struct Namect *);
void makeinfo(struct Namect *);
void showinfo(const struct Namect *);
char *s_gets(char *st, int n);
```



# Додаткові можливості структур

```
int main(void)
{
    struct Namect person;

    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);

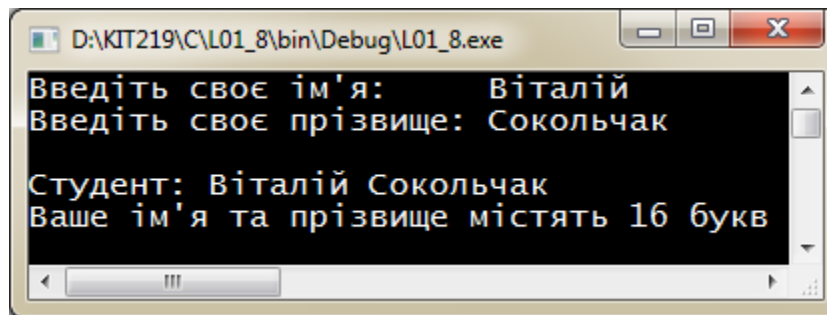
    getinfo(&person);
    makeinfo(&person);
    showinfo(&person);
    return 0;
}

void getinfo(struct Namect *pst)
{
    printf("Введіть своє ім'я: ");
    s_gets(pst->fname, NLEN);
    printf("Введіть своє прізвище: ");
    s_gets(pst->lname, NLEN);
}
```

# Додаткові можливості структур

```
void makeinfo(struct Namect *pst)
{
    pst->letters = strlen(pst->fname) + strlen(pst->lname);
}

void showinfo(const struct Namect *pst)
{
    printf("\nСтудент: ");
    printf("%s %s\nВаше ім'я та прізвище містять %d букв\n",
           pst->fname, pst->lname, pst->letters);
}
```





# Повернення структури

```
#include <stdio.h>
#include <string.h>
#include <windows.h>
#define NLEN 30

struct Namect
{
    char fname[NLEN];
    char lname[NLEN];
    int letters;
};

struct Namect getinfo(void);
struct Namect makeinfo(struct Namect);
void showinfo(struct Namect);
char *s_gets(char *st, int n);
```





# Повернення структури

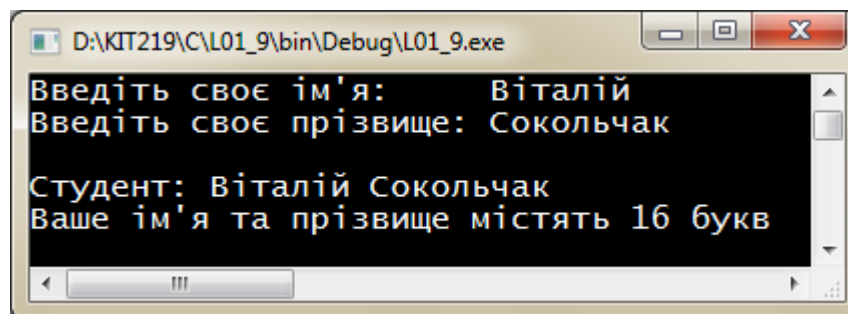
```
int main(void)
{
    struct Namect person;
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    person = getinfo();
    person = makeinfo(person);
    showinfo(person);
    return 0;
}

struct Namect getinfo(void)
{
    struct Namect temp;
    printf("Введіть своє ім'я:      ");
    s_gets(temp.fname, NLEN);
    printf("Введіть своє прізвище: ");
    s_gets(temp.lname, NLEN);
    return temp;
}
```

# Повернення структури

```
struct Namect makeinfo(struct Namect info)
{
    info.letters = strlen(info.fname) + strlen(info.lname);
    return info;
}

void showinfo(struct Namect info)
{
    printf("\nСтудент: ");
    printf("%s %s\nВаше ім'я та прізвище містять %d букв\n",
        info.fname, info.lname, info.letters);
}
```



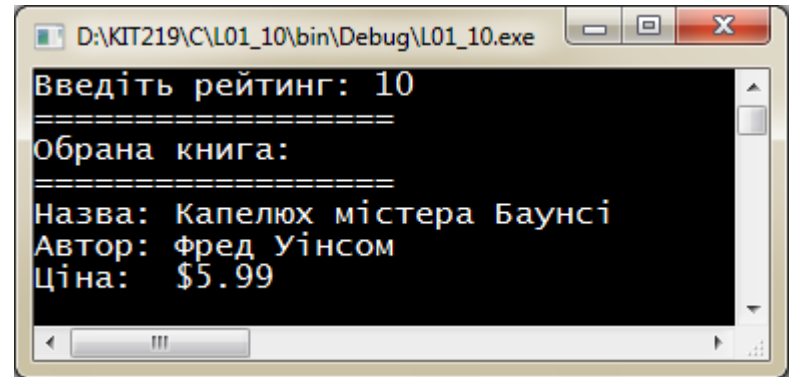
# Складені літерали та структури

```
#include <stdio.h>
#include <windows.h>
#define MAXTITL 41
#define MAXAUTL 31

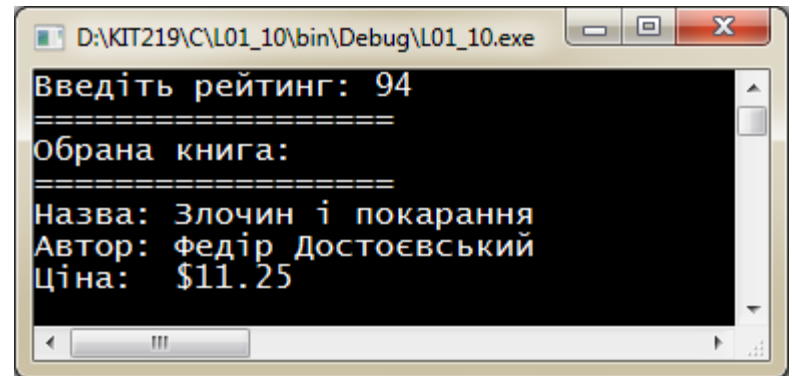
struct Book
{
    char title[MAXTITL];
    char author[MAXAUTL];
    float value;
};

int main(void)
{
    struct Book readfirst;
    int score;

    SetConsoleOutputCP(1251);
```



```
D:\KIT219\C\L01_10\bin\Debug\L01_10.exe
Введіть рейтинг: 10
=====
Обрана книга:
=====
Назва: Капелюх містера Баунсі
Автор: Фред Уїнсом
Ціна: $5.99
```



```
D:\KIT219\C\L01_10\bin\Debug\L01_10.exe
Введіть рейтинг: 94
=====
Обрана книга:
=====
Назва: Злочин і покарання
Автор: Федір Достоевський
Ціна: $11.25
```



# Складені літерали та структури

```
printf("Введіть рейтинг: ");
scanf("%d", &score);
if(score >= 84)
    readfirst = (struct Book) { "Злочин і покарання",
                                "Федір Достоевський",
                                11.25 };
else
    readfirst = (struct Book) { "Капелюх містера Баунсі",
                                "Фред Уінсом",
                                5.99 };

printf("=====\n");
printf("Обрана книга:      \n");
printf("=====\n");
printf("Назва: %s\nАвтор: %s\nЦіна:  $%.2f\n\n",
       readfirst.title, readfirst.author, readfirst.value);
return 0;
}
```



# Складені літерали та структури

Складені літерали можна використовувати також як аргументи функцій. Якщо функція очікує структуру, то їй можна передавати в якості фактичного аргументу складений літерал:

```
struct Rect { double x; double y; };  
double rect_area(struct Rect r) { return r.x * r.y; }  
...  
double area;  
area = rect_area( (struct Rect) { 10.5, 20.0 } );
```

Якщо функція очікує адресу, то їй можна передати адресу складеного літерала:

```
struct Rect { double x; double y; };  
double rect_areap(struct Rect *rp) { return rp->x * rp->y; }  
...  
double area;  
area = rect_areap( &(struct Rect) { 10.5, 20.0 } );
```

В обох випадках змінна **area** отримує значення **210.0**.



# Члени з типами гнучких масивів

В стандарті c99 пропонується новий засіб, що називається **членом типу гнучкого масиву**. Він дозволяє оголошувати структуру, останній член в якій є масивом зі спеціальними властивостями.

Правила створення члена типу гнучкого масиву:

- член типу гнучкого масиву повинен бути останнім у структурі;
- у структурі повинен бути присутнім, принаймні, ще один член іншого типу;
- гнучкий масив оголошується подібно звичайному масиву, але з порожніми квадратними дужками.



# Члени з типами гнучких масивів

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

struct Flex
{
    size_t count;
    double average;
    double scores[];    // член з типом гнучкого масиву
};

void showFlex(const struct Flex *p);

int main(void)
{
    struct Flex *pf1, *pf2;
    int n = 5;
    int i;
    int tot = 0;
```



# Члени з типами гнучких масивів

```
SetConsoleOutputCP(1251);
```

```
// виділення пам'яті для структури та масиву
pf1 = malloc(sizeof(struct Flex) + n * sizeof(double));
pf1->count = n;
for(i = 0; i < n; i++)
{
    pf1->scores[i] = 20.0 - i;
    tot += pf1->scores[i];
}
pf1->average = tot / n;
showFlex(pf1);

n = 9;
tot = 0;
pf2 = malloc(sizeof(struct Flex) + n * sizeof(double));
pf2->count = n;
```





# Члени з типами гнучких масивів

```
for(i = 0; i < n; i++)
{
    pf2->scores[i] = 20.0 - i / 2.0;
    tot += pf2->scores[i];
}
pf2->average = tot / n;
showFlex(pf2);
free(pf1); free(pf2);
return 0;
```

```
D:\KIT219\C\L01_11\bin\Debug\L01_11.exe
Рейтинги: 20 19 18 17 16
Середнє значення: 18
Рейтинги: 20 19.5 19 18.5 18 17.5 17 16.5 16
Середнє значення: 17
```

```
void showFlex(const struct Flex *p)
{
    int i;
    printf("Рейтинги: ");
    for(i = 0; i < p->count; i++)
        printf("%g ", p->scores[i]);
    printf("\nСереднє значення: %g\n", p->average);
}
```



# Члени з типами гнучких масивів

До обробки структур, що містять члени з типами гнучких масивів, пред'являється ряд спеціальних вимог:

1) не використовуйте присвоювання структур для копіювання:

```
struct Flex *pf1, *pf2; // *pf1 і *pf2 є структурами
...
*pf2 = *pf1;           // не робіть таким чином
```

Замість цього застосовуйте функцію `memcpy()`.

2) не використовуйте такі структури спільно з функціями, які передають структури по значенню.

Замість цього застосовуйте функції, які передають адресу структури.

3) не використовуйте структуру з членом типу гнучкого масиву в якості елемента масиву або члена іншої структури.



# Анонімні структури (C11)

**Анонімна структура** – це член структури, який є неіменованою структурою.

```
struct Person
{
    int id;
    struct          // анонімна структура
    {
        char first[20];
        char last[20];
    };
};

struct Person ted = { 8483, { "Ted", "Grass" } };
```

Доступ до членів спрощується, оскільки для цього застосовуються імена членів на зразок **first**, ніби якщо б вони були членами **Person**:

```
puts(ted.first);
```



# Функції, що використовують масив структур

```
#include <stdio.h>
#include <windows.h>
#define FUNDLEN 50
#define N      2
```

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

```
double sum(const struct Funds money[], int n);
```



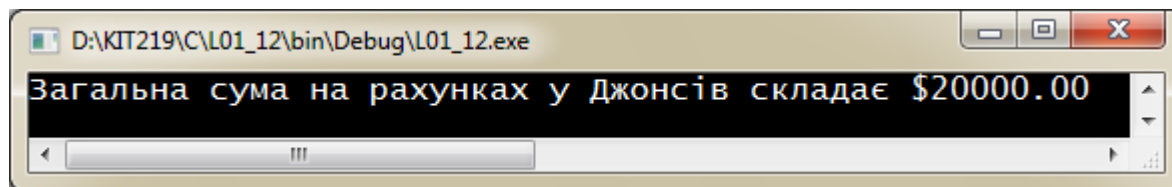
# Функції, що використовують масив структур

```
int main(void)
{
    struct Funds jones[N] =
    {
        {
            "Garlic-Melon Bank",
            4032.27,
            "Lucky's Savings and Loan",
            8543.94
        },
        {
            "Honest Jack's Bank",
            3620.88,
            "Party Time Savings",
            3802.91
        }
    };
    SetConsoleOutputCP(1251);
}
```



# Функції, що використовують масив структур

```
printf("Загальна сума на рахунках у Джонсів складає $%.2f\n",  
      sum(jones, N));  
return 0;  
}  
  
double sum(const struct Funds money[], int n)  
{  
    double total;  
    int i;  
  
    for(i = 0, total = 0; i < n; i++)  
        total += money[i].bankfund + money[i].savefund;  
    return total;  
}
```





# Зберігання вмісту структур у файлі

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

#define MAXTITL 40
#define MAXAUTL 40
#define MAXBOOKS 10           // максимальна кількість книг

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

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



# Зберігання вмісту структур у файлі

```
int main(void)
{
    struct Book library[MAXBOOKS]; // масив структур
    int count = 0;
    int index, filecount;
    FILE *pbooks;

    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);

    int size = sizeof(struct Book);
    if((pbooks = fopen("book.dat", "a+b")) == NULL)
    {
        fputs("Не вдається відкрити файл book.dat\n", stderr);
        exit(1);
    }
    rewind(pbooks);           // перехід на початок файлу
```





# Зберігання вмісту структур у файлі

```
while (count < MAXBOOKS && fread(&library[count],
    size, 1, pbooks) == 1)
{
    if (count == 0)
        puts("Поточний вміст файлу book.dat:");
    printf("%s, автор: %s, $%.2f\n", library[count].title,
        library[count].author, library[count].value);
    count++;
}
filecount = count;
if (count == MAXBOOKS)
{
    fputs("Файл book.dat заповнений", stderr);
    exit(2);
}
puts("Введіть назви нових книг");
puts("Натисніть [enter] на початку рядка, щоб закінчити ввід.");
```



# Зберігання вмісту структур у файлі

```
while (count < MAXBOOKS && s_gets(library[count].title,
    MAXTITL) != NULL && library[count].title[0] != '\0')
{
    puts("Тепер введіть ім'я автора: ");
    s_gets(library[count].author, MAXAUTL);
    puts("Тепер введіть ціну книги: ");
    scanf("%f", &library[count++].value);
    while (getchar() != '\n')
        continue;    // очистити вхідний рядок
    if (count < MAXBOOKS)
        puts("Введіть назву наступної книги: ");
}
if (count > 0)
{
    puts("Каталог ваших книг:");
    for (index = 0; index < count; index++)
        printf("%s авторства %s:  $%.2f\n", library[index].title,
            library[index].author, library[index].value);
```

# Зберігання вмісту структур у файлі

```
    fwrite(&library[filecount], size,  
          count - filecount, pbooks);  
}  
else puts("Зовсім немає книг? Дуже погано.\n");  
puts("Програма завершена.\n");  
fclose(pbooks);  
return 0;  
}
```

```
D:\KIT219\C\L01_13\bin\Debug\L01_13.exe  
Поточний вміст файлу book.dat:  
My Life as a Budgie, автор: Mack Zackles, $12.95  
The CEO Power Diet, автор: Buster Downsize, $19.25  
Введіть назви нових книг  
Натисніть [enter] на початку рядка, щоб закінчити ввід.  
Coping with Coping  
Тепер введіть ім'я автора:  
Dr. Rubin Thonkwacker  
Тепер введіть ціну книги:  
3.02  
Введіть назву наступної книги:  
Diaphanous Frivolity  
Тепер введіть ім'я автора:  
Neda McFey  
Тепер введіть ціну книги:  
29.99  
Введіть назву наступної книги:  
  
Каталог ваших книг:  
My Life as a Budgie авторства Mack Zackles: $12.95  
The CEO Power Diet авторства Buster Downsize: $19.25  
Coping with Coping авторства Dr. Rubin Thonkwacker: $3.02  
Diaphanous Frivolity авторства Neda McFey: $29.99  
Програма завершена.
```