**МІНІСТЕРСТВО ВНУТРІШНІХ СПРАВ УКРАЇНИ**

**ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВНУТРІШНІХ СПРАВ**

**Кафедра кібербезпеки та DATA-технологій факультету № 6**

# МЕТОДИЧНІ МАТЕРІАЛИ ДО ПРАКТИЧНИХ ЗАНЯТЬ

з навчальної дисципліни **"Системне програмування"**
вибіркових компонент
**освітньої програми першого рівня вищої освіти**

**125 "Кібербезпека" (Безпека інформаційних та комунікаційних систем)**

**Харків 2023**

**ЗАТВЕРДЖЕНО**
Науково-методичною радою
Харківського національного
університету внутрішніх справ
Протокол від 29.09.23 № 8

**СХВАЛЕНО**
Вченою радою факультету №6
Протокол від 21.09.23 № 8

**ПОГОДЖЕНО**
Секцією науково-методичної ради
ХНУВС з технічних дисциплін
Протокол від 22.09.23 № 8

Розглянуто на засіданні кафедри кібербезпеки та DATA-технологій (протокол від 11.09.23 № 9)

**Розробник:**

старший викладач кафедри кібербезпеки та DATA-технологій ХНУВС Калякін С.В.

**Рецензенти:**

завідувач кафедри інформаційних управляючих систем ХНУРЕ, д.т.н., професор Петров К.Е.

доцент кафедри протидії кіберзлочинності факультету №4 ХНУВС, к.т.н., доцент Світличний В.А.

## 1. Розподіл часу навчальної дисципліни за темами

| Номер та назва навчальної теми | Кількість годин відведених на вивчення навчальної дисципліни | | | | | Вид контролю |
|---|---|---|---|---|---|---|
| | Всього | з них: | | | | |
| | | Лекції | Практичні заняття | Лабораторні заняття | Самостійна робота | |
| **Семестр №5** | | | | | | |
| Тема № 1. Особливості системного програмування в ОС Windows. | 6 | 2 | 0 | 0 | 4 | екза-мен |
| Тема № 2. Мова програмування C/C++. Типи даних, вирази й операції. | 16 | 4 | 4 | 0 | 8 | |
| Тема № 3. Препроцесор мови C/C++. | 10 | 2 | 0 | 0 | 8 | |
| Тема № 4. Управляючі структури мови C/C++. | 18 | 2 | 4 | 4 | 8 | |
| Тема № 5. Обробка строк. Unicode. | 14 | 2 | 0 | 4 | 8 | |
| Тема № 6. Робота з динамічною пам’ятью. | 14 | 2 | 4 | 0 | 8 | |
| Тема № 7. Динамічні структури даних. | 14 | 2 | 0 | 4 | 8 | |
| Тема № 8. Обробка файлів засобами мови C/C++. | 14 | 2 | 0 | 4 | 8 | |
| Тема № 9. Обробка файлів засобами Windows API. | 14 | 2 | 0 | 4 | 8 | |
| **Всього за семестр №5** | **120** | **20** | **12** | **20** | **68** | |
| **Всього за дисципліною** | **120** | **20** | **12** | **20** | **68** | |

## 2. Методичні вказівки до практичних занять

### Практичне заняття № 1

**Тема:** «Ознайомлення з інтегрованим середовищем Visual C ++»
**Кількість годин:** 4 год.

**Навчальна мета заняття**: Ознайомитись з особливостями використання інтегрованого середовища Visual C ++ для побудови проектів мовою C ++.

**Навчальні питання**
1. Освоєння інтегрованого середовища Visual C ++
2. Вивчення структури програми на мові C ++.
3. Отримання навичок в побудові проектів в середовищі Visual C ++.

**Матеріально-технічне забезпечення:** комп'ютерний клас із встановленим інтегрованим середовищем Visual C ++.

### План проведення заняття

**1. Загальні відомості**

Сучасні інтегровані засоби розробки застосувань Windows дозволяють автоматизувати процес створення програми. Для цього використовуються генератори додатків. Програміст відповідає на питання генератора додатків і визначає властивості додатка - чи підтримує воно багатовіконний режим, технологію OLE, тривимірні елементи управління, довідкову систему і т.д. Генератор додатків створить додаток, що відповідає вимогам, і надасть вихідні тексти. Користуючись ним як шаблоном, програміст зможе швидко розробляти свої додатки.

Подібні засоби автоматизованого створення додатків включені в компілятор Microsoft Visual C ++ і називаються AppWizard (майстер додатків). Заповнивши кілька діалогових панелей, можна вказати характеристики програми та отримати його тексти, забезпечені великими коментарями.

У середовищі Visual C ++ можна будувати різні типи проектів. Такі проекти після їх створення можна компілювати і запускати на виконання. Фірма Microsoft розробила спеціальний інструментарій, який полегшує і прискорює створення проектів в середовищі Visual C ++.

Розглянемо деякі типи проектів, які можна створювати за допомогою різних засобів (майстрів проектів) Microsoft Visual C ++.

Типи майстрів проектів:

MFC AppWizard (exe) - за допомогою майстра застосувань можна створити проект Windows-застосування, яке може мати однодокументний, багатодокументний або діалоговий інтерфейс. Однодокументне застосування може надавати користувачеві в будь-який момент часу працювати тільки з одним файлом. Багатодокументне застосування, навпаки, може одночасно

представляти декілька документів, кожен у власному вікні. Інтерфейс діалогового додатки є єдине діалогове вікно.

MFC AppWizard (dll) - цей майстер застосувань дозволяє створити структуру DLL, засновану на MFC. За допомогою нього можна визначити характеристики майбутньої DLL.

AppWizard ATL COM - це засіб дозволяє створити елемент управління ActiveX або сервер автоматизації, використовуючи нову бібліотеку шаблонів ActiveX (ActiveX Template Library - ATL). Опції цього майстра дають можливість вибрати активний сервер (DLL) або виконуваний зовнішній сервер (exe-файл).

Custom AppWizard - за допомогою цього засобу можна створювати власні майстри AppWizard. Призначений для користувача майстер може базуватися на стандартних майстрів для додатків MFC або DLL, а також на існуючих проектах або містити тільки определеямие розробником кроки.

DevStudio Add-in Wizard - майстер доповнень дозволяє створювати доповнення до Visual Studio. Бібліотека DLL розширень може підтримувати панелі інструментів і реагувати на події Visual Studio.

MFC ActiveX ControlWizard - майстер елементів управління реалізує процес створення проекту, що містить один або кілька елементів управління ActiveX, заснованих на елементах управління MFC.

Win32 Application - цей майстер дозволяє створити проект звичайного Window-застосування. Проект створюється незаповненим, файли з вихідним кодом в нього слід додавати вручну.

Win32 Console Application - майстер створення проекту консольного застосування. Консольна програма - це програма, яка виконується з командного рядку вікна DOS або Windows і не має графічного інтерфейсу (вікон). Проект консольного застосування створюється порожнім, припускаючи додавання файлів вихідного коду в нього вручну.

Win32 Dynamic-Link Library - створення порожнього проекту динамічно підключається бібліотеки. Установки компілятора і компоновщика будуть налаштовані на створення DLL. Вихідні файли слід додавати вручну.

Win32 Static Library - це засіб створює порожній проект, призначений для генерації статичної (об'єктної) бібліотеки. Файли з вихідним кодом в нього слід додавати вручну.

Створення нового проекту.

Перед написанням програми відкриємо інтегровану середу Visual C ++:

Пуск / Програми / Microsoft Visual Studio / Microsoft Visual C ++

Далі створимо проект. Для цього:

1) File> New ...

2) У вікні:

- виберіть тип Win32 Console Application;
- введіть ім'я проекту в текстовому полі Project Name;
- введіть (виберете за допомогою кнопки ...) ім'я каталогу розміщення файлів проекту в текстовому полі Location, наприклад: D: / Projects /.
- клацніть лівою кнопкою миші на кнопці ОК.

3) відкривається діалогове вікно Win32 Console Application - Stepl of 1 і в ньому:

- виберіть тип An empty project;
- клацніть на кнопці Finish.

4) Після клацання з'явиться вікно New Project, в якому клацніть на кнопці ОК.

Далі створимо файл:

1) File> New .... В результаті відкриється діалогове вікно New.

2) На вкладці Files:

• виберіть тип файлу (в даному випадку: C ++ Source File);
• в текстовому полі File Name введіть потрібне ім'я файлу;
• прапорець Add to project потрібно включити;
• клацніть на кнопці ОК.

Будь-яка програма на C ++ складається з функцій, одна з яких повинна мати ім'я main, яке вказує, що саме з неї починається виконання програми. Після круглих дужок в фігурних дужках {} записується тіло функції, тобто ті оператори, які потрібно виконати.

Будь-яка заготовка при написанні програми має вигляд:

```
#include <…>
#include <…>
 int main()
{
   оголошення змінних;
   введення вихідних даних;
   розрахунок результату;
   вивід результату;
   return 0;
}
```

**Приклад 1. Ввод и вивід через операторы С**

```
#include <stdio.h>
void main () {
 int f;
 fflush (stdin); //зброс буферу потока stdin
 scanf ("%d", &f);            //ввод цілого числа з
клавиатури
 printf ("\n f=%d",f);       //вивід цілого числа с
нової строки
}
```

**Приклад 2. Ввод и вивід через потоки С++**

```
#include <iostream>
using namespace std;
void main () {
 int f;
 cin >> f;   //ввод  з клавіатури
```

```
    cout << "\nf=" << f;    // вивід в поток, пов'язаний
з монітором
    }
```

**Приклад 3. Розмір типів даних в байтах.**

```
#include <iostream.h>
void main( void ){
 cout << " int = " << sizeof(int) << endl;
 cout << " short = " << sizeof(short) << endl;
 cout << " char = " << sizeof(char) << endl;
 cout << " float = " << sizeof(float) << endl;
 cout << " double = " << sizeof(double) << endl;
 cout << " long = " << sizeof(long) << endl;
 cout << " long double = " << sizeof(long double) <<
endl;
}
```

Далі компілюємо програму. Для цього натискаємо кнопку на панелі інструментів або комбінацію клавіш Ctrl + F7. У вікні виводу (внизу екрану) має вивестися повідомлення 0 error (s), 0 warning (s) (0 помилок, 0 попереджень). Якщо є помилки - звірте введену програму з оригіналом.

Для запуску програми натискаємо відповідну кнопку на панелі інструментів або комбінацію клавіш Ctrl + F5.

**2. Завдання:**

2.1. Створити новий проект консольного застосування в середовищі Visual C ++.

2.2. Реалізувати приклади 1-3 в середовищі Visual C ++ і перевірити їх на працездатність.

2.3. Розробити в середовищі Visual C ++ програму, яка буде виводити на екран Ваше прізвище, ім'я та по-батькові, в наступному рядку – дату Вашого народження і в останньому рядку – групу в якій Ви навчаєтесь. Кожна строчка повинна закінчуватись числом у дужках, яке показує скільки байт вона займає в пам'яті.

2.4. Проаналізувати допущені помилки (якщо вони були).

**3. Зміст звіту**

3.1. Тема та мета роботи.

3.2. Постановка завдання.

3.3. Текст програм.

3.4. Результати виконання програм.

3.5. Аналіз помилок.

3.6. Висновки по роботі.

## Практичне заняття № 2

**Тема:** «Програмування лінійних алгоритмів. Робота з відладчиком. »
**Кількість годин:** 4 год.

**Навчальна мета заняття**: Закріплення навичок роботи в інтегрованому середовищі Visual C ++, отримання навичок в реалізації лінійних алгоритмів мовою C ++

**Навчальні питання**
1. Вивчення функцій відладчіка інтегрованого середовища Visual C ++
2. Вивчення базових типів даних мови C ++.
3. Реалізація лінійних алгоритмів мовою C ++.

**Матеріально-технічне забезпечення:** комп'ютерний клас із встановленим інтегрованим середовищем Visual C ++.

## План проведення заняття

**1. Загальні відомості**

Якщо в програмі всі оператори виконуються послідовно, один за іншим, така програма називається лінійної. Розглянемо як приклад програму, яка обчислює результат по заданій формулі.

Приклад 1. Розрахунок за формулою

Написати програму, яка переводить температуру в градусах за Фаренгейтом в градуси Цельсія за формулою:

C = 5/9 (F-32),

де C - температура за Цельсієм, а F - температура за Фаренгейтом.

Перед написанням будь-якої програми треба чітко визначити, що в неї потрібно ввести і що ми повинні отримати в результаті.

В даному випадку:

- в якості вихідних даних виступає одне дійсне число, яке представляє собою температуру за Цельсієм,
- як результат - інше дійсне число.

Перед написанням програми відкриємо інтегровану середу Visual C ++:

Пуск / Програми / Microsoft Visual Studio / Microsoft Visual C ++

Далі створимо новий проект консольного застосування (див. Заняття 1).

Набираємо наступний текст програми:

```
#include <iostream.h>
int main()                              //1
{
  float fahr, cels;                                 //2
  cout<<endl<<" Введите температуру По Фаренгейту"<<endl;    //3
  cin>>fahr;                                //4
  cels=5/9 * (fahr - 32);                           //5
  cout<<" По Фаренгейту: "<<fahr<<", в градусах Цельсия: "<<cels<<endl;
  return 0;                                    // 7
}
```

Розглянемо кожен рядок програми окремо.

На початку програми записана директива препроцесора, по якій до початкового тексту програми підключається заголовки <iostream>. Це файл, який містить опису операторів введення-виведення cin і cout.

Для зберігання вихідних даних і результатів треба виділити достатньо місця в оперативній пам'яті. Для цього служить оператор 2. У нашій програмі потрібно зберігати два значення: температуру за Цельсієм і температуру за Фаренгейтом, тому в операторі визначаються дві змінні. Одна, для зберігання температури за Фаренгейтом, названа fahr, інша (за Цельсієм) - cels. Імена змінних дає програміст, виходячи з їх призначення. Ім'я може складатися тільки з латинських букв, цифр і знака підкреслення і повинно починатися з цифри.

Для того, щоб користувач програми знав, в який момент потрібно ввести з клавіатури дані, застосовується так зване запрошення до вводу (оператор 3). На екран виводиться зазначена в операторі cout рядок символів, і курсор переводиться на наступний рядок. Для переходу на наступний рядок використовується endl.

В операторі 4 Ви набираєте текст, клавіатури одного числа в змінну fahr. Для цього використовується стандартний об'єкт cin і операція витягання (читання) >>. Якщо потрібно ввести декілька величин, використовується ланцюжок операцій >>.

В операторі 5 обчислюється вираз, записане праворуч від операції присвоювання (позначається знаком =), і результат присвоюється змінної cels, тобто заноситься в відведену цієї змінної пам'ять. Сначала ціла константа 5 буде поділена на цілу константу 9, потім результат цієї операції помножений на результат віднімання числа 32 з змінної fahr.

Для виведення результату в операторі 6 застосовується об'єкт cout. Виводиться ланцюжок, що складається з п'яти елементів. Це рядок "За Фаренгейтом:", значення змінної fahr, рядок ", в градусах Цельсія:", значення змінної cels і оператор переходу на новий рядок endl.

Останній оператор (оператор 7) цієї програми призначений для повернення з неї і передачі значення в зовнішнє середовище.

Далі компілюємо програму. Для цього натискаємо кнопку на панелі інструментів або комбінацію клавіш Ctrl + F7. У вікні виводу (внизу екрану) має вивестися повідомлення 0 error (s), 0 warning (s) (0 помилок, 0 попереджень). Якщо є помилки - звірте з оригіналом.

Для запуску програми натискаємо кнопку на панелі інструментів або комбінацію клавіш Ctrl + F5.

Як ви можете бачити, результат виконання програми зі стабільністю виявляється рівним нулю! Це відбувається через спосіб обчислення виразу. Давайте знову звернемося до оператора 4. Константи 5 і 9 мають цілий тип, тому результат їх розподілу також цілочисельний. Природно, що результат подальших обчислень не може бути нічим, крім нуля. Виправити цю помилку просто - достатньо записати хоча б одну з констант у вигляді дійсного числа, наприклад:

```
cels = 5.0 / 9 * (fahr - 32);
```

Переконайтесь, що програма працює без помилок, якщо це не так самостійно знайдіть помилки і виправте їх.

**2. Завдання** обираються за останньою цифрою номеру в журналі (якщо 0 – обираємо 10):

2.1. Написати програму перерахунку з миль в кілометри (1 миля = 1609.344 м).

2.2. Написати програму перерахунку з доларів в гривню.

2.3. Написати програму перерахунку відстані з верст у кілометри (1 верста = 1066.8 м).

2.4. Написати програму розрахунку напруги в електричному ланцюзі по відомим силі струму і опору.

2.5. Написати програму розрахунку площі круга.

2.6. Написати програму розрахунку довжини кола.

2.7. Написати програму обчислення обсягу куба.

2.8. Написати програму обчислення обсягу циліндра.

2.9. Написати програму, яка обчислює швидкість, з якою бігун пробіг задану дистанцію.

2.10. Написати програму перерахунку ваги з фунтів в кілограми (1 фунт = 405,9 грама)

**3. Зміст звіту**

3.1. Тема та мета роботи.

3.2. Постановка завдання.

3.3. Схеми алгоритму програм.

3.4. Текст програм.

3.5. Результати виконання програм.

3.6. Аналіз помилок.

3.7. Висновки по роботі.

## Практичне заняття № 3

**Тема: «Робота з динамічною пам’ятью»**
**Кількість годин:** 4 год.

**Навчальна мета заняття**: Отримання практичних навичок в роботі з динамічною пам’ятью.

**Навчальні питання**
1.1 Закріплення знань про масиви.
1.2 Закріплення знань про покажчики.
1.3 Закріплення знань про алгоритми упорядкування.

**Матеріально-технічне забезпечення:** комп’ютерний клас із встановленим інтегрованим середовищем Visual C ++.

## План проведення заняття

### 1. Загальні відомості про масиви

Масив - це сукупність змінних одного типу, до яких звертаються за допомогою загального імені. Доступ до окремого елементу масиву може здійснюватися за допомогою індексу. У мові С все масиви складаються з дотичних ділянок пам'яті. Найменший адреса відповідає першому елементу, найбільший адреса відповідає останньому елементу. Масиви можуть мати одну або кілька розмірностей.

### 1.1. Одновимірні масиви

Стандартний тип оголошення одновимірного масиву наступний:

```
тип ім'я_змінної [розмір];
```

В С масиви повинні визначатися однозначно, щоб компілятор міг виділити для них місце в пам'яті. Тут тип оголошує базовий тип масиву і є типом кожного елемента масиву. Параметр розмір визначає, скільки елементів містить масив. В одновимірному масиві стандартний розмір масиву в байтах обчислюється таким чином:

загальне число байт = sizeof (базовий тип) * число елементів;

У всіх масивів перший елемент має індекс 0. Тому, якщо написати

Char p [10];

то буде оголошений масив символів з 10 елементів, причому ці елементи адресуються індексом від 0 до 9. Наступна програма вводить цілочисельний масив з клавіатури і виводить його на дисплей:

```
#include <stdio.h>
main ()
{Const int n = 10;
int x [n]; / * Резервує місце для 10 цілочисельних елементів *
/
int t;
for (t = 0; t <n; ++ t)
```

```
{
printf ( "Введіть% d-й елемент масиву:", t + 1);
scanf ( "% d", & x [t]);
}
for (t = 0; t <n; ++ t)
printf ( "% d", x [t];
}
```

У мові С відсутня перевірка меж масивів. Можна вийти за один кінець масиву і записати значення в будь-яку змінну, що не відноситься до масиву, або навіть в код програми. Робота з перевірки меж масиву покладається на програміста. Наприклад, слід переконатися, що масив символів, куди здійснюється введення, має достатню довжину для прийняття найдовшої послідовності символів.

Одномірні масиви - це насправді списки інформації одного типу.

**1.2. Створення покажчика на масив**

Можна створити покажчик на перший елемент масиву, вказавши ім'я масиву без індексу. нехай є

int sample [10];

Можна створити покажчик на перший елемент масиву, використовуючи ім'я sample. Отже, наступний фрагмент привласнює змінної p адреса першого елемента масиву sample:

```
int * p;
int sample [10];
p = sample;
```

Можна також отримати адресу першого елемента масиву за допомогою оператора &. наприклад:

```
p = sample;
i
p = & sample [0];
```

Призводять до однакового результату. Проте запис & sample [0] в професійних програмах практично не зустрічається.

Індексація за допомогою покажчиків

Покажчики та масиви тісно пов'язані між собою. Як пояснювалося вище, ім'я масиву без індексу - це покажчик на перший елемент масиву. Нехай є масив

```
char p [100];
```

вираз

```
p = = & p [0];
```

видає істину, оскільки адреса першого елемента і адреса масиву збігаються.

Справедливо і зворотне. Будь-який покажчик може бути проіндексований, як ніби це масив. наприклад:

```
int * p, i [10];
p = I;
p [5] = 100; / * Привласнення за допомогою індексу * /
* (P + 5) = 100; / * Привласнення за допомогою покажчика * /
```

Обидва оператора присвоювання поміщають значення 100 в шостий елемент масиву i. Перший оператор використовує індексацію з p, а другий - арифметику покажчиків. Так чи інакше, результат однаковий.

### 1.3. Сортування

Розглянемо масив цілих або дійсних чисел $a_1$, $a_2$, ..., $a_n$. Нехай потрібно переставити елементи цього масиву так, щоб після перестановки вони були впорядковані за зростанням $a_1 \le a_2 \le ... \le a_n$ або за спаданням $a_1 \ge a_2 \ge ... \ge a_n$. Це завдання називається завданням сортування або впорядкування масиву. Для вирішення цього завдання можна скористатися, наприклад, такими алгоритмами:

а) Знайти елемент масиву, що має найменше (найбільше) значення, переставити його з першим елементом. Потім виконати те ж саме, почавши з другого елементу і так далі. (Сортування вибором)

б) Послідовним переглядом чисел $a_1$, $a_2$, ..., $a_n$ знайти найменше і таке, що $a_i > a_{i+1}$ або ai $<a_{i+1}$. Поміняти $a_i$ і $a_{i+1}$ місцями, відновити перегляд з елемента $a_{i+1}$ і так далі. Тим самим саме найбільше або найменше число пересунеться на останнє місце. Наступні перегляди слід починати знову спочатку, зменшуючи на одиницю кількість переглядаються елементів. Масив буде впорядкований після перегляду, в якому брали участь тільки перший і другий елементи. (Сортування обмінами)

в) Переглядати послідовно $a_2$, ..., $a_n$ і кожен новий елемент вставляти на відповідне місце в уже впорядковану послідовність $a_1$, ..., $a_{i-1}$. Це місце визначається послідовним порівнянням $a_i$ з впорядкованими елементами $a_1$, ..., $a_{i-1}$. (Сортування простими вставками)

г) Порівняти елементи $a_1$ і $a_2$ і, якщо $a_1 > a_2$ (або $a_1 < a_2$), то ці елементи переставити. Далі порівняти елементи $a_2$ і $a_3$ і, якщо $a_2 > a_3$ (або $a_2 < a_3$), то їх переставити. Далі порівняти елементи $a_3$ і $a_4$ і так далі до елементів $a_{n-1}$ і $a_n$ включно. Далі ці дії повторити, починаючи знову з першого елемента. Останнім є контрольний прохід, при якому не буде перестановок елементів. (Сортування за методом бульбашки)

Коли масив використовується в якості аргументу функції, передається тільки адреса масиву, а не копія всього масиву. При виконанні функції з ім'ям масиву в функцію передається покажчик на перший елемент масиву. Параметр повинен мати тип, сумісний з покажчиком. Т. до передається адреса на початок масиву, то інформація про розмірності втрачається, і її необхідно передавати окремим параметром.

```
#include <iostream>
#include <time>
#include <stdlib>
void display (float *, int);
float minim (float *, int);

int main (void)
{
   float t [10];
int i;
```

```
   srand (time (NULL));
   for (i = 0; i <10; i ++)
t [i] = 20. * rand () / RAND_MAX-10;
   display (t, 10);
   cout << "min =" << minim (t, 10) << endl;
   return 0;
}
// функція виведення на екран масиву
void display (float * a, int k)
{
   int j;
   for (j = 0; j <k; j ++)
cout << a [j] << '\ t';
cout << endl;
}
// функція пошуку мінімального елемента масиву
float minim (float * a, int k)
{
int i;
float min = a [0];
for (i = 0; i <k; i ++)
{
if (a [i] <min)
min = a [i];
}
return min;
}
```

Оскільки масив передається з використанням покажчиків, це означає, що масив передається за адресою, тобто при зміні елементів масиву в функції, змінюються значення елементів в викликає функції.

Тому в попередній задачі введення елементів масиву можна оформити також в окремій функції, наприклад

```
// функція введення елементів масиву
void input (float * a, int k)
{
   int j;
   for (j = 0; j <k; j ++)
cin >> a [j];
}
```

У C ++ є потужний засіб параметризації - шаблони. За допомогою шаблону функції можна визначити алгоритм, який буде застосовуватися до даних різних типів, а конкретний тип даних передається функції у вигляді параметра на етапі компіляції. Компілятор автоматично генерує правильний код, відповідний переданому типу. Таким чином, створюється функція, яка автоматично перевантажує сама себе і при цьому не містить накладних витрат, пов'язаних з параметризацією.

Формат найпростішої функції-шаблону:

```
template <class Type> заголовок
{
/ * Тіло функції * /
}
```

Замість слова Type може використовуватися довільне ім'я.

Наприклад, функція -шаблон для пошуку мінімального елемента масиву

```
template <c1ass Type> minim (Type * a, int k)
{
int i;
Type min = a [0];
for (i = 0; i <k; i ++)
{
if (a [i] <min)
min = a [i];
}
return min;
}
```

**2. Постановка завдання**

Для конкретного варіанту (обирається за останньою цифрою номеру за журналом) ввести масив вихідних даних і виконати над ним зазначені дії. Вивчивши алгоритми впорядкування, вибрати один з них. Написати програму, яка працює з будь-яким набором даних. Вхідну інформацію і результати рахунки вивести на друк, забезпечивши їх відповідними заголовками. Кожен пункт завдання реалізується окремою функцією, в яку масив передається як параметр. Обов’язково використати арифметику покажчиків при обробці масивів (хочаб в одній функції).

**Варіант 1**

В одновимірному масиві, що складається з n дійсних елементів, обчислити:

1) суму негативних елементів масиву;

2) суму елементів масиву, розташованих між максимальним і мінімальним елементами.

3) Впорядкувати елементи масиву по зростанню.

**Варіант 2**

В одновимірному масиві, що складається з n дійсних елементів, обчислити:

1) суму позитивних елементів масиву;

2) суму елементів масиву, розташованих між максимальним по модулю і мінімальним за модулем елементами.

3) Впорядкувати елементи масиву по спаданню.

**Варіант 3**

В одновимірному масиві, що складається з n цілих елементів, обчислити:

1) суму елементів масиву з парними номерами;

2) суму елементів масиву, розташованих між першим і останнім нульовими елементами.

3) Перетворити масив таким чином, щоб спочатку розташовувалися всі позитивні елементи, а потім - все негативні (елементи, рівні 0, вважати позитивними).

**Варіант 4**

В одновимірному масиві, що складається з n дійсних елементів, обчислити:

1) суму елементів масиву з непарними номерами;

2) суму елементів масиву, розташованих між першим і останнім негативними елементами.

3) Стиснути масив, видаливши з нього всі елементи, модуль яких не перевищує 1. Вивільнені в кінці масиву елементи заповнити нулями.

**Варіант 5**

В одновимірному масиві, що складається з n дійсних елементів, обчислити:

1) максимальний елемент масиву;

2) суму елементів масиву, розташованих до останнього позитивного елемента.

3) Стиснути масив, видаливши з нього всі елементи, модуль яких знаходиться в інтервалі [a, b]. Вивільнені в кінці масиву елементи заповнити нулями.

**Варіант 6**

В одновимірному масиві, що складається з n дійсних елементів, обчислити:

1) мінімальний елемент масиву;

2) суму елементів масиву, розташованих між першим і останнім позитивними елементами.

3) Перетворити масив таким чином, щоб спочатку розташовувалися всі елементи, рівні нулю, а потім - всі інші.

**Варіант 7**

В одновимірному масиві, що складається з n цілих елементів, обчислити:

1) номер максимального елемента масиву;

2) суму елементів масиву, розташованих між першим і другим нульовими елементами.

3) Перетворити масив таким чином, щоб в першій його половині розташовувалися елементи, що стояли в непарних позиціях, а в другій половині - елементи, що стояли в парних позиціях.

**Варіант 8**

В одновимірному масиві, що складається з n дійсних елементів, обчислити:

1) номер мінімального елемента масиву;

2) суму елементів масиву, розташованих між першим і другим негативними елементами.

3) Перетворити масив таким чином, щоб спочатку розташовувалися всі елементи, модуль яких не перевищує 1, а потім - всі інші.

**Варіант 9**

В одновимірному масиві, що складається з n дійсних елементів, обчислити:

1) максимальний по модулю елемент масиву;

2) суму елементів масиву, розташованих між першим і другим позитивними елементами.

3) Перетворити масив таким чином, щоб елементи, рівні нулю, розташовувалися після всіх інших.

**Варіант 10**

В одновимірному масиві, що складається з n цілих елементів, обчислити:

1) мінімальний за модулем елемент масиву;

2) суму модулів елементів масиву, розташованих після першого елемента, рівного нулю.

3) Перетворити масив таким чином, щоб в першій його половині розташовувалися елементи, що стояли в парних позиціях, а в другій половині - елементи, що стояли в непарних позиціях.

**3. Зміст звіту**

3.1. Тема та мета роботи.
3.2. Постановка завдання.
3.3. Текст програм.
3.4. Результати виконання програм.
3.5. Аналіз помилок.
3.6. Висновки по роботі.

## 3. Рекомендована література (основна, допоміжна), інформаційні ресурси в Інтернеті

**Основна**


1. Галісєєв Г.В. Системне програмування // Видавництво Університету “Україна”, 2018. – 253 с.
2. С.В. Єфіменко, О.В. Сугакова. Програмування: мови С і С++. - К.: ВПЦ "Київський університет", 2006 р.


**Допоміжна**


3. Шпак З. Я. Програмування мовою С.- Львів: Видавництво Львівська політехніка, 2011. – 436 с.
4. Ivor Horton’s Beginning Visual C++. - Wiley Publishing, Inc., 2010 – 1272 p.


**Інформаційні ресурси**

1. http://cppreference.com/
2. http://www.learncpp.com/
3. https://www.fluentcpp.com/