О. О. Решетнік, О. М. Ткаченко

Методичні вказівки до виконання лабораторних робіт з дисципліни «Основи програмування» зі спеціальності «Інженерія програмного забезпечення»

Міністерство освіти і науки України
Вінницький національний технічний університет

Методичні вказівки до виконання лабораторних робіт з дисципліни «Основи програмування» зі спеціальності «Інженерія програмного забезпечення»

Вінниця
ВНТУ
2025

Рекомендовано до видання Радою з якості освіти Вінницького національного технічного університету Міністерства освіти і науки України (протокол № 5 від 21.05.2025 р.)

Рецензенти:
О. К. Колесницький, кандидат технічних наук, професор
А. В. Дудатьєв, кандидат технічних наук, доцент

Методичні вказівки до виконання лабораторних робіт з дисципліни «Основи програмування» зі спеціальності «Інженерія програмного забезпечення» [Електронний ресурс]/ Уклад. О. О. Решетнік, О. М. Ткаченко.– Вінниця : ВНТУ, 2025. – 68 с.

Ці методичні вказівки створено для студентів, які починають знайомство з основами програмування мовою С. У посібнику послідовно розглядаються базові теми курсу: від налаштування середовища розробки та перших простих програм — до роботи з масивами, функціями, рядками та файлами. Окрема увага приділяється таким важливим поняттям, як циклічні структури, умовні оператори та рекурсія. Завершує курс короткий вступ до комп'ютерної графіки. Для кожної теми запропоновано практичні завдання, приклади коду та поетапний опис виконання лабораторних робіт.

Мета цих вказівок — допомогти студентам не лише засвоїти синтаксис мови С, а й навчитися мислити алгоритмічно, розв'язувати прикладні задачі та впевнено працювати з кодом. Матеріал подано у доступній формі, з поступовим нарощуванням складності, що дає змогу використовувати його як у межах аудиторної роботи, так і для самостійного навчання.

ЗМІСТ

Вступ	5
Вступне заняття. - Знайомство з мовою С та середовищем розробки	6
Лабораторна робота №1. - Змінні та типи даних, умовні оператори	8
Лабораторна робота №2. - Цикли	14
Лабораторна робота №3. - Робота з одномірними масивами	21
Лабораторна робота №4. - Функції та рекурсія	30
Лабораторна робота №5. - Робота з багатовимірними масивами	37
Лабораторна робота №6. - Робота з рядками	42
Лабораторна робота №7. - Структури та об'єднання	49
Лабораторна робота №8. - Файли та робота з ними	56
Додаткове завдання.Комп’ютерна графіка	62
Список використаної літератури	67

ВСТУП

Сьогодні програмування — це не просто професійна навичка. Воно стало частиною сучасного світу: ми зустрічаємо його всюди — у телефонах, побутовій техніці, банківських системах, транспорті, медицині й навіть у мистецтві. Уміння програмувати відкриває чимало можливостей: воно допомагає створювати нові речі, автоматизувати рутину й краще розуміти, як влаштовані сучасні технології. А ще програмування вчить мислити: чітко, послідовно, логічно.

Ці методичні вказівки створені спеціально для студентів першого курсу, які щойно починають знайомство з основами програмування. Ми обрали мову програмування С, оскільки вона чудово підходить для старту: дає змогу зрозуміти, як саме «мислить» комп'ютер, і навчитися будувати чіткі, зрозумілі алгоритми. Попри свій поважний вік, С залишається основою багатьох сучасних технологій — і якщо ви опануєте її, інші мови будуть даватися значно легше [1-3].

У посібнику подано вісім лабораторних робіт, що поступово вводять у базові поняття: змінні, умовні оператори, цикли, масиви, функції, роботу з файлами та рядками. Наприкінці — трохи графіки, щоб побачити, як код може перетворюватися на зображення. Кожна тема містить коротке пояснення, приклади та завдання для самостійного виконання. Усе побудовано так, щоби можна було засвоювати матеріал у власному темпі, поступово ускладнюючи задачі.

Варто розуміти, що ті хто починають не можуть знати все. Не бійтесь пробувати, помилятися та пробувати ще. Тільки через практику приходить розуміння. Сміливо експериментуйте, змінюйте код, ставте запитання. Нехай це видання допоможе увійти у світ програмування — світ, де багато викликів, але ще більше відкриттів.

Вступне заняття.

Знайомство з мовою С та середовищем розробки

Мета – Написання простої програми, робота з компілятором, запуск програм.

1. **Підготовка робочого середовища:**

- Встановіть середовище розробки (IDE) або текстовий редактор для роботи з мовою С (наприклад, Visual Studio Code [4], Code::Blocks, або будь-який інший зручний редактор).
- Встановіть компілятор мови С (наприклад, GCC) і переконайтеся, що він правильно налаштований [5].

2. **Створення першої програми:**

- Відкрийте середовище розробки і створіть новий файл з розширенням `.c` (наприклад, `hello_world.c`).

```
#include <stdio.h>


int main() {
    printf("Hello, World!\n");
    return 0;

}
```

- Збережіть файл.

3. **Компіляція програми:**

- Відкрийте термінал або командний рядок у вашому середовищі розробки.
- Перейдіть до директорії, де збережений файл програми (`hello_world.c`).
- Виконайте команду для компіляції програми:

```
gcc hello_world.c -o hello_world
```

```
```

- Після виконання цієї команди в тій же директорії з’явиться виконуваний файл з назвою `hello_world` (або `hello_world.exe` на Windows).

4. **Запуск програми:**

- У тому ж терміналі виконайте команду для запуску програми:

```
./hello_world
```

5. **Внесення змін та повторна компіляція:**

- Внесіть зміни у код програми, наприклад, змініть текст у функції `printf`, на:

```
#include <stdio.h>


int main() {
    printf("Welcome to C programming!\n");
    return 0;
}
```

- Збережіть файл і повторно виконайте компіляцію та запуск програми, як було зроблено раніше.

ЛАБОРАТОРНІ РОБОТИ

Лабораторна робота №1.
Змінні та типи даних, умовні оператори

Мета лабораторної роботи – навчитися створювати змінні різних типів даних, навчитись складати програми для обчислення за простими алгоритмами.

Хід роботи :

Теоретична підготовка:

- Ознайомтесь із базовими типами даних у мові С: `int`, `float`, `double`, `char`.
- Вивчіть, як використовуються умовні оператори: `if`, `else`, `else if`, а також тернарний оператор `? :`.

Створення програми для роботи зі змінними різних типів:

- Створіть новий файл з розширенням `.c` (наприклад, `variables_conditions.c`).
- Напишіть програму, яка оголошує змінні різних типів (цілі, дійсні числа, символи):

```
#include <stdio.h>

int main() {
    int a = 5;
    float b = 3.14;
    double c = 6.283185;
    char d = 'A';

    printf("Integer a = %d\n", a);
    printf("Float b = %.2f\n", b);
    printf("Double c = %.5f\n", c);
    printf("Char d = %c\n", d);

    return 0;
}
```

- Запустіть програму та переконайтеся, що вона правильно виводить значення змінних.

Обчислення за простими алгоритмами:

- Створіть програму для обчислення площі прямокутника за формулою `S = a * b`, де `a` та `b` – сторони прямокутника:

```
#include <stdio.h>

int main() {
    float a, b, area;

    printf("Enter the length of the rectangle: 
");
    scanf("%f", &a);
    printf("Enter the width of the rectangle: 
");
    scanf("%f", &b);

    area = a * b;
    printf("Area of the rectangle: %.2f\n", 
area);

    return 0;
}
```

- Переконайтеся, що програма коректно працює для різних значень сторін.

Використання умовних операторів:

- Доповніть програму, додайте перевірку вхідних даних, використовуючи умовний оператор `if`:

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

```
int main() {
    float a, b, area;

    printf("Enter the length of the rectangle:
");
    scanf("%f", &a);
    if (a < 0) {
        printf("Length must be positive.\n");
        return 33;
    }

    printf("Enter the width of the rectangle:
");
    scanf("%f", &b);
    if (b < 0) {
        printf("Width must be positive.\n");
        return 33;
    }

    area = a * b;
    printf("Area of the rectangle: %.2f\n",
area);

    return 0;
}
```

Виконання індивідуального завдання

- Написати програму, відповідно до запропонованого індивідуального варіанту.
- Варіант вибрати відповідно вашого номеру по списку в журналі групи.
- Дати відповідь на 3 контрольні питання починаючи з вашого номеру по списку в журналі групи.

Вміст звіту

1. Титульна сторінка
2. Тема та мета роботи
3. Навести текст розробленої програми.
4. Рисунки з результатами роботи програми.
5. Навести блок-схему розробленої програми.
6. Висновок
7. Відповідь на контрольні питання.

Варіанти завдань

Завдання 1. Напишіть програму, яка перевіряє, чи можна зі сторін трьох введених трикутника скласти трикутник. Для цього необхідно виконати умови: сума двох будь-яких сторін має бути більше третьої.

Завдання 2. Напишіть програму, яка обчислює корені квадратного рівняння.

Завдання 3. Напишіть програму, яка зчитує з клавіатури три числа та перевіряє, чи утворюють вони арифметичну прогресію.

Завдання 4. Напишіть програму, яка зчитує з клавіатури три числа та перевіряє, чи утворюють вони геометричну прогресію.

Завдання 5. Обчислити значення виразу:

$$P = \frac{\sin(a+b)}{3 * cos^3(a-b) - \pi}$$

Завдання 6. Трикутник заданий координатами x_1, y_1, x_2, y_2, x_3, y_3, своїх вершин. Знайти периметр трикутника.

Завдання 7. Дано змінні x, y. З'ясувати, чи належить крапка з координатами (x, y):
кільцю з центром на початку координат із зовнішнім радіусом 5 і внутрішнім радіусом 10.

Завдання 8. Дано додатні a, b, c, x. З'ясувати чи пройде цегла з ребрами a, b, c у квадратний отвір зі стороною x. Просовувати цегла в отвір можна тільки так, щоб кожне з його ребер було рівнобіжно або перпендикулярно кожної зі сторін отвору. Відповідь одержати в текстовій формі: можна або не можна.

Завдання 9. Нехай на площині задані два концентричних кола радіусів R і r з центром у точці O(0,0) і точка M(x,y). Визначити, до якого кола вона ближча.

Завдання 10. З клавіатури ввести значення трьох дійсних змінних: z1, z2 та z3. Поміняти місцями значення цих змінних так, щоб значення z1 було найменшим, а z3 – найбільшим. Вивести на екран нові значення z1, z2 та z3.

Завдання 11. Дане ціле число, що лежить в діапазоні від 1 до 9999. Вивести рядок –словесний опис даного числа вигляду "парне двозначне число", "непарне чотиризначне число" і т.д.

Завдання 12. Обчислити суму цифр введеного тризначного числа.

Завдання 13. Визначити найбільшу цифру введеного тризначного числа.

Завдання 14. Складіть програму обчислення напруги на кожному з послідовно сполучених ділянок електричного ланцюга опором R1, R2, R3.

Завдання 15. Складіть програму обчислення значення сили струму на ділянці, що складається з двох паралельно сполучених резисторів опором *R1* і *R2*, якщо напруга на кінцях цієї ділянки рівна *U*.

Завдання 16. Напишіть програму, яка перевіряє, чи можна зі сторін трьох введених трикутника скласти прямокутний трикутник.

Завдання 17. Напишіть програму, яка обчислює корені бі-квадратного рівняння.

Завдання 18. Напишіть програму, яка зчитує з клавіатури три числа та перевіряє, чи утворюють вони спадну послідовність.

Завдання 19. Напишіть програму, яка зчитує з клавіатури три числа та перевіряє, чи утворюють вони зростаючу послідовність.

Завдання 20. Обчислити значення виразу:

$$P = \frac{\sin(a+b)}{3 * \sqrt{a-b} - \pi}$$

Завдання 21. Трикутник заданий координатами x_1, y_1, x_2, y_2, x_3, y_3, своїх вершин. Знайти площу трикутника.

Завдання 22. Дано змінні x, y. З'ясувати, чи належить точка з координатами (x, y):
кільцю з центром на початку координат із зовнішнім радіусом 15 і внутрішнім радіусом 30.

Завдання 23. Нехай на площині задані два кола радіусів R і r з центром у точці O(a,b) і точці M(x,y). Визначити, чи перетинаються кола.

Завдання 24. З клавіатури ввести значення трьох дійсних змінних: z1, z2 та z3. Поміняти місцями значення цих змінних так, щоб значення z1 було найбільшим, а z3 – найменшим. Вивести на екран нові значення z1, z2 та z3.

Питання для самоконтролю

1. Чому мова Сі отримала таке поширене використання при програмуванні?
2. Перелічіть етапи створення програми на мові Сі.
3. З чого складається будь-яка програма на мові Сі?
4. З якої функції починається програма на мові Сі?
5. Що входить до алфавіту мови Сі?
6. Поясніть дію операцій інкремента і декремента.
7. Правила використання коментарів в мові Сі.
8. Які знаки логічних і порозрядних операцій в мові Сі ви знаєте?
9. Які типи даних оголошують змінні цілого типу?
10. Які типи даних оголошують змінні дійсного типу?
11. Як збільшити діапазон значень для цілого і дійсного типів?
12. Які ви знаєте символи перетворення в функції виведення printf?
13. Як працює тернарна операція в мові Сі?
14. Поясніть пріоритет виконання операцій мовою Сі.
15. Особливості потокового введення-виведення.

Лабораторна робота №2.
Цикли

Мета лабораторної роботи – навчитися створювати з використанням циклів (for, while, do-while).

Хід роботи :

Теоретична підготовка:

- Ознайомтесь із конструкціями циклів (for, while, do-while)

Створення програми із циклом for:

- Розгляньте програму, що виводить на екран числа від 1 до 10 та їх суму. Прийом по якому в циклі обчислюється сума називається Accumulator [6]. Кожна ітерація додає нові числа в одну і ту ж змінну.
- Цикл for зручний для ітерацій, коли відома кількість повторень заздалегідь.

```
#include <stdio.h>

int main() {
    int sum = 0;
    // Друк чисел від 1 до 10
    for (int i = 1; i <= 10; i++) {
        printf("%d\n", i);
        sum +=i;
    }
    printf("%d\n", sum);

    return 0;
}
```

- Запустіть програму та переконайтеся, що вона правильно виводить значення.

Створення програми із циклом while:

- Розгляньте програму, яка запитує від користувача ціле число більше за 0 поки таке число не буде введено. Цей прийом називається Reprompting **[6].**
- Цикл while зручний для ітерацій, коли не відома кількість повторень заздалегідь, при цьому, якщо умови не задовольняються цикл може взагалі не виконуватись.

```
#include <stdio.h>

int main() {
    int number;

    // Запит користувача ввести число більше 0
    printf("Введіть число більше 0: ");
    scanf("%d", &number);

    while (number <= 0) {
        printf("Невірне значення! Спробуйте ще раз: ");
        scanf("%d", &number);
    }

    printf("Дякую! Ви ввели: %d\n", number);

    return 0;
}
```

- Запустіть програму.

Створення програми із циклом do-while:

- Розгляньте програму, яка запитує від користувача ціле число більше за 0 поки таке число не буде введено.
- Цикл do-while зручний для ітерацій, коли не відома кількість повторень заздалегідь, при цьому, навіть якщо умови не задовольняються цикл виконається хоча б 1 раз.

```
#include <stdio.h>

int main() {
    int number;

    // Запит користувача ввести число більше 0
    do {
        printf("Введіть число більше 0: ");
        scanf("%d", &number);
    } while (number <= 0);

    printf("Дякую! Ви ввели: %d\n", number);

    return 0;
}
```

- Запустіть програму.

Створення програми із if та goto:

- Розгляньте програму, яка виводить на екран числа від 0 до 1 з кроком 0.1.
- Безумовних переходів є дуже потужним інструментом, але надмірне їх використання може призвести до неочікуваних помилок

```
#include <stdio.h>

int main() {
    float i = 0.0;
print_number:

    printf("%.1f\n", i);

    // Збільшуємо i на 0.1
    i += 0.1;

    if (i <= 1.0) {
        goto print_number;
    }

    return 0;
}
```

- Запустіть програму.

Виконання індивідуального завдання

- Написати програму, відповідно до запропонованого індивідуального варіанту. Програма повинна вимагати від користувача вводу правильних параметрів і пропонувати ввести значення повторно.
- Варіант вибрати відповідно вашого номеру по списку в журналі групи.
- Дати відповідь на 3 контрольні питання починаючи з вашого номеру по списку в журналі групи.

Вміст звіту

1. Титульна сторінка
2. Тема та мета роботи
3. Навести текст розробленої програми.
4. Рисунки з результатами роботи програми.
5. Навести блок-схему розробленої програми.
6. Висновок
7. Відповідь на контрольні питання.

Варіанти завдань

Завдання 1. Напишіть програму, яка перевіряє, чи можна зі сторін трьох введених трикутника скласти трикутник. Для цього необхідно виконати умови: сума двох будь-яких сторін має бути більше третьої. Обчислити параметри трикутника: довжини сторін, кути, площу.

Завдання 2. Знайти і роздрукувати всі натуральні трьохзначні числа, які дорівнюють сумі кубів своїх цифр. Використовувати операції ділення по модулю.

Завдання 3. Визначити чи введене число є простим.

Завдання 4. Вивести задану кількість чисел послідовності Фібоначі.

Завдання 5. Задати параметри арифметичної прогресії і порахувати суму її елементів.

Завдання 6. Задати параметри геометричної прогресії і порахувати суму її елементів.

Завдання 7. За введеним цілим числом *M* роздрукувати всі тризначні десяткові числа, сума цифр яких дорівнює М. Підрахувати кількість таких цифр.

Завдання 8. Вивести всі дільники введеного цілого числа.

Завдання 9. За введеним натуральним числом *N* визначити, чи є воно досконалим. Досконале число дорівнює сумі усіх своїх дільників, включаючи одиницю і не включаючи себе.
Наприклад: 6=1+2+3 - досконале число; 8=1+2+4 – недосконале.

Завдання 10. Виведіть всі шестизначні паліндромні числа.

Завдання 11. Дано натуральне число *N>10000*. Обчислити суму його цифр, використовуючи операції ділення по модулю.

Завдання 12. Роздрукувати всі чотиризначні натуральні десяткові числа з діапазону [2000..3000], у запису яких немає двох однакових цифр. Підрахувати кількість таких чисел.

Завдання 13. Трое друзів були свідками ДТП. Перший помітив, що номер порушника ділиться на 2, 7 і 11. Другий запам'ятав, що в запису номера беруть участь всього дві різні цифри, а третій – що сума цифр дорівнює 30. Визначити чотиризначний номер порушника.

Завдання 14. Знайти всі тризначні числа, які можна представити різницею між квадратом числа, утвореного першими двома цифрами, і квадратом третьої цифри.

Завдання 15. Обчислити $\sum_{k=1}^{n} \frac{(-1)^{k+1}}{k \cdot (k+1)}$

Завдання 16. Обчислити $\sum_{i=1}^{100} \sum_{j=1}^{100} \frac{j-i+1}{i+j}$

Завдання 17. Обчислити $\sum_{k=1}^{n} k \cdot (k+1) \cdot \ldots \cdot k^2$

Завдання 18. Обчислити $\prod_{i=1}^{10} \left(2 + \frac{1}{i!}\right)$

Завдання 19. Задано два цілих беззнакових числа. Визначити найменше спільне кратне цих чисел.

Завдання 20. Згенерувати і вивести на екран 5 випадкових трицифрових чисел, кожне з яких повинно містити хоча б одну цифру 7.

Завдання 21. Згенерувати і вивести на екран 5 простих чотирицифрових чисел.

Завдання 22. Напишіть програму, яка виводить на екран прямокутний трикутник, заповнений символами "*", висота якого рівна N. Використовуйте вкладені цикли *while.*

Завдання 23. Визначіть цифри a, b, c, які задовольняють рівняння abb + cab = bac. Дослідіть можливість скорочення повного перебору.

Завдання 24. Визначіть кількість тризначних чисел, сума цифр у яких дорівнює деякому заданому числу n. Виведіть їх на екран. Використовуйте лише два вкладені цикли.

Завдання 25. Напишіть програму, яка виводить ромб, складений з цифр. Висота ромба має бути введена користувачем.

```
    1
   121
  12321
 1234321
123454321
 1234321
  12321
   121
    1
```

Завдання 26. Напишіть програму, яка будує піраміду з чисел Фібоначчі. Кількість рівнів піраміди вводиться користувачем.

```
          0
        1   1
      2   3   5
  8  13  21  34
55 89 144 233 377
```

Завдання 27. Роздрукуйте таблицю значень функції cos(2x) на проміжку [-2;2] з кроком 0,25.

Завдання 28. Роздрукуйте таблицю значень функції sin(3x) на проміжку [-2;2] з кроком 0,25.

Завдання 29. Числа a, b, c називаються «числами Піфагора», якщо $a^2+b^2=c^2$. Визначіть n наборів «чисел Піфагора».

Завдання 30. Знайти суму ряду з точністю ε =10^{-3}, загальний член якого $a_n = 1/2^n + 1/3^n$,

Завдання 31. Знайти суму ряду з точністю ε =10^{-3}, загальний член якого $a_n = (2n-1)/2^n$.

Завдання 32. Надрукувати трикутник Паскаля для двозначних чисел. Кількість
рівнів задає користувач.

Завдання 33. Надрукувати трикутник Хосоя (Фібоначчі) для двозначних чисел. Кількість рівнів задає користувач.

Питання для самоконтролю

1. Поясніть дію операцій інкремента і декремента.
2. Правила використання коментарів в мові Сі.
3. Які знаки логічних і порозрядних операцій в мові Сі ви знаєте?
4. Яка функція дозволяє ввести тільки один символ?
5. Скільки типів різних операцій в мові Сі?
6. Запишіть вираз мовою Сі: “Збільшити на одиницю вміст комірки пам’яті з адресою Z ”.
7. Запишіть вираз мовою Сі: “Збільшити значення по адресу x в 5 разів”
8. Які ви знаєте правила перетворення типів?
9. Поясніть пріоритет виконання операцій мовою Сі.
10. Поясніть роботу оператора розгалуження.
11. Коли величина в умовному операторі може дорівнювати нулю?
12. Які дві форми використання має оператор розгалуження?
13. Наведіть приклади використання простих і складених операторів в умовному виразі.
14. Що таке скорочена форма операторів if ?
15. Як працює умовна тернарна операція?
16. Запишіть мінімальне значення a і b за допомогою умовної операції.

Лабораторна робота №3.

Робота з одномірними масивами

Мета лабораторної роботи – Ознайомитись із простими операціями над масивами: ініціалізація масивів, виконання обчислень з масивами, сортування та пошук.

Хід роботи :

Теоретична підготовка:

- Ознайомтесь із методом декомпозиції.

  Декомпозиція в програмуванні — це процес розбиття великої задачі або складної системи на менші, незалежні частини (підзадачі), які легше розуміти, розробляти, тестувати та підтримувати. Це основний принцип модульного програмування, що дозволяє програмістам створювати більш організовані та керовані програми.

  Декомпозиція допомагає:

- Спрощувати складні завдання, перетворюючи їх на простіші підзадачі.
- Покращувати повторне використання коду**,** дозволяючи використовувати вже написані компоненти.
- Полегшувати тестування та налагодження**,** тому що можна перевіряти кожен модуль окремо.
- Зменшувати складність проекту, покращуючи зрозумілість коду.

Функціональна декомпозиція Це процес розбиття програми на окремі функції, кожна з яких виконує окрему задачу. Кожна функція має чітку відповідальність і може бути використана незалежно.

Приклад**:** Розробимо програму, яка обчислює площу та периметр прямокутника.

```
#include <stdio.h>

// Функція для обчислення площі
float calculateArea(float length, float width) {
    return length * width;
}
```

```
// Функція для обчислення периметра
float calculatePerimeter(float length, float
width) {
    return 2 * (length + width);
}

int main() {
    float length = 5.0, width = 3.0;

    printf("Площа: %.2f\n",
calculateArea(length, width));
    printf("Периметр: %.2f\n",
calculatePerimeter(length, width));

    return 0;
}
```

Пояснення**:** Тут програма розбита на дві функції: `calculateArea` та `calculatePerimeter`, кожна з яких вирішує окреме завдання.

Модульна декомпозиція В цьому випадку програма розбивається на окремі модулі або файли, кожен з яких відповідає за певну частину програми. Це дозволяє працювати з окремими компонентами незалежно.

Приклад**:** Уявімо систему, яка складається з трьох модулів: модуль для роботи з файлами, модуль для математичних операцій та головний модуль програми.

```

// file_operations.h
void readFile();
void writeFile();

// math_operations.h
int add(int a, int b);
int subtract(int a, int b);

// main.c
#include "file_operations.h"
#include "math_operations.h"

int main() {
```

```
    readFile();
    int result = add(5, 3);
    writeFile();
    return 0;
}
```

Пояснення**:** Програма розбита на три частини, що відповідають за різні аспекти роботи: операції з файлами, математичні операції та головний модуль.

Об'єктно-орієнтована декомпозиція У об'єктно-орієнтованому програмуванні (ООП) програма розбивається на класи та об'єкти. Кожен клас відповідає за певну функціональність і об'єднує дані та методи для роботи з ними.

```
class Rectangle {
    private:
        float length;
        float width;
    public:
        Rectangle(float l, float w) : length(l),
width(w) {}

        float calculateArea() {
            return length * width;
        }

        float calculatePerimeter() {
            return 2 * (length + width);
        }
};

int main() {
    Rectangle rect(5.0, 3.0);
    printf("Площа: %.2f\n",
rect.calculateArea());
    printf("Периметр: %.2f\n",
rect.calculatePerimeter());
    return 0;
}
```

Пояснення: Клас `Rectangle` інкапсулює всі операції, що стосуються прямокутника (площа і периметр), об'єднуючи їх у єдину сутність.

Декомпозиція на основі даних Цей підхід полягає в розбитті проблеми на частини, базуючись на структурі даних. Кожна частина обробляє окремі компоненти даних або їх типи.

Приклад: Програма для обробки списку студентів.

```
typedef struct {
    char name[50];
    int age;
    float grade;
} Student;

void printStudent(Student s) {
    printf("Ім'я: %s, Вік: %d, Оцінка: %.2f\n",
s.name, s.age, s.grade);
}

void processStudents(Student students[], int
size) {
    for (int i = 0; i < size; i++) {
        printStudent(students[i]);
    }
}

int main() {
    Student students[2] = {{"Андрій", 20, 89.5},
{"Олена", 22, 91.0}};
    processStudents(students, 2);
    return 0;
}
```

Пояснення: Програма розбивається на частини для обробки окремих даних (студенти) та функцій для роботи з ними.

Приклад заповнення масиву випадковими числами

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

int main() {
    int n; // розмір масиву
    printf("Введіть розмір масиву: ");
    scanf("%d", &n);

    int array[n]; // оголошення масиву

    // Ініціалізація генератора випадкових чисел
    srand(time(NULL));

    // Заповнення масиву випадковими числами
    for (int i = 0; i < n; i++) {
        // випадкове число від 0 до 99
        array[i] = rand() % 100;
    }

    // Виведення масиву
    printf("Масив випадкових чисел:\n");
    for (int i = 0; i < n; i++) {
        printf("%d ", array[i]);
    }

    printf("\n");
    return 0;
}
```

Виконання індивідуального завдання:

- Написати програму, відповідно до запропонованого індивідуального варіанту. Програма повинна вимагати від користувача вводу правильних параметрів і пропонувати ввести значення повторно.
- Варіант вибрати відповідно вашого номеру по списку в журналі групи.
- Дати відповідь на 3 контрольні питання починаючи з вашого номеру по списку в журналі групи.

Вміст звіту

1. Титульна сторінка
2. Тема та мета роботи
3. Декомпозиція задачі
4. Навести текст розробленої програми.
5. Рисунки з результатами роботи програми.
6. Висновок
7. Відповідь на контрольні питання.

Варіанти завдань

Завдання 1. До кондитерського відділу магазину завезли 15 видів цукерок за різною ціною і в різній кількості. Знаючи дані про ціни і кількість товарів, знайти, на які цукерки було витрачено найбільші кошти.

Завдання 2. Сформувати цілочисловий масив A(N), елементами якого є випадкові числа з діапазону [-8, 10]. Знайти серед його елементів два, модуль різниці яких має найбільше значення.

Завдання 3. Дано дійсний масив A(N). Відсортувати його так, щоб всі додатні числа знаходилися на початку, а від‘ємні – в кінці масиву і був збережений початковий порядок проходження елементів в обох групах.

Завдання 4. Сформувати масив SM(50), елементами якого є числа
1, 3, 5, ..., 49, 50, 48, 46, ..., 2.

Завдання 5. Сформувати цілочисловий масив IM(N), елементами якого є випадкові числа з діапазону [3..42]. Підрахувати суму елементів масиву, значення яких кратні 8.

Завдання 6. Дано масив A(N). Знайти пару сусідніх елементів, найближче розташованих один до одного. Міра близькості: R = | A[I+1] – A[I] |.

Завдання 7. Дано масив X(324). Знайти суму елементів масиву, що передують першому від‘ємному елементу. Якщо серед елементів масиву немає від‘ємних, то підсумувати всі елементи.

Завдання 8. Сформувати масив IM(100), елементами якого є числа 2, 1, 4, 3, 6, 5, ... 100, 99 .

Завдання 9. Сформувати масив IM(100), елементами якого є числа 1, 100, 2, 99, 3, 98, ... 50, 51 .

Завдання 10. Сформувати цілочисловий масив A(120), елементами якого є випадкові числа з діапазону [-2..3]. Визначити, скільки разів в ньому зустрілися два нульові елементи, що йдуть підряд.

Завдання 11. Дано цілочисловий масив S[26]. Сформувати масив A який міститиме елементи масиву з парними номерами, а другий масив B – з непарними.

Завдання 12. Сформувати дійсний масив A1[75], елементами якого є випадкові числа з діапазону [16..53]. Переслати з нього в масив A2 всі елементи, значення яких більше 25,8 і менше 34,7.

Завдання 13. У цілочисловому масиві MP(100) непарні елементи збільшити в 2 рази, а у елементів з парними номерами змінити знаки на протилежні.

Завдання 14. У складі баскетбольної команди 12 гравців. Скільки гравців у команді мають зріст, менший за середній зріст команди?

Завдання 15. Сформувати цілочисловий масив A(75), елементами якого є випадкові числа з діапазону [-5, 40]. Переслати в масив У всі елементи, значення яких менше 20.

Завдання 16. У магазині стоїть черга з *N* людей. Час обслуговування і-го покупця t_i – випадкова величина, розподілена за законом рівномірної густини в інтервалі [2.5, 10.4]. Одержати $i_1,i_2,\ldots i_N$ час перебування в черзі кожного покупця. Вказати номер тієї людини, для обслуговування якого потрібно мінімальний час.

Завдання 17. У заданому цілочисловому масиві роздрукувати ті елементи, порядкові номери яких – числа Фібоначчі. (Числа Фібоначчі – 1,1,2,3,5,8,13, 21 і т.д.)

Завдання 18. Даний масив X(100). Переписати в масив Y елементи масиву X з непарними номерами, а в масив Z - елементи масиву X, значення яких кратні п'яти.

Завдання 19. Дано дійсний масив X[n]. Знайти елемент масиву, значення якого найбільш близьке до якого-небудь цілого числа.

Завдання 20. Сформувати цілочисловий масив A[85], елементами якого є випадкові числа з діапазону [-20..10]. Знайти величину найбільшого серед від‘ємних чисел цього масиву.

Завдання 21. Сформувати масив з *n* елементів (n<50), якими можуть бути тільки цілі числа 0 і 1. Перевірити, чи існує строге чергування 0 і 1.

Завдання 22. Ціле число M задане масивом своїх двійкових цифр. Надрукувати масив двійкових цифр числа M + 1.

Завдання 23. Дано цілочисловий масив A(M). Визначити, чи утворюють елементи цього масиву неспадкову послідовність.

Завдання 24. Є цілочисловий масив з *n* елементів. Необхідно "стиснути" цей масив, викинувши з нього нульові елементи. Якщо нульових елементів немає, повідомити, що стиснення неможливе. Додатковий масив не застосовувати.

Завдання 25. Задано одновимірний масив: Написати програму, що визначає індекс елемента, значення якого найближче до середнього арифметичного значення елементів цього масиву.

Завдання 26. В масиві, який заповнений наполовину, продублювати всі елементи з збереженням порядку послідовності (наприклад, якщо задано масив X(3,8,...), одержати масив X(3,3,8,8,...)).

Завдання 27. Задано масив X[i], i=1:n, елементами якого є нулі, одиниці та двійки. Переставити елементи масиву так, щоб масив починався нулями, далі йшли одиниці, а потім двійки. Додатковий масив не використовувати.

Завдання 28. Сформувати цілочисловий масив A(7), елементами якого є випадкові числа з діапазону [-5, 20]. Знайти серед його елементів два, різниця між якими має найбільше значення.

Завдання 29. Проведено вимірювання росту 70 студентів. Дані записані в масиві ROST. Розмістити в масиві NR номери тих студентів, чий ріст менше 180 см, і підрахувати число таких студентів.

Завдання 30. Елементами масиву IM(N) є числа 0 і 1. Відсортувати цей масив так, щоб всі нулі знаходилися на початку, а одиниці – в кінці масиву.

Завдання 31. Сформувати масив IM(100), елементами якого є числа 1, -1, 2, -2,... 50, -50, тобто відбувається чергування знаків.

Завдання 32. Заповніть випадковим чином одномірний масив із n елементів і визначте номер елемента з максимальною сумою сусідів. Сусіди 1-го елемента – останній і другий. Сусіди останнього елемента – передостанній і перший. Виведіть результат на екран.

Питання для самоконтролю

1. Дати визначення терміну “масив”.
2. Які характерні особливості має оголошення масиву в мові Сі.
3. Якими засобами можливо ініціювати масив?
4. Поясніть оператори оголошення масивів рядкових і дійсних даних.
5. У якій послідовності розташовуються в ОП елементи одновимірних масивів і який об'єм ОП вони займають?
6. Як визначити розмір масиву за допомогою команди препроцесора?
7. Що таке покажчик, для чого він використовується в масивах?
8. Якого типу масиви є в мові Сі?
9. Як оголосити одновимірний масив використовуючи покажчики?
10. Як визначити необхідний розмір пам’яті для розміщення масиву?
11. Як оголошуються багатовимірні масиви в мові Сі?
12. Що являє собою трьохвимірний масив?
13. В якому діапазоні змінюються індекси для десятиелементного масиву?
14. Чим відрізняються звернення до елементів масиву в мові С
arr[i], *(ptr + i)
15. Що може використовуватися в якості індексу при оголошенні масиву? При роботі з ним в середині програми.
16. Які є обмеження на кількість вимірювань масиву?
17. Чим може бути корисним рахівник циклу при сумісній роботі з масивом?
18. Що таке покажчик, що зберігається в покажчику?
19. Що таке операція взяття адреси?
20. Що таке константний покажчик? Приклад.
21. Які арифметичні дії можна проводити над покажчиками? Для чого вони використовуються?

Лабораторна робота №4.
Функції та рекурсія

Мета лабораторної роботи – Створення простих та рекурсивних функцій, передача параметрів, повернення значень.

Хід роботи :

Теоретична підготовка:

Функція в мові С — це блок коду, який виконує певне завдання і може бути викликаний кілька разів з різними аргументами. Функції дозволяють структурувати код, робити його більш читабельним та повторно використовуваним.

Основні елементи функцій:

- **Оголошення функції** (або прототип): визначає ім'я функції, її тип повернення та типи параметрів. Воно вказує компілятору, що така функція існує.

```
int sum(int a, int b); // оголошення функції
```

- **Визначення функції**: містить тіло функції, яке виконує певні дії.

```
int sum(int a, int b) {
  return a + b; // тіло функції
}
```

- **Виклик функції**: коли потрібно використовувати функцію, ми викликаємо її з необхідними аргументами.

```
int result = sum(5, 3); // виклик функції sum
```

Передача параметрів у функції

- **Передача за значенням**: передається копія значення аргументів, і зміни в функції не впливають на вихідні змінні.

```
void modify(int x) {
```

```
    x = x + 10;
}
```

- **Передача за вказівником**: передається адреса змінної, тому функція може змінювати оригінальне значення.

```
void modify(int *x) {
    *x = *x + 10;
}
```

Рекурсія — це метод, коли функція викликає саму себе. У програмуванні на C, як і в багатьох інших мовах, рекурсія використовується для розв'язання задач, які можна розбити на підзадачі того ж самого типу.

Рекурсивна функція повинна мати:

- **Базовий випадок** (умова завершення), який визначає, коли рекурсія повинна припинитися.
- **Рекурсивний виклик**, де функція викликає саму себе для вирішення меншої підзадачі.

Проста рекурсія: Факторіал числа

Рекурсія часто використовується для обчислення факторіала числа, де n!n!n! визначається як n!=n×(n−1)!n! = n \times (n-1)!n!=n×(n−1)! і 0!=10! = 10!=1.

```
#include <stdio.h>
// Рекурсивна функція для обчислення факторіала
int factorial(int n) {
    if (n == 0) {
        return 1; // базовий випадок
    } else {
        // рекурсивний виклик
        return n * factorial(n - 1);
    }
}
```

```
int main() {
    int num;
    printf("Введіть число: ");
    scanf("%d", &num);
    printf("Факторіал %d = %d\n", num,
factorial(num));
    return 0;
}
```

Пояснення:

- Якщо n=0n = 0n=0, функція повертає 1 — це **базовий випадок**.
- В іншому випадку функція повертає n×factorial(n−1)n \times factorial(n - 1)n×factorial(n−1), що є **рекурсивним викликом**.

Види рекурсії

- **Пряма рекурсія** — це коли функція викликає саму себе безпосередньо (як у прикладі вище).
- **Непряма рекурсія** — це коли функція викликає іншу функцію, яка в свою чергу викликає першу функцію. Наприклад:

Глибина рекурсії

Рекурсія завжди повинна мати базовий випадок, щоб уникнути **безкінечної рекурсії**, яка призведе до переповнення стека викликів (stack overflow). Глибина рекурсії — це кількість рекурсивних викликів, що виконуються до завершення програми.

Виконання індивідуального завдання:

- Написати програму, відповідно до запропонованого індивідуального варіанту. Програма повинна вимагати від користувача вводу правильних параметрів і пропонувати ввести значення повторно.
- Намагайтесь отримати якнайбільшу кількість функцій у вашій програмі.
- Використайте рекурсію, якщо це можливо.
- Варіант вибрати відповідно вашого номеру по списку в журналі групи.
- Дати відповідь на 3 контрольні питання починаючи з вашого номеру по списку в журналі групи.

Вміст звіту

1. Титульна сторінка
2. Тема та мета роботи
3. Декомпозиція задачі.
4. Навести текст розробленої програми.
5. Рисунки з результатами роботи програми.
6. Висновок
7. Відповідь на контрольні питання.

Варіанти завдань

Завдання 1. Дано натуральне число *n.* Роздрукувати число, що вийде після виписування цифр числа *n* у зворотному порядку. (Для одержання нового числа скласти функцію.)

Завдання 2. Написати і протестувати функцію, що перетворить рядок восьмеричних цифр в еквівалентне їй ціле десяткове число.

Завдання 3. Написати і протестувати функцію, що по заданому натуральному числу визначає кількість цифр у ньому і їхню суму.

Завдання 4. Написати і протестувати функцію, що обчислює $y = \sqrt[3]{x}$ (0<|x|<2), використовуючи ітераційну формулу

$$Y_{i+1} = Y_i + \frac{1}{3}\left(Y_i - \frac{Y_i^4}{X} \right).$$

Початкове наближення $y_o = x$. Ітерації припинити при $|y_{i+1} - y_i| < 2*10^{-6}$.

Завдання 5. З'ясувати, скільки простих чисел знаходиться в інтервалі [*n,m*], і роздрукувати їх. Для визначення, чи є чергове число простим, скласти функцію.

Завдання 6. Написати і протестувати функцію для знаходження в прямокутній матриці номера рядка, що має максимальну суму елементів.

Завдання 7**.** Написати і протестувати функцію, що перетворить рядок двійкових цифр в еквівалентне їй ціле десяткове число.

Завдання 8**.** Написати і протестувати функцію, що у рядку, переданому їй як параметр, заміняє кожен другий елемент на заданий символ.

Завдання 9. Написати і протестувати функцію, що перетворить ціле без знака в його восьмеричне символьне представлення (бібліотечні функції для перетворення числа в рядок і формат виведення ”%o” нс використовувати).

Завдання 10. Розробити функцію, що повертає найменше загальне кратне трьох заданих натуральних чисел.

Завдання 11. Знайти натуральне число з інтервалу [*n1,n2*]з максимальною сумою дільників. (Для знаходження суми дільників числа використати функцію.)

Завдання 12. Площа трикутника, заданого координатами своїх вершин, знаходиться за формулою

$$s = 0.5 \cdot \left| x_1 y_2 + x_2 y_3 + x_3 y_1 - x_1 y_3 - x_2 y_1 - x_3 y_2 \right|$$

Використовуючи функцію для обчислення площі трикутника, визначити площу опуклого чотирикутника ABCD, заданого координатами своїх вершин.

Завдання 13. Написати і протестувати функцію для наближеного обчислення функції $f(x) = e^{-x}$ за формулою

$$f(x) = \frac{1}{\left(\sum_{k=0}^{3} a_k x^k \right)^4}$$

де $a_0 = 1.0$; $a_1 = 0.250721$; $a_2 = 0.029273$; $a_з = 0.003828$.

Завдання 14. Написати і протестувати функцію, що перетворить ціле без знака в його двійкове символьне представлення (бібліотечні функції для перетворення числа в рядок не використовувати).

Завдання 15. Написати і протестувати функцію, що за натуральним k і дійсним

$x > 0$, $0 < \varepsilon < 10^{-6}$ обраховує значення $\sqrt[k]{x}$, використовуючи таку ітераційну формулу: $y_0 = x; \quad y_{i+1} = y_i + \frac{1}{k}\left(\frac{x}{y_i^{k-1}} - y_i\right)$

Як результат береться те значення y_{i+1}, для якого $|y_{i+1} - y_i| < \varepsilon$.

Завдання 16. Написати і протестувати функцію, що переставляє в прямокутній матриці рядки в зворотньому порядку.

Завдання 17**.** Дано натуральні числа *n* и *m.* Написати і протестувати функцію, що повертає результат операції додавання двох чисел, утворених *k* молодшими цифрами числа *n* и *k* старшими цифрами числа *m,*

Завдання 18. Написати і протестувати функцію, що перетворить рядок шістнадцяткових цифр в еквівалентне їй ціле десяткове число.

Завдання 19. Написати функцію для обчислення значення

$$S(n) = \sum_{i=1}^{n} \frac{(2i)!}{(n+1)!}$$

Обчислити з її допомогою значення *S(n)* для *n* від 12 до 24 із кроком 4.

Завдання 20. Написати і протестувати функцію для визначення полярних координат точки за її прямокутними декартовими координатами. Залежність полярних і декартових координат така:

$$r = \sqrt{x^2 + y^2}; \varphi = arctg(y / x)$$

Завдання 21. Дано натуральні числа *n, m* и *k*. Написати і протестувати функцію, що повертає суму, отриману в результаті додавання *k* молодших цифр числа *n* и *k* старших цифр числа *m.*

Завдання 22. Із заданого рядка вибрати і надрукувати ті символи, які зустрічаються в ньому рівно один раз (у тому порядку, як вони зустрічаються в рядку). Обов'язково використати функцію.

Завдання 23. Знайти найбільший спільний дільник трьох натуральних чисел a,b,c. Обов'язково використати функцію

Завдання 24. Знайти найбільше число з трьох натуральних чисел. Обов'язково використати функцію.

Завдання 25. Дано координати вершин двох трикутників. Визначити, який із них має більшу площу. Обов'язково використати функцію

Завдання 26. Знайти найменше спільне кратне двох натуральних чисел M і N з використанням підпрограми знаходження найбільшого спільного дільника двох натуральних чисел.

Завдання 27. Дані відрізки a, в, c, d. Для кожної трійки цих відрізків, з яких можна побудувати трикутник, надрукувати площу даного трикутника (визначити процедуру AREA, яка друкує площу трикутника зі сторонами x, y, z, якщо такий трикутник існує). Обов'язково використати функцію.

Завдання 28. Скласти програму типа FUNCTION для обчислення дисперсії і математичного очікування за формулою:

$$M = \frac{\Sigma x}{n}; \quad D = \sqrt{\frac{\Sigma(x_i - M)^2}{n-1}}$$

n>1, $x_1, x_{2,\ldots} x_n$, – дійсні числа.

Завдання 29. В підпрограмі – функції підрахувати значення функції f(x), визначеній у формулі:

$$f(x) = \sum_{n=0}^{\infty} (-1) \frac{x^n}{(n+3)!}$$

при x=3 для всіх значень n від 1 до 15. Для кожного випадку в підпрограмі надрукувати n і відповідне f(x).

Завдання 30. Описати рекурсивну функцію pow(x,n) від дійсного x (x≠0) і цілого n, який підраховує величину x^n згідно формулі:

$$x^n = \begin{cases} 1, n = 0 \\ \dfrac{1}{x^{|n|}}, n < 0 \\ x \cdot x^{n-1}, n > 0 \end{cases}$$

Питання для самоконтролю

1. Поясніть форму і призначення директив препроцесора.
2. Назвіть основні директиви препроцесора.
3. Поясніть призначення директив #include і #define
4. Поясніть будову функції.
5. Якими способами на мові Сі можна повернути з функції сформований результат?
6. У якому місці файлу програми можна розмістити тексти функцій?
7. Коли і де треба оголосити функцію?
8. Що таке глобальні і локальні змінні?
9. Що таке формальні і фактичні параметри?
10. Які типи даних можуть бути параметрами функцій?
11. Яка відповідність повинна бути між формальними і фактичними параметрами?
12. Як оголосити формальний параметр, якщо фактичним повинний бути вираз?
13. Як оголосити формальний параметр, якщо з його допомогою повинен бути повернений результат виконання функції скалярний, масив?
14. Як можна оголосити формальний параметр, якщо фактичним повинен бути масив?
15. Поясніть форму звернення до значення параметра – змінної функції, якщо формальним параметром є покажчик на скалярну змінну.

Лабораторна робота №5.

Робота з багатовимірними масивами

Мета лабораторної роботи – Ініціалізація багатовимірних масивів, виконання операцій з масивами.

Хід роботи :

Виконання індивідуального завдання

- Написати програму, відповідно до запропонованого індивідуального варіанту. Програма повинна вимагати від користувача вводу правильних параметрів і пропонувати ввести значення повторно.
- Намагайтесь отримати якнайбільшу кількість функцій у вашій програмі.
- Варіант вибрати відповідно вашого номеру по списку в журналі групи.
- Дати відповідь на 3 контрольні питання починаючи з вашого номеру по списку в журналі групи.

Вміст звіту

1. Титульна сторінка
2. Тема та мета роботи
3. Декомпозиція задачі.
4. Навести текст розробленої програми.
5. Рисунки з результатами роботи програми.
6. Висновок
7. Відповідь на контрольні питання.

Варіанти завдань

Завдання 1. Напишіть програму яка отримує від користувача розміри матриці. Потім генерує дві матриці і виводить їх на екран. Виконайте додавання двох матриць і виведіть результат на екран.

Завдання 2. Напишіть програму яка отримує від користувача розміри матриці. Потім генерує дві матриці і виводить їх на екран. Виконайте віднімання двох матриць і виведіть результат на екран.

Завдання 3. Напишіть програму яка отримує від користувача розміри матриці. Потім генерує дві матриці і виводить їх на екран. Виконайте множення двох матриць і виведіть результат на екран.

Завдання 4. Напишіть програму яка отримує від користувача розміри матриці. Потім генерує матрицю і виводить її на екран. Виконайте транспонування і виведіть результат на екран. Виконайте транспонування отриманого результату і виведіть новий результат на екран.

Завдання 5. Напишіть програму яка отримує від користувача розміри матриці (n, m). Потім генерує матрицю і виводить її на екран.
Знайти найбільший за модулем елемент матриціб та його рядок та стовбець. Одержати матрицю порядку (n-1, m-1) шляхом викидання з початкової матриці будь-якого рядка і стовпця, на перетині яких розташований елемент зі знайденим значенням.

Завдання 6. Напишіть програму яка отримує від користувача розміри матриці. Потім генерує дві матриці і виводить їх на екран. Одержати матрицю A×B – B×A.

Завдання 7. Напишіть програму яка отримує від користувача розміри квадратної матриці (n). Потім генерує матрицю A порядку n і вектор b з n елементами. Одержати вектор: $A^2 \times b$

Завдання 8. Напишіть програму яка отримує від користувача розміри квадратної матриці (n). Потім генерує матрицю A порядку n і вектор b з n елементами. Одержати вектор: (A-E) × b, де E- одинична матриця порядку n.

Завдання 9. Напишіть програму яка отримує від користувача розміри квадратної матриці (n). Потім генерує матрицю A порядку n і вектори X і Y з n елементами. Одержати вектор A×(X + Y).

Завдання 10. Напишіть програму яка отримує від користувача розміри квадратної матриці (n). Потім генерує A и B порядку n. Одержати матрицю A×(B - E)+C, де E – одинична матриця порядку n, а елементи матриці C обчислюються по формулі

$$C_{ij} = \frac{1}{i+j}, \quad i,j = 1,2,\ldots,n...$$

Завдання 11. Напишіть програму, яка обчислює суму елементів кожного рядка та кожного стовпця матриці розміром 4x4. Виведіть суми для кожного рядка та стовпця окремо.

Завдання 12. Напишіть програму, яка перевіряє, чи є введена двовимірна матриця 3x3 симетричною відносно головної діагоналі.

Завдання 13. Говорять, що матриця має *сідлову точку* a_{ij}, якщо a_{ij} є мінімальним у і-му рядку і максимальним у, j-му стовпці. Знайти номер рядка і стовпця якої-небудь сідлової точки заданої матриці.

Завдання 14. Знайти максимальний серед всіх елементів тих рядків заданої матриці, що упорядковані (або за зростанням, або за спаданням).

Завдання 15. Напишіть програму, яка генерує матрицю симетричну відносно головної діагоналі заданої розмірності.

Завдання 16. Підрахувати кількість рядків заданої цілочислової матриці розміром 10×10, що є перестановкою чисел 0, 1, 2, ..., 9.

Завдання 17. Знайти рядків заданої цілочислової матриці розміром 10×10, що є перестановкою інших рядків цієї матриці.

Завдання 17. Знайти діагоналі паралельні головній діагоналі заданої цілочислової матриці розміром 10×10, що є підмножиною елементів головної діагоналі цієї матриці.

Завдання 19. Серед стовпців заданої цілочислової матриці, що містять тільки такі елементи, що за модулем не більше 10, знайти стовпець з мінімальним добутком елементів.

Завдання 20. Напишіть програму для генерації магічних квадратів.

Завдання 21.У квадратній матриці A[n×n] знайти суму додатних елементів над головною діагоналлю і під побічною діагоналлю окремо.

Завдання 22. Дана матриця b[m×n] і масив [1],y[2],...,y[n]. Побудувати масив x[1],x[2]..x[n], де x[i] дорівнює сумі трьох елементів і-го рядка матриці які більші за y[i].

Завдання 23. У матриці A[m×n] підсумувати елементи першого рядка, починаючи з першого елемента, другий – з другого і т.д. Результати запам'ятати у масиві x[1], x[2],..., x[n].

Завдання 24. Задана матриця C[m×n]. Знайти окремо суму елементів кожного стовбця і запам'ятати у масиві y[1], y[2],..., y[n]. Причому додавати елементи в стовбці треба тільки до першого від'ємного.

Завдання 25. Дана матриця B[m×n] і масив x[1],x[2],...,x[n], де x[i]>0. Знайти окремо для кожного стовбця суму додатних елементів, але менших за відповідне значення x[i]. Результати запам'ятати у масиві y[1], y[2],...,y[n].

Завдання 26. Сформувати прямокутну матрицю A(10×20) такого вигляду:

$$\begin{bmatrix} 1\ 2\ 3\ .....20 \\ 1\ 2\ 3\ .....20 \\ ..... \\ 1\ 2\ 3\ .....20 \end{bmatrix}$$

Завдання 27. Сформувати квадратну матрицю A(12×12) такого вигляду:

$$\begin{pmatrix} 1\,2\,3\,..\,12 \\ 0\,1\,2\,..\,11 \\ 0\,0\,1\,..\,10 \\ .\ .\ .\ .\ .\ . \\ .\ .\ .\ .\ .\ . \\ 0\,0\,0\,0\,0\,1 \end{pmatrix}$$

Завдання 28. Дана цілочислова матриця A(N×M) (N,M<=10). Побудувати по ній цілочисловий масив B, присвоївши його *k-му* елементу значення 1, якщо *k -й* рядок матриці A симетричний (тобто перший елемент рівний останньому, другий – передостанньому і т.д.), і 0 – інакше.

Завдання 29. Сформувати квадратну матрицю (15×15) такого вигляду:

$$\begin{pmatrix} 1\ 1\ 1\ldots..1 \\ 1\ 2\ 2\ldots..2 \\ 1\ 2\ 3\ldots..3 \\ \ldots\ldots\ldots\ldots \\ 1\ 2\ 3\ \ldots15 \end{pmatrix}$$

Завдання 30. Сформувати квадратну матрицю (15×15) такого вигляду:

$$\begin{pmatrix} 0\,0\,..\,0\,1 \\ 0\,0\,..\,1\,0 \\ .\ .\ .\ .\ .\ . \\ .\ .\ .\ .\ .\ . \\ 0\,1\,..\,0\,0 \\ 1\,0\,..\,0\,0 \end{pmatrix}$$

Завдання 31. Дані дійсні числа a і b (a>b). Сформувати матрицю XY[17×20], елементами якої є дійсні випадкові числа, рівномірно розподілені на відрізку [a,b]. Визначити суму елементів, номери рядків яких кратні 3, а стовпців – 4.

Питання для самоконтролю

1. Дати визначення терміну “масив”.
2. Які характерні особливості має оголошення масиву в мові Сі.
3. Якими засобами можливо ініціювати масив?
4. Поясніть оператори оголошення масивів рядкових і дійсних даних.
5. У якій послідовності розташовуються в ОП елементи одновимірних масивів і який об'єм ОП вони займають?
6. Як визначити розмір масиву за допомогою команди препроцесора?
7. Що таке покажчик, для чого він використовується в масивах?
8. Якого типу масиви є в мові Сі?
9. Як оголосити одновимірний масив використовуючи покажчики?
10. Як визначити необхідний розмір пам’яті для розміщення масиву?
11. Як оголошуються багатовимірні масиви в мові Сі?
12. Що являє собою трьохвимірний масив?
13. В якому діапазоні змінюються індекси для десятиелементного масиву?
14. Чим відрізняються звернення до елементів масиву в мові Сі від відомого раніше.
15. Що може використовуватися в якості індексу при оголошенні масиву? При роботі з ним в середині програми.
16. Які є обмеження на кількість вимірювань масиву?
17. Чим може бути корисним рахівник циклу при сумісній роботі з масивом?
18. Що таке покажчик, що зберігається в покажчику?
19. Що таке операція взяття адреси?
20. Що таке константний покажчик? Приклад.
21. Які арифметичні дії можна проводити над покажчиками? Для чого вони використовуються?

Лабораторна робота №6.
Робота з рядками

Мета лабораторної роботи – Опанувати на практиці операції з рядками та використання бібліотечних функцій для обробки рядків.

Хід роботи :

Unit testing — це процес тестування окремих модулів або компонентів коду для того, щоб перевірити їх правильність. Модулем може бути окремий метод або клас, який виконує певну функцію. Метою є ізоляція кожної частини програми та переконання, що вона працює коректно незалежно від інших частин.

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

int max(int a, int b) {
    return (a > b) ? a : b;
}
// Тест для функції max
void test_max() {
    // очікуємо 5, бо 5 більше за 3
    assert(max(5, 3) == 5);
    // очікуємо 1, бо 1 більше за -1
    assert(max(-1, 1) == 1);
    // очікуємо 0, бо обидва значення рівні
    assert(max(0, 0) == 0);
    // очікуємо -3, бо -3 більше за -5
    assert(max(-5, -3) == -3);
    printf("Тести пройдено\n");
}

void demo();

int main() {
    //Запуск тестового коду
    test_max();
    printf("Усі тести пройдено успішно.\n");
    //Демонстрація роботи
    demo();

    return 0;
}
```

```
void demo() {
    int a = 5;
    int b = 10;
    printf("a = %d, b = %d\n", a, b);

    int m = max(a, b);
    printf("max is %d\n", m);
}
```

Ось приклад юніт-тесту на мові C, що використовує стандартну бібліотеку `assert.h`, яка надає макрос `assert` для перевірки умов. `assert` зручно використовувати для простих перевірок у тестах, коли вам не потрібні додаткові функції з бібліотек, як-от CUnit.

Основні ідеї unit testing:

- **Ізоляція**: Тести виконуються на окремих частинах коду, не залучаючи інших залежностей (наприклад, бази даних чи сторонніх сервісів). Якщо код має залежності, їх замінюють на **mock** або **stub** об'єкти.
- **Автоматизація**: Тести мають бути автоматизованими, щоб їх можна було виконувати при кожній зміні коду. Це допомагає швидко виявляти помилки.
- **Чіткі очікування**: Кожен тест має перевіряти конкретну функціональність та давати однозначну відповідь: тест пройдено чи ні.
- **Швидкість виконання**: Юніт-тести повинні виконуватись дуже швидко, щоб можна було їх запускати часто без значних витрат часу.
- **Документування**: Тести також служать документом, що описує поведінку коду, оскільки показують, як очікується, що модуль працюватиме за певних умов.

Якщо будь-який виклик `assert` у `test_max()` поверне значення `false` (тобто результат не відповідає очікуванню), програма зупиниться і виведе повідомлення про помилку, вказуючи, яка саме перевірка не пройшла. Якщо всі перевірки пройдені успішно, програма завершиться з повідомленням.

Звичайно, це досить спрощений приклад, який демонструє саму ідею юніт-тестування. Він показує основи тестування на C, використовуючи `assert`, але не забезпечує зручностей і потужних можливостей, які потрібні в реальних проєктах. Використання `assert` підходить для невеликих демонстрацій, проте в реальному житті зазвичай потрібні спеціальні бібліотеки та фреймворки для тестування (**Cunit, Check, Unity, Google Test**), які мають розширені функції:

- **Чітке групування тестів**: можливість організувати тести в набори або категорії.
- **Докладний звіт про помилки**: інформація про те, який саме тест не пройдено, а також можливість перегляду результатів усіх тестів одразу.
- **Автоматичне виконання**: можливість інтеграції тестів у систему CI/CD для автоматичного запуску при зміні коду.

Виконання індивідуального завдання

- Вибрати варіант завдання відповідно вашого номеру по списку в журналі групи.
- Написати програму, відповідно до запропонованого індивідуального варіанту.
- Перед написанням коду, провести декомпозицію.
- Намагайтесь отримати якнайбільшу кількість функцій у вашій програмі, що відповідають моделі проведеній декомпозиції.
- Якщо це можливо, то дані для роботи програми потрібно задавати константами, або генерувати за допомогою генератора випадкових чисел.
- Програма повинна вимагати від користувача вводу правильних параметрів і пропонувати ввести некоректні значення повторно.
- Для двох ваших функцій (одна з функцій має бути та яка згадується в завданні) написати unit-тести, використовуючи запропонований підхід з функцією assert.
- Для демонстрації роботи програми, після unit-тестів, реалізувати демонстраційний виклик функцій (лише тих, що згадуються в завданні) програми, роздрукувавши вхідні і вихідні данні.
- Дати відповідь на 3 контрольні питання починаючи з вашого номеру по списку
-

Вміст звіту

1. Титульна сторінка
2. Тема та мета роботи
3. Декомпозиція задачі.
4. Навести текст розробленої програми.
5. Рисунки з результатами роботи програми.
6. Висновок
7. Відповідь на контрольні питання.

Варіанти завдань

Завдання 1. Реалізувати функції:
void showEscape(char* source, char* destination) – копіює текст із строки source в строку destination, при копіюванні тексту перетворить літери “новий рядок” і “табуляція” у видимі послідовності літер “\n” і “\t”.
void hideEscape(char* source, char* destination) – копіює текст із строки source в строку destination, при копіюванні тексту перетворить послідовності літер “\n” і “\t” в літери “новий рядок” і “табуляція”.

Завдання 2. Реалізувати функцію:
int isStrongPassword(char* password) – повертає 1 якщо пароль відповідає всім необхідним правилам, інакше повертає 0.
Правила: 1) містить великі і малі літери англійського алфавіту, 2) містить не менше 2 цифр, 3) довший за 8 символів, 4) однакові символи не можуть йти під ряд.

Завдання 3. Реалізувати функцію:
char* findClosest(char* strings[], int count, char* target) – шукає у масиві строк strings розміром count слово яке є найближчим по відстані до target.
(Відстань між словами – це кількість позицій, у яких слова розрізняються. Наприклад, відстань між словами *МАМА* і ПАПА або *МИШКА* і *КІШКА* дорівнює двом.)

Завдання 4. Реалізувати функцію:
int nextBlank(char *str, int pos) - аналізує рядок *str*, починаючи з позиції *pos* (при цьому нумерація символів починається з 1), і повертає номер першого знайденого пропуску (‘ ‘, ‘\t‘, ‘\n‘), при цьому нумерація символів починається з 1 (тобто якщо пропуском є перший символ рядка то функція поверне 1). Якщо пропуску нема, повертається 0; якщо *pos<1* чи більше довжини рядка, то повертається -1.

Завдання 5. Реалізувати функцію:
void encode(char *source, char *dest) – шифрує текст із source у такий спосіб: записати його в матрицю по символах, а потім переписати по спіралі від центра. Максимальна довжина тексту 25 символів. Для шифрування використовувати матрицю 5 на 5.

Завдання 6. Реалізувати функцію:
char* find(char* strings[], int count, char* target) – шукає у масиві строк strings розміром count слово яке є дорівнює target.

Завдання 7. Реалізувати функцію:
void sort(char* strings[], int count) – сортує масив строк strings розміром count.

Завдання 8. Реалізувати функцію:
void isSubStr(char *str, chat *sub) – визначає чи є рядок sub є підрядком рядка str. Функція повинна повертати номер позиції, з якої починається підрядок, або – 1, якщо підрядок не знайдений.

Завдання 9. Реалізувати функцію:
int removeBracketContent(char *str) – вилучити з рядка str ті символи, що знаходяться між дужками '(' ')', повернути кількість вилучених символів. Самі дужки не видаляти. Якщо хоча б однієї дужки немає повернути -1.

Завдання 10. Реалізувати функцію:
int switchRegister(char *str) – якщо рядок str містить англійські літери в нижньому регістрі, то замінити їх на відповідні літери верхнього регістру та повернути 1, якщо всі літери у верхньому регістрі, то замінити їх на літери в нижньому регістрі і повернути -1.

Завдання 11. Реалізувати функцію:
int strToInt(char *str, int *n) - перетворить рядок десяткових цифр str у ціле число n, якщо перетворення виконати неможливо, то функція має повернути -1.

Завдання 12. Реалізувати функцію:
int strToFloat(char *str, float *n) - перетворить рядок десяткових цифр str у ціле число n, якщо перетворення виконати неможливо, то функція має повернути -1.

Завдання 13. Реалізувати функції:
char topChar(char *str) – повертає символ який зустрічається найчастіше.
char missingLetter(char *str) – повертає англійську літеру яка відсутня в тексті, якщо в тексті немає декількох літер, то повернути ту що перша по алфавіту.

Завдання 14. Реалізувати функцію:
void textStat(char *str, int *words, int *sentences) – записує в words та sentences, кількість слів та речень тексту.

Завдання 15. Реалізувати функцію:
int extractNumbers(char *str, int pos, int *numbers, size) – вилучає з тексту str починаючи з позиції pos цілі числа і заповнює ними масив numbers розміру size, якщо масив заповнено, то функція зупиняє роботу і повертає позицію з строки до якої дійшла.

Завдання 16. Реалізувати функцію:
void toBase64(char *str, char *res) – перетворює строку str у її представлення res у кодування base64.

Завдання 17. Написати і протестувати аналог функції *STRNCPY()*.(Копіює К символів рядка S1 в рядок S2, „хвіст” відкидається або доповнюється пропусками)

Завдання 18. Реалізувати функцію:
void fromBase64(char *base, char *res) – перетворює строку base у форматі base64 у її представлення res у звичайному тексті.

Завдання 19. Реалізувати функції:
int stringCompareCS(char *str1, char *str2) – порівнює дві строки з урахуванням регістру літер і повертає 0 – якщо вони рівні, -1 – якщо str1 лексикографічно перша за str2, 1 – якщо str2 лексикографічно перша за str1.
int stringCompareCIS(char *str1, char *str2) – порівнює дві строки без урахування регістру літер і повертає 0 – якщо вони рівні, -1 – якщо str1 лексикографічно перша за str2, 1 – якщо str2 лексикографічно перша за str1.

Завдання 20. Реалізувати функцію:
void sameWords(char *str1, char *str2, char *words) – записує в строку words слова через кому які є в обох текстах str1 та str2.

Завдання 21. Реалізувати функцію:
void diffWords(char *str1, char *str2, char *words) – записує в строку words слова через кому які є в лише тексті str1 та немає у str2.

Завдання 22. Реалізувати функцію:
void palindromes(char *str, char *words) – аналізує текст із str та записує в строку words слова через кому які є паліндромами.

Завдання 23. Реалізувати функцію:
int anagrams(char *str1, char *str2) – повертає 0 якщо str1 та str2 є анаграмами.

Завдання 24. Реалізувати функцію:
int extractName(char *str, int pos, char *name) – заповнює name іменем яке знайдено в тексті str за певними ознаками починаючи з позиції pos. Повертає позицію за яким знайдено текст або -1 якщо не знайдено. Ознаки імені: лише два слова поряд які починаються з великої букви, велика буква лише на початку слів.

Питання для самоконтролю

1. Дайте визначення рядка? Який символ в С відмічає кінець рядка?
2. Що означає термін конкатенація?
3. Як записується символьна літеральна константа? Рядкова літеральна константа?
4. Що не пишеться в назві рядка, якщо вона передається якій-небудь функції обробки рядків.
5. Чому важливо очищувати вхідний буфер після форматного введення?
6. Напишіть оператор який створює змінну-рядок, яка може містити до 80 символів. Назвіть Line. Char Line[80];
7. В чому різниця між максимальною довжиною і поточною довжиною рядка? Як С визначає поточну довжину? Яка функція визначає поточну довжину?
8. Де знаходяться прототипи функцій обробки рядків?
9. Яка функція використовується для копіювання перших n символів із одного рядка в інший?
10. В чому різниця між функціями strchr() і strrchr()?
11. Що таке поле? Для чого воно використовується?
12. Що таке суміш? Які функції вона виконує в мові Сі? Поясніть на прикладі.
13. За допомогою якого ключового слова можна визначити синонім типу?
14. Що таке нульовий символ? У чому його важливість?
15. У чому різниця між максимальною довжиною і поточною довжиною рядка? Як С визначає поточну довжину? Яка функція повертає поточну довжину?
16. Де знаходяться прототипи функцій обробки рядків?
17. Яка функція використовується для копіювання перших n символів з одного рядка в іншу?
18. У чому різниця між функціями strchr() і strrchr()?

Лабораторна робота №7.
Структури та об'єднання

Мета лабораторної роботи – Опанувати створення та використання структур та об'єднань, робота зі складними структурами даних.

Хід роботи :

Виконання індивідуального завдання

- Вибрати варіант завдання відповідно вашого номеру по списку в журналі групи.
- Написати програму, відповідно до запропонованого індивідуального варіанту.
- Перед написанням коду, провести декомпозицію.
- Намагайтесь отримати якнайбільшу кількість функцій у вашій програмі, що відповідають моделі проведеній декомпозиції.
- Якщо це можливо, то дані для роботи програми потрібно задавати константами, або генерувати за допомогою генератора випадкових чисел.
- Програма повинна вимагати від користувача вводу правильних параметрів і пропонувати ввести некоректні значення повторно.
- Для двох ваших функцій (одна з функцій має бути та яка згадується в завданні) написати unit-тести, використовуючи запропонований підхід з функцією assert.
- Для демонстрації роботи програми, після unit-тестів, реалізувати демонстраційний виклик функцій (лише тих, що згадуються в завданні) програми, роздрукувавши вхідні і вихідні данні.
- Дати відповідь на 3 контрольні питання починаючи з вашого номеру по списку

Вміст звіту

1. Титульна сторінка
2. Тема та мета роботи
3. Декомпозиція задачі.
4. Навести текст розробленої програми.
5. Рисунки з результатами роботи програми.
6. Висновок
7. Відповідь на контрольні питання.

Варіанти завдань

Задача 1. Ввести переліковні типи ***Suit, Value***. З їхньою допомогою описати як структуру змінну ***CARD***.
Скласти і протестувати функцію:
int compare (CARD* hand1, CARD* hand2) – порівнює комбінації з 5 карт по правилам гри в Покер, повертає 0 – якщо вони рівні, -1 – якщо hand1 > hand2, 1 hand1 < hand2.

Задача 2. Описати як структуру **TIME** (з полями години, хвилини, секунди). Скласти і протестувати функції:
int diffSeconds(TIME t1, TIME t2) – обчислює різницю між двома точками в часі в секундах;
int diffMinutes(TIME t1, TIME t2) – обчислює різницю між двома точками в часі в хвилинах;
int diffHours(TIME t1, TIME t2) – обчислює різницю між двома точками в часі в годинах;
TIME add(TIME t1, TIME t2) – додає два проміжки часу.

Задача 3. Ввести переліковні типи **vertical, horizontal** для позначення клітин шахової дошки. Описати структуру CELL для опису координат клітини на дошці.
Скласти і протестувати функцію:
int horseMoves (CELL position, CELL target) - яка обчислює, за скільки ходів кінь може перейти з поля position на поле target.

Задача 4. Увести структуру FRACTION (з полями **чисельник і знаменник**) для опису поняття **раціональне** число. Увести структуру FRACTION_ARRAY, що містить масив та кількість елементів у ньому.
Реалізувати функції:
int compare(FRACTION a, FRACTION b) – повертає 0 якщо числа рівні, -1 – a < b, 1 – a>b.
void sort(FRACTION_ARRAY* arr) – сортує масив методом вставок.

Задача 5. Увести структуру FRACTION (з полями **чисельник і знаменник**) для опису поняття **раціональне** число. Увести структуру FRACTION_ARRAY, що містить масив та кількість елементів у ньому.
Реалізувати функції:
int compare(FRACTION a, FRACTION b) – повертає 0 якщо числа рівні, -1 – a < b, 1 – a>b.
void sort(FRACTION_ARRAY* arr) – сортує масив бульбашковим методом.

Задача 6. Увести структуру FRACTION (з полями **чисельник і знаменник**) для опису поняття **раціональне** число. Увести структуру FRACTION_ARRAY, що містить масив та кількість елементів у ньому.
Реалізувати функції:
int compare(FRACTION a, FRACTION b) – повертає 0 якщо числа рівні, -1 – a < b, 1 – a>b.
void sort(FRACTION_ARRAY* arr) – сортує масив методом швидкого сортування.

Задача 7**.** Увести структуру FRACTION (з полями **чисельник і знаменник**) для опису поняття **раціональне** число. Увести структуру FRACTION_ARRAY, що містить масив та кількість елементів у ньому.
Реалізувати функції:
int compare(FRACTION a, FRACTION b) – повертає 0 якщо числа рівні, -1 – a < b, 1 – a>b.
void sort(FRACTION_ARRAY* arr) – сортує масив методом сортування злиттям.

Задача 8. Увести структуру FRACTION (з полями **чисельник і знаменник**) для опису поняття **раціональне** число.
Реалізувати функції:
FRACTION add(FRACTION a, FRACTION b)
FRACTION sub(FRACTION a, FRACTION b)
FRACTION mul(FRACTION a, FRACTION b)
FRACTION div(FRACTION a, FRACTION b)
FRACTION pow(FRACTION base, FRACTION exp)
Результат кожної операції має бути спрощено, наприклад 3/9 спрощено до 1/3.

Задача 9. Ввести структуру DATE (з полями **число, місяць, рік**) для опису поняття **дата**. Реалізувати функції:
DATE addHours(DATE d, int hours) – обчислює дату через задану кількість годин, при цьому -10000 < hours < 10000. Якщо введено некоректні параметри, то повернути структуру із всіма полями заповненими -1.
int diffMonths(DATE start, DATE end) – визначає різницю між датами в повних місяцях, при цьому start <= end, різниця між введеними датами не більша за 100 років. Якщо введено некоректні параметри, то повернути -1.
int diffYears(DATE start, DATE end) – визначає різницю між датами в повних роках, при цьому start <= end, різниця між введеними датами не більша за 100 років. Якщо введено некоректні параметри, то повернути -1.

Задача 10. Ввести структуру DATE (з полями **число, місяць, рік, назваДня**) для опису поняття **дата**. Реалізувати функції:
int checkDate2000(DATE d) – перевіряє чи дата коректна, тобто чи назва дня відповідає даті, дата лише в рамках 3 тисячоліття 2000 < дата < 3000.

Задача 11. Ввести структуру DATE (з полями **число, місяць, рік**) для опису поняття **дата**. Реалізувати функції:
DATE addDays(DATE d, int days) – обчислює дату через задану кількість днів, при цьому -1000 < days < 1000. Якщо введено некоректні параметри, то повернути структуру із всіма полями заповненими -1.
int diffDays(DATE start, DATE end) – визначає різницю між датами в днях, при цьому start <= end, різниця між введеними датами не більша за 100 років. Якщо введено некоректні параметри, то повернути -1.

Задача 12. Визначити структури, SPHERE та POINT, що описують **сферу** і **точку** в тривимірному просторі.
Скласти і протестувати функції:
int contains(SPHERE s, POINT p) – повертає 0, якщо точка знаходиться всередині сфери.
int cross(SPHERE a, SPHERE b) – повертає 0, якщо сфери перетинаються.

Задача 13. Увести структуру CAR для опису автомашини. Вона повинна мати такі поля: дату реєстрації (структура з полями - день, місяць, рік),
марку машини;
рік випуску;
колір;
vin номер.
Увести структуру CAR_STORE для зберігання списку машин.
Скласти і протестувати функції:
void insert(CAR_STORE store, CAR car) – додає машину в список,
CAR search(CAR_STORE store, char* vin) – шукає машину за номером і повертає як результат роботи функції.
CAR extract(CAR_STORE store, char* vin) – шукає машину за номером, видаляє і повертає як результат роботи функції.

Задача 14. Увести структуру CAR для опису автомашини. Вона повинна мати такі поля: дату реєстрації (структура з полями - день, місяць, рік),
марку машини;
рік випуску;
колір;
vin номер.
Увести структуру CAR_STORE для зберігання списку машин.
Скласти і протестувати функції:
void print(CAR_STORE store) – виводить список машин,
void insert(CAR_STORE store, CAR car) – додає машину в кінець списку,
CAR sort(CAR_STORE store) – сортує машини по номеру.

Задача 15. Увести структуру CAR для опису автомашини. Вона повинна мати такі поля: дату реєстрації (структура з полями - день, місяць, рік),
марку машини;
рік випуску;
колір;
vin номер.
Увести структуру CAR_STORE для зберігання списку машин.
Скласти і протестувати функції:
void insert(CAR_STORE store, CAR car) – додає машину в список,
CAR* search(CAR_STORE store, char* color) – шукає машини за кольором.
CAR* search(CAR_STORE store, int year) – шукає машини за роком.

Задача 16. Ввести переліковні типи ***Suit, Value***. З їхньою допомогою описати як структуру змінну ***CARD***.
Скласти і протестувати функцію:
int compare (CARD* hand1, CARD* hand2, CARD* river) – порівнює комбінації з 2 карт по правилам гри в Техаський Покер, з урахуванням 5 карт в river, повертає 0 – якщо вони рівні, -1 – якщо hand1 > hand2, 1 hand1 < hand2.

Задача 17. Визначити структури POINT та POLAR для опису точки в декартових та полярних координатах.
Скласти і протестувати функції:
POINT convert(POLAR p) – конвертує полярну точку в звичайну,
POLAR convert(POINT p) – конвертує звичайну точку в полярну,
float distance(POLAR a, POLAR b) – знаходить відстань між точками,
float distance(POINT a, POINT b) – знаходить відстань між точками.

Задача 18. Увести структуру COMPLEX для опису поняття **комплексне** число.
Реалізувати функції:
COMPLEX add(COMPLEX a, COMPLEX b) – додавання чисел,
COMPLEX sub(COMPLEX a, COMPLEX b) – віднімання чисел,
Int compare(COMPLEX a, COMPLEX b) – порівняння чисел, повертає 0 – якщо рівні, -1 якщо a < b. 1 якщо a > b.

Задача 19. Увести структуру COMPLEX для опису поняття **комплексне** число.
Реалізувати функції:
COMPLEX mul(COMPLEX a, COMPLEX b) – множення чисел,
COMPLEX div(COMPLEX a, COMPLEX b) – ділення чисел,
COMPLEX pow(COMPLEX base, COMPLEX exp) – піднесення до степені.

Задача 20. Увести структуру COMPLEX для опису поняття **комплексне** число. Увести структуру SQUARE_EQ для опису квадратного рівняння з дійсними коефіцієнтами. Увести структуру SQUARE_RESULT для зберігання пари комплексних коренів квадратного рівняння.
Реалізувати функції:
SQUARE_RESULT solve(SQUARE_EQ eq) – для розв'язання квадратного рівняння і повернення результату як пари комплексних чисел.

Задача 21. Ввести структуру DATETIME для опису поняття **дата та час**. Увести структур EVENT, що описує якусь подію, що має початок і кінець. Події, що перетинаються в часу конфліктують.
Реалізувати функції:
void findConflicts(EVENT* events, EVENT* conflicts) – шукає в масиві events ты події які конфліктують і повертає їх в масиві conflicts.
void findFree(EVENT* events, EVENT* free), шукає в масиві неконфліктуючих подій events проміжки вільного часу і повертає їх в масив free.

Задача 22. Ввести структуру DATETIME для опису поняття **дата та час**. Увести структур EVENT, що описує якусь подію, що має початок і кінець. Події, що перетинаються в часу конфліктують.
Реалізувати функції:
void findFree(EVENT* events1, EVENT* events2, EVENT* free), шукає в масивах неконфліктуючих подій events1 та events1 (розклади двох людей) проміжки вільного часу, які є вільні для обох людей повертає їх в масив free. Час людини до першої події і після останньої події вважати зайнятим.

Задача 23. Ввести переліковні типи **vertical, horizontal** для позначення клітин шахової дошки. Описати структуру CELL для опису координат клітини на дошці.
Скласти і протестувати функцію:
void horseOptions (CELL position, CELL* options) – визначає на які клітинки може перейти кінь та повертає їх в масиві options.

Задача 24. Ввести переліковні типи ***Suit, Value***. З їхньою допомогою описати як структуру змінну ***CARD***.
Скласти і протестувати функцію:
CARD change (CARD* hand) – повертає одну з 5 карт, що передано в hand, яку варто обміняти, щоб отримати кращу комбінацію за правилами гри в Покер.

Питання для самоконтролю

1. Дайте визначення структури. Що називається ярликом структури?
2. Яке призначення операції крапки?
3. У чому розходження застосування typedef і struct при визначенні структури?
4. Як звернутися до елемента масиву, що входить у структуру? До структури, що є елементом масиву?
5. Що потрібно для включення структури в структуру великого розміру?
6. Яка область дії змінних і функцій програми?
7. Який об'єм ОП одержує змінна і які початкові значенні якщо це зовнішня або автоматична змінна?
8. Що визначає оголошення char *sh[7];, який об'єм ОП і які початкові значення набуває змінна sh?
9. Які значення одержують елементи масиву покажчиків рядків, що ініціалізує значеннями рядкових констант?
10. Що таке структура і як можна її оголосити?
11. Який об'єм ОП одержує структура, масив структур?
12. Як можна звернутися до полів структури, елементів масиву структур, якщо задане ім'я структури, елементу масиву структур, покажчики на структуру або покажчик на елемент масиву структур?
13. Поясніть призначення і правила застосування різних типів формату введення і виведення.
14. Коли і навіщо формується значення типу st.fio[12]='\0'?
15. Що може бути фактичним параметром, якщо формальний заданий в вигляді STUD * s?

Лабораторна робота №8.

Файли та робота з ними

Мета лабораторної роботи – Опанувати введення/виведення даних у файли, робота з текстовими та бінарними файлами.

Хід роботи :

Виконання індивідуального завдання

- Вибрати варіант завдання відповідно вашого номеру по списку в журналі групи.
- Написати програму, відповідно до запропонованого індивідуального варіанту.
- Перед написанням коду, провести декомпозицію.
- Намагайтесь отримати якнайбільшу кількість функцій у вашій програмі, що відповідають моделі проведеній декомпозиції.
- Якщо це можливо, то дані для роботи програми потрібно задавати константами, або генерувати за допомогою генератора випадкових чисел.
- Програма повинна вимагати від користувача вводу правильних параметрів і пропонувати ввести некоректні значення повторно.
- Для двох ваших функцій (одна з функцій має бути та яка згадується в завданні) написати unit-тести, використовуючи запропонований підхід з функцією assert.
- Для демонстрації роботи програми, після unit-тестів, реалізувати демонстраційний виклик функцій (лише тих, що згадуються в завданні) програми, роздрукувавши вхідні і вихідні данні.
- Дати відповідь на 3 контрольні питання починаючи з вашого номеру по списку

Вміст звіту

1. Титульна сторінка
2. Тема та мета роботи
3. Декомпозиція задачі.
4. Навести текст розробленої програми.
5. Рисунки з результатами роботи програми.
6. Висновок
7. Відповідь на контрольні питання.

Варіанти завдань

Задача 1. Логістична компанія має парк однотипного транспорту. Встановлено, що безпечним є перевезення до 12 тон вантажу однією машиною за рейс. При цьому, рентабельним є перевезення більше 4 тон. Данні про перевезення знаходяться у файлі voyage.csv у форматі CSV. Кожен рядок містить відомості про: машину, водія, рейс, вагу вантажу.
Вивести на екран імена водіїв та ті рейси де вони допустили перевантаження. Після цього вивести імена водіїв та ті рейси де вони допустили недовантаження.

Задача 2. Логістична компанія має парк однотипного транспорту. Встановлено, що безпечним є перевезення із середньою швидкістю 70 км.год. Данні про перевезення знаходяться у файлі voyage.csv у форматі CSV. Кожен рядок містить відомості про: машину, водія, рейс, вагу вантажу.
Вивести на екран імена водіїв та ті рейси де вони допустили перевищення швидкості. Після цього вивести імена водіїв та ті рейси де швидкісний режим виконувався найточніше.

Задача 3. Логістична компанія має парк транспорту різних типів. При цьому, для кожного типу встановлено максимально допустиму вагу вантажу. Данні про типи транспортних засобів знаходяться у файлі у форматі CSV cars.csv. Данні про перевезення знаходяться у файлі voyage.csv у форматі CSV. Кожен рядок містить відомості про: машину, тип транспортного засобу, водія, рейс, вагу вантажу.
Вивести на екран імена водіїв та ті рейси де вони допустили перевантаження.

Задача 4. Логістична компанія має парк транспорту різних типів. При цьому, для кожного типу встановлено максимально допустиму середню швидкість. Данні про типи транспортних засобів знаходяться у файлі у форматі CSV cars.csv. Данні про перевезення знаходяться у файлі у форматі CSV. Кожен рядок містить відомості про: машину, тип транспортного засобу, водія, рейс, вагу вантажу.
Вивести на екран імена водіїв та ті рейси де вони допустили перевищення швидкості.

Задача 5. Логістична компанія має парк транспорту різних типів. При цьому, для кожного типу встановлено максимально допустиму вагу вантажу. Данні про типи транспортних засобів знаходяться у файлі у форматі CSV cars.csv. Данні про перевезення знаходяться у файлі voyage.csv у форматі CSV. Кожен рядок містить відомості про: машину, тип транспортного засобу, водія, рейс, вагу вантажу.
Вивести на екран ті машини які більше експлуатувались з перевантаженням.

Задача 6. Створіть файл у форматі CSV input.csv, що містить відомості про абітурієнтів. Відомості про кожного абітурієнта — це прізвище, шифр спеціальності і оцінки, одержані на трьох вступних іспитах.
В окремому файлі speciality.csv зберігаються для кожної спеціальності прохідний бал і кількість місць.
Програма видає відсортований список зарахованих абітурієнтів для кожної спеціальності відсортований за спаданням по середньому балу.

Задача 7. Створіть файл у форматі CSV input.csv, що містить відомості про абітурієнтів. Відомості про кожного абітурієнта — це прізвище, шифр спеціальності і оцінки, одержані на трьох вступних іспитах.
В окремому файлі speciality.csv зберігаються для кожної спеціальності прохідний бал і кількість місць.
Програма видає список спеціальностей і кількість вільних місць для кожної спеціальності на яку не вдалось зарахувати абітурієнтів на всі вільні місця.

Задача 8. Створіть файл у форматі CSV input.csv, що містить відомості про абітурієнтів. Відомості про кожного абітурієнта — це прізвище, шифр спеціальності і оцінки, одержані на трьох вступних іспитах.
При запуску програма запитує в користувача для кожної спеціальності прохідний бал і кількість місць.
Програма видає відсортований список зарахованих абітурієнтів для кожної спеціальності відсортований за спаданням по середньому балу.
Коли користувач увів всі необхідні данні, програма зберігає список зарахованих абітурієнтів у файл у форматі CSV result.csv, при цьому групує записи по спеціальностям і сортує за спаданням по середньому балу.

Задача 9. Створіть файл у форматі CSV input.csv, що містить відомості про гірські вершини. Кожен рядок файлу містить назву гірської вершини і її висоту та час підйому. Використовуючи структуру для опису поняття ***вершина,*** одержати назву найвищої вершини за даними файлу і вивести на екран. Розрахувати статистичні дані про вершини і вивести на екран: найвищу, найнижчу, найближчу до середньої висоти, середню висоту, моду, медіану. Відсортувати вершини по часу підйому на них і записати у файл у форматі CSV result.csv.

Задача 10. Папка містить декілька файлів з у форматі CSV. У деяких файлах містяться відомості про деякі товари. Товари можна описати структурою ITEM з полями id, name, producerId, producerNumber, size, color. Потрібно визначити які файли в папці містять дані про товари, прочитати дані з файлів і записати всі дані в результуючий файл result.csv. Деякі дані у різних файлах дублюються. Тобто деякі товари можуть зберігатись під різним іменем. Потрібно визначити, товари які дублюються (ті в які мають

однакові поля producerId, producerNumber, size, color) і вилучити із результуючого списку і записати у файл final.csv.

Задача 11. У файл flight.csv у форматі CSV знаходиться інформація про розклад руху літаків (рейс, дата відправлення та прибуття, аеропорти відправлення та призначення). При запуску програми видає список аеропортів відправлення та список аеропортів призначення. Далі користувач вводить аеропорт відправлення та призначення. Вивести на екран один з варіантів списку рейсів якими користувач може здійснити подорож. Між прибуттям в аеропорт і наступним вильотом має бути запас часу хоча б 60хв.

Задача 12. У файл flight.csv у форматі CSV знаходиться інформація про розклад руху літаків (рейс, дата відправлення та прибуття, аеропорти відправлення та призначення). Рейс може складатись із кількох перельотів, у такому випадку, в файлі буде кілька записів з одним номером рейсу. Визначити аеропорт через який пролітає (цей аеропорт не є початковою або кінцевою точкою маршруту) найбільше рейсів.

Задача 13. У файл flight.csv у форматі CSV знаходиться інформація про розклад руху літаків (рейс, дата відправлення та прибуття, аеропорти відправлення та призначення). Рейс може складатись із кількох перельотів, у такому випадку, в файлі буде кілька записів з одним номером рейсу. Користувач вводить з клавіатури дату. Визначити аеропорт з якого злітає найбільше літаків цього дня.

Задача 14. У файл flight.csv у форматі CSV знаходиться інформація про розклад руху літаків (рейс, дата відправлення та прибуття, аеропорти відправлення та призначення). Рейс може складатись із кількох перельотів, у такому випадку, в файлі буде кілька записів з одним номером рейсу. Користувач вводить з клавіатури дату. Визначити аеропорт до якого сідає найбільше літаків цього дня.

Задача 15. У файл flight.csv у форматі CSV знаходиться інформація про розклад руху літаків (рейс, дата відправлення та прибуття, аеропорти відправлення та призначення). Рейс може складатись із кількох перельотів, у такому випадку, в файлі буде кілька записів з одним номером рейсу. Визначити найдовший рейс від першого відправлення до крайнього прибуття.

Задача 16. Створіть файл у форматі CSV input.csv, що містить відомості про гірські вершини. Кожен рядок файлу містить назву гірської вершини і її висоту та час підйому. Використовуючи структуру для опису поняття

вершина. Розрахувати статистичні дані про вершини і вивести на екран: найвищу, найнижчу, найближчу до середньої висоти, середню висоту, моду, медіану. Відсортувати вершини по висоті і записати у файл у форматі CSV result.csv.

Задача 17. У файлі store.csv у форматі CSV міститься інформація про запаси на складі фірми, яка збирає та продає комп'ютери. На складі знаходяться процесори, материнські плати, пам'ять, відеокарти, диски, корпуси. Щоб зібрати один ПК потрібно по одному елементу кожного типу. Всі комплектуючі сумісні між собою. Вивести на екран кількість ПК які можна зібрати із наявних компонентів. Вивести список комплектуючих які потрібно докупити, щоб зібрати ПК використавши всі запаси на складі.

Задача 18. У файлі store.csv у форматі CSV міститься інформація про запаси на складі фірми, яка збирає та продає комп'ютери. На складі знаходяться процесори, материнські плати, пам'ять, відеокарти, диски, корпуси. Щоб зібрати один ПК потрібно по одному елементу кожного типу. Вивести на екран найдорожчу та найдешевшу конфігурацію ПК який можна зібрати із наявних компонентів.

Задача 19. Дано текстовий файл F1. Переписати його вміст у файл F2, розбивши на рядки таким чином, щоб кожен рядок, або містив не більше 20 символів, якщо це можливо додати пробіли між словами рядка, щоб текст був вирівняний по ширині рядка.

Задача 20. Написати програму порівняння рядків із двох файлів. У третій файл записати вміст наступним чином:
- якщо рядки з відповідним номером ідентичні то записати цей рядок у вихідний файл
- якщо рядки різні, то записати спочатку рядок з першого файлу і закінчити його символом [+] , а потім рядок з другого файлу і закінчити його символом [-]
- якщо рядок присутній лише в одному з файлів, а в іншому немає, бо він коротший, записати цей рядок у вихідний файл і закінчити його символом [+] якщо це рядок з першого файлу, і [-] якщо з другого.

Задача 21. Створити програму, що у текстовому файлі шукає імена і друкує на екран у вигляді списку. Іменем вважається послідовність із двох або трьох слів, що починаються з великої літери. Всі інші літери маленькі. При цьому, якщо маємо послідовність більше трьох слів, що починаються з великої літери, то компоненти такої послідовності не вважаються іменами.

Задача 22. Створити файл, що містить відомості про гравців хокейних команд певної молодіжної ліги (мінімум 8 команд).
Структура записів у файлі:

прізвище, ім'я гравця, роль, команда;
число закинутих шайб;
число зроблених результативних передач.

За даними, що вибираються з вхідного файлу, створити новий файл, що містить дані про шість найкращих гравців команд з яких варто створити молодіжну збірну.

Задача 23. Створити програму, що у текстовому файлі шукає телефонні номер і друкує на екран у вигляді списку у форматі 80667893445. Телефонні номери можуть бути у таких форматах: 380667893445, +3806678934456 80667893445, 0667893445

Задача 24. Створити програму, що у текстовому файлі шукає адреси електронної пошти і друкує на екран у вигляді списку у форматі.

Питання для самоконтролю

1. Що таке файловий буфер? Як зв'язаний буфер з потоком даних?
2. Назвіть функцію для підготовки файлу до роботи. Що повертає ця функція?
3. Укажіть розходження між двійковим і текстовим режимами доступу до файлу.
4. Яку операцію С можна використовувати, щоб забезпечити точний збіг розміру блоку даних, що читаються з файлу (або записуваних у нього), з розмірами відповідної конструкції, у якій ви маєте намір зберігати ці блоки даних у пам'яті?
5. Укажіть розходження між послідовним і довільним доступом до файлу? Поясніть призначення покажчика позиції в цій схемі. Яка функція використовується для установки індикатора позиції в задане місце у файлі?
6. Що таке багато файлова програма?
7. Якими засобами можна реалізувати багато файлову програму?
8. Поясніть реалізацію багато файлової програми за допомогою директиви #include.
9. Поясніть процес формування і виконання файлу проекта.
10. Що таке логічний (фізичний) файл?
11. Поясніть правила введення із файлу рядкових і дійсних даних.
12. Поясніть правила форматного виведення даних символьного, рядкового і дійсного типів.

Додаткове завдання.

Комп’ютерна графіка

Мета – опанувати використання базових графічних примітивів та програмування комп’ютерної графіки.

Виконання індивідуального завдання

- Вибрати варіант завдання відповідно вашого номеру по списку в журналі групи.
- Написати програму, відповідно до запропонованого індивідуального варіанту.
- Перед написанням коду, провести декомпозицію.
- Намагайтесь отримати якнайбільшу кількість функцій у вашій програмі, що відповідають моделі проведеній декомпозиції.
- Якщо це можливо, то дані для роботи програми потрібно задавати константами, або генерувати за допомогою генератора випадкових чисел.
- Програма повинна вимагати від користувача вводу правильних параметрів і пропонувати ввести некоректні значення повторно.
- Для двох ваших функцій (одна з функцій має бути та яка згадується в завданні) написати unit-тести, використовуючи запропонований підхід з функцією assert.
- Для демонстрації роботи програми, після unit-тестів, реалізувати демонстраційний виклик функцій (лише тих, що згадуються в завданні) програми, роздрукувавши вхідні і вихідні данні.

Варіанти завдань

Задача 1. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати координати гострокутного трикутника і зобразити його. Вписати коло в трикутник. Товщина лінії кола має бути більшою за товщину лінії трикутника.
Додаткове завдання: реалізувати обертання зображення навколо центра кола.

Задача 2. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити чорним кольором. Згенерувати координати гострокутного трикутника і зобразити його. Описати коло навколо трикутника. Товщина лінії кола має бути більшою за товщину лінії трикутника.
Додаткове завдання: реалізувати обертання зображення навколо центра кола.

Задача 3. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати координати прямокутного трикутника і зобразити його. Вписати коло в трикутник. Товщина лінії кола має бути більшою за товщину лінії трикутника.
Додаткове завдання: реалізувати відображення зображення зліва-направо (horizontal image flip)

Задача 4. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити чорним кольором. Згенерувати координати прямокутного трикутника і зобразити його. Описати коло навколо трикутника. Товщина лінії кола має бути більшою за товщину лінії трикутника.
Додаткове завдання: реалізувати відображення зображення зверху-вниз (vertical image flip)

Задача 5. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати координати правильного шестикутника і зобразити його. Вписати коло в фігуру. Товщина лінії кола має бути більшою за товщину лінії фігури.
Додаткове завдання: реалізувати обертання зображення навколо центра кола.

Задача 6. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити чорним кольором. Згенерувати координати правильного шестикутника і зобразити його. Описати коло навколо фігури. Товщина лінії кола має бути більшою за товщину лінії фігури.
Додаткове завдання: реалізувати обертання зображення навколо центра кола.

Задача 7. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати координати правильного п’ятикутника і зобразити його. Вписати коло в фігуру. Товщина лінії кола має бути більшою за товщину лінії фігури.
Додаткове завдання: реалізувати відображення зображення зліва-направо (horizontal image flip)

Задача 8. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити чорним кольором. Згенерувати координати правильного п’ятикутника і зобразити його. Описати коло навколо фігури. Товщина лінії кола має бути більшою за товщину лінії фігури.

Додаткове завдання: реалізувати відображення зображення зверху-вниз (vertical image flip)

Задача 9. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати координати двох трикутників які перетинаються. Зафарбувати область перетину кольором який відрізняється від інших.
Додаткове завдання: реалізувати обертання зображення навколо центра вікна.

Задача 10. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати координати трьох кіл які перетинаються. Зафарбувати область перетину всіх фігур кольором який відрізняється від інших.
Додаткове завдання: реалізувати обертання зображення навколо центра вікна.

Задача 11. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати правильного п’ятикутника. З’єднати вершини лініями різного кольору. Зафарбувати область перетину всіх фігур кольором який відрізняється від інших.
Додаткове завдання: реалізувати відображення зображення зверху-вниз (vertical image flip)

Задача 12. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати правильного п’ятикутника. З’єднати вершини лініями. Зафарбувати малий п’ятикутник який утворився на перетині всіх ліній кольором який відрізняється від інших.
Додаткове завдання: реалізувати відображення зображення зверху-вниз (vertical image flip)

Задача 13. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати правильного п’ятикутника. З’єднати вершини лініями. Зафарбувати зірку яка утворилась перетином всіх ліній кольором який відрізняється від інших.
Додаткове завдання: реалізувати відображення зображення зліва-направо (horizontal image flip)

Задача 14. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати правильного п’ятикутника. З’єднати вершини лініями. Зафарбувати фігури які утворились всередині різними кольорами.

Додаткове завдання: реалізувати обертання зображення навколо центра вікна.

Задача 15. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати правильного шестикутника. З'єднати вершини лініями різного кольору. Зафарбувати область перетину всіх фігур кольором який відрізняється від інших.
Додаткове завдання: реалізувати обертання зображення навколо центра вікна.

Задача 16. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати правильного шестикутника. З'єднати вершини лініями. Зафарбувати малий п'ятикутник який утворився на перетині всіх ліній кольором який відрізняється від інших.
Додаткове завдання: реалізувати відображення зображення зліва-направо (horizontal image flip)

Задача 17. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати правильного шестикутника. З'єднати вершини лініями. Зафарбувати зірку яка утворилась перетином всіх ліній кольором який відрізняється від інших.
Додаткове завдання: реалізувати відображення зображення зверху-вниз (vertical image flip)

Задача 18. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. Згенерувати правильного шестикутника. З'єднати вершини лініями. Зафарбувати фігури які утворились всередині різними кольорами.
Додаткове завдання: реалізувати обертання зображення навколо центра вікна.

Задача 19. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. За допомогою геометричних фігур зобразити символ радіаційної небезпеки.
Додаткове завдання: реалізувати плавну зміну кольору елементів зображення.

Задача 20. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. За допомогою геометричних фігур зобразити символ електричної небезпеки.
Додаткове завдання: реалізувати плавну зміну кольору елементів зображення.

Задача 21. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. За допомогою геометричних фігур зобразити дорожній знак зупинка заборонена.
Додаткове завдання: реалізувати плавну зміну кольору елементів зображення та обертання.

Задача 22. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. За допомогою геометричних фігур зобразити дорожній знак головна дорога
Додаткове завдання: реалізувати плавну зміну кольору елементів зображення та обертання.

Задача 23. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. За допомогою геометричних фігур зобразити символ Інь-Янь.
Додаткове завдання: реалізувати плавну зміну кольору елементів зображення та обертання.

Задача 24. Створити вікно програми розміром не менше ніж 500 на 500 пікселів. Фон вікна залити білим кольором. За допомогою геометричних фігур зобразити символ Тризуб
Додаткове завдання: реалізувати обертання зображення навколо центра вікна.

Список використаної літератури

1. Kalin M. Modern C Up and Running: A Programmer’s Guide to Finding Fluency and Bypassing the Quirks 1st ed. Edition. Chicago : Apress, 2022. 380 с. ISBN 978-1484286753.
2. Seacord R. C. Effective C, 2nd Edition: An Introduction to Professional C Programming. San Francisco : No Starch Press, 2024. 312 с. ISBN 978-1718504127.
3. TIOBE Index for April 2025. *TIOBE Index*. 01.05.2025. URL: https://www.tiobe.com/tiobe-index/ (дата звернення: 07.05.2025).
4. C/C++ for Visual Studio Code. *Visual Studio Code*. 04.03.2025. URL: https://code.visualstudio.com/docs/languages/cpp (дата звернення: 07.05.2025).
5. Installing GCC. *GCC, the GNU Compiler Collection*. 01.03.2025. URL: https://gcc.gnu.org/install/ (дата звернення: 07.05.2025).
6. Bernstein D. Patterns for Beginning Programmers. Harrisonburg : James Madison University Pressbooks, 2024. 181 с.

Електронне навчальне видання

Олександр Олександрович Решетнік,
Олександр Миколайович Ткаченко

Методичні вказівки до виконання лабораторних робіт з дисципліни «Основи програмування» зі спеціальності «Інженерія програмного забезпечення»

Рукопис оформив О. О. Решетнік

Видається в авторській редакції

Оригінал-макет виготовила О. Кушнір

Підписано до видання 15.06.2025
Гарнітура Times New Roman.
Зам. № Р2022-071
Видавець та виготовлювач
Вінницький національний технічний університет,
Редакційно-видавничий відділ.
ВНТУ, ГНК, к. 114.
Хмельницьке шосе, 95,
м. Вінниця, 21021.
press.vntu.edu.ua;
Email: irvc.vntu@gmail.com
Свідоцтво суб'єкта видавничої справи
серія ДК No 3516 від 01.07.2009 р.