Міністерство освіти і науки України
Чернігівський національний технологічний університет

І Н Ф О Р М А Т И К А

Методичні вказівки

до виконання лабораторних робіт для студентів галузі знань
0908 ”Електроніка” напрямку підготовки 6.090802 “Електронні пристрої та системи”

Мова програмування “С++”
та об'єктно-орієнтоване програмування

ЧЕРНІГІВ ЧНТУ 2014

Міністерство освіти і науки України
Чернігівський національний технологічний університет

І Н Ф О Р М А Т И К А

Методичні вказівки

до виконання лабораторних робіт для студентів галузі знань 0908 ”Електроніка” напрямку підготовки 6.090802 “Електронні пристрої та системи”

Мова програмування “С++”
та об'єктно-орієнтоване програмування

Затверджено
на засіданні кафедри
промислової електроніки
протокол №__ від __.__.2014р.

ЧЕРНІГІВ ЧНТУ 2014

Інформатика. Методичні вказівки до виконання лабораторних робіт для студентів галузі знань 0908 ”Електроніка” напрямку підготовки 6.090802 “Електронні пристрої та системи”. Мова програмування “С++” та об'єктно-орієнтоване програмування/ Укл. Ревко А.С. –Чернігів: ЧНТУ, 2014. – 58 с.

Укладачі: Ревко Анатолій Сергійович, кандидат технічних наук, доцент

Відповідальний за випуск: Денисов Ю. О., завідувач кафедри промислової електроніки, доктор технічних наук, професор

Рецензент: Бивойно П. Г., кандидат технічних наук, доцент Чернігівського державного технологічного університету

ЗМІСТ

ВСТУП

Ці методичні вказівки призначені для студентів першого курсу галузі знань “Електроніка” (0908) напрямку підготовки “Електронні пристрої та системи” (6.090802) денної та заочної форм навчання та слугують для допомоги студентам під час підготовки та виконання лабораторних робіт з дисципліни “Інформатика” у другому семестрі.

Метою проведення лабораторних робіт є опанування мови програмування "С++" та об’єктно-орієнтованого програмування(ООП), вивчення та освоєння основних методів та прийомів програмування, отримання практичних навичок роботи за комп’ютером по відлагодженню та тестуванню програм.

Цикл лабораторних робіт виконується протягом семестру і охоплює основні теми курсу "Інформатика" другого семестру. Теоретичною основою для виконання лабораторних робіт є курс лекцій, теоретичні відомості на початку кожної лабораторної роботи даних методичних вказівок та навчальна література.

Ефективність мови "С++" в самих різноманітних прикладних програмах, від вирішення невеликих задач чисельного характеру до розробки складних програмних систем – компіляторів, баз даних, операційних систем, програмування мікроконтролерів т.ін. обумовила її широке розповсюдження та популярність. Мова відобразила найважливіші концепції технології розробки програм: розвинута система типів даних, орієнтація на ООП. Основною метою лабораторних робіт є опанування мови програмування С++ та ООП, вивчення та освоєння основних методів та прийомів програмування цією мовою, отримання практичних навичок роботи за комп'ютером по відлагодженню та тестуванню програм.

Описи лабораторних робіт виконані по єдиній структурі та включають до себе мету роботи, короткі теоретичні відомості, методичні вказівки для підготування до роботи та її виконання, індивідуальні завдання до лабораторної роботи, контрольні запитання для підготування до захисту роботи. Тематика лабораторних робіт охоплює практично всі основні розділи мови "С++".

Для допущення до виконання лабораторної роботи студенту необхідно підготувати заздалегідь план виконання роботи, який повинен включати:

1) Номер лабораторної роботи.

2) Тему лабораторної роботи.

4) Мету лабораторної роботи.

4) Короткі теоретичні відомості.

5) Методичні вказівки.

6) Завдання для лабораторної роботи свого варіанту з відповідними вказівками.

7) Блок-схеми програм.

Після виконання лабораторної роботи для її захисту необхідно скласти звіт, який в себе включає: пункти 1-7 плану роботи, текст програми, візуальний

інтерфейс (вікно) програми, результати виконання вказівок та тестів по програмі, висновок по роботі.

Лабораторна робота захищається на лабораторному занятті або на консультації (за дозволом викладача). Студент, що не захистив дві попередні роботи не допускається до наступної. Також до виконання лабораторної роботи не допускається студент, який не має плану проведення поточної лабораторної роботи.

Після виконання лабораторних робіт студент складає звіт по циклу лабораторних робіт, який містить титульний лист, зміст та всі звіти по кожній лабораторній роботі. Приклад титульного листа звіту приведений у додатку Б.

Під час захисту лабораторної роботи студент повинен відповісти на одне з питань, які приведені в кінці кожної роботи в методичних вказівках, а також вирішити просту задачу по темі роботи.

Після успішного виконання та захисту всіх лабораторних робіт, а також здавання звіту по циклу лабораторних робіт викладачу, студент допускається до складання іспиту чи заліку.

Для підготування до виконання та захисту лабораторних робіт можна використовувати конспект лекцій, ці методичні вказівки, іншу літературу з мови програмування "C++". Список рекомендованої літератури наведений в кінці методичних вказівок.

1 ЛАБОРАТОРНА РОБОТА №1. ПРОГРАМУВАННЯ ПРОСТИХ ЗАДАЧ У ВІЗУАЛЬНОМУ РЕЖИМІ

Мета роботи: отримання навичок з програмування простих програм у візуальному режимі мовою С++ з використанням ООП, навичок по введенню – виводу у візуальному режимі.

1.1 Теоретичні відомості

Відмінності С++ від С

- у С++ ключове слово *void* не обов'язкове (еквівалентно *int m(); і int m(void)*);
- у С++ всі функції повинні мати прототипи;
- якщо у С++ функція повертає тип, відмінний від *void*, то оператор *return* повинен мати значення відповідного типу;
- у С++ можна вибирати місце для оголошення локальних змінних не тільки на початку блоку;
- у С++ введення-виведення може здійснюватися не тільки за допомогою функцій, але й за допомогою операцій.(<<, >>);
- логічний тип даних ***bool***.

Об'єкти типу **bool** можуть приймати тільки два значення: ***true*** і ***false***. Бульові змінні автоматично перетворюються в цілі числа і навпаки. А саме, всі ненульові значення перетворюються в значення ***true***, а нуль — в значення ***false***. Зворотне перетворення: значення ***true*** перетворюється в одиницю, а ***false*** — у нуль.

Існують два стилі написання програм мовою С++: старий та новий. В ході розвитку мова С++ значно змінювалась. В результаті фактично існують дві версії мови С++. Перша з них — традиційна, розроблена Б'ярном Страуструпом. Друга версія розроблена Страуструпом і комітетом ANSI/ISO зі стандартизації. Вони дуже схожі, хоча друга версія має більші можливості. Якщо компілятор не підтримує новий стиль — можна писати в старому.

Принципова різниця між старим та новим стилями полягає у двох властивостях: новий стиль підключення бібліотек і використання оператору ***namespace***.

Оголошення у директивах *#include* нового стилю не завжди є іменами файлів, тому у них не має розширення ***.h***. Вони складаються лише з імені заголовку, що поміщене в кутові дужки. Єдина різниця між старим та новим стилем у заголовках є те, що заголовки нового стилю не обов'язково є іменами файлів.

Оскільки мова С++ спадкувала всі бібліотеки функцій мови С, то підтримуються і стандартні бібліотеки мови С. (Тобто файли *stdio.h*, *ctype.h* і т.д. доступні) Але стандарт С++, крім них, визначає і заголовки нового виду, котрі можуть замінювати собою реальні файли.

В програмах на мові С++ до імен стандартних файлів мови С додається префікс "c", а розширення "**.h**" відкидається.

Наприклад, файлу ***math.h*** в старому стилі відповідає заголовок ***<cmath>***, а файлу ***string.h*** — заголовок ***<cstring>***. Як правило, імена заголовків старого і нового стилю співпадають, за винятком розширення ***".h"***. Всі компілятори дозволяють використовувати заголовкові файли старого стилю. Однак цей стиль вважається застарілим, і його не рекомендується використовувати в нових програмах.

Простір імен (***namespace***) — це звичайна область видимості (область локалізації). Він призначений для локалізації імен ідентифікаторів і запобігає конфліктів між ними. Елементи, що оголошені в одному просторі імен, відокремлені від елементів, що належать іншому просторові. Якщо програма, що написана на мові С++, містить заголовок старого стилю, його вміст треба погрузити в глобальний простір імен: ***using namespace std;.*** Це дозволяє компілятору С++ компілювати програми, що написані на С.

Приклад одної і тої самої простої програми, що написані у старому та новому стилях програмування:

```
// Програма на С++ у старому стилі
#include <iostream.h>
int main()
{
return 0;
}

// Новий стиль програми мовою С++
#include <iostream>
using namespace std;
int main()
{
return 0;
}
```

Оператори введення-виведення ***<<, >>.*** Для роботи з цими операторами необхідно підключати заголовковий файл *<iostream>*. Відбувається робота з потоками введення-виведення.

Потік (stream) — це логічний пристрій, що отримує чи передає інформацію. Потік пов'язаний з фізичним пристроєм введення-виведення. Всі потоки функціонують однаково, хоча фізичні пристрої, з якими вони пов'язані, можуть бути різними. Оскільки всі потоки однакові, то одна функція введення-виведення може працювати з різними типами фізичних пристроїв. Наприклад, за допомогою одної функції можна виводити дані як на принтер, так і на екран чи до файлу.

У таблиці 1.1 приведені стандартні вбудовані потоки у мові С++. Потоки ***cin***, ***cout*** і ***cerr*** відповідають потокам ***stdin***, ***stdout*** и ***stderr***. Чотири додаткових потоки: ***win***, ***wout***, ***werr*** і ***wlog*** – версії потоків для введення-виведення розширених символів. Для них використовується тип ***wchar_t*** і 16-битові значення.

Приклад введення-виведення з використанням потоків C++ у консольному режимі:

```
#include <iostream>
using namespace std;      // Простір імен
int main()
{
  int i;
  cout << "Це - виведення. \n"; // Однорядковий коментар
  /*Коментар в стилі мови C */
  // Введення числа  з допомогою оператору >>
  cout << "Введіть число: ";
  cin >> i; // Тепер виведемо число за допомогою оператора <<
  cout << i << " в квадраті дорівнює " << i*i << "\n";
  cin >> i; //для зупинки у консолі Qt
  return 0;
}
```

Таблиця 1.1 - Вбудовані потоки у C++

Потік	**Значення**	**Пристрій за замовченням**
cin	Стандартне введення	Клавіатура
cout	Стандартне виведення	Екран
cerr	Стандартне виведення помилок	Екран
clog	Буферізований варіант потоку cerr	Екран

Можливе введення–виведення і з форматуванням. Можна встановити ширину поля, вказати основу системи числення або кількість цифр після коми і т.д.

Для форматування використовуються два схожих, але різних способи.

1. Пряме звертання до членів класу ***ios***. (самостійно задавати різні прапорці форматування, що визначені у класі ***ios***, або викликати різноманітні функції-члени.

2. У виразах введення-виведення можна використовувати спеціальні функції, що називаються маніпуляторами (***manipulators***).

Кожний потік пов’язаний з набором прапорців формату, що керують представленням інформації. У класі ***ios*** є бітова маска ***fmtflags***, в якій є значення: ***adjustfield, basefield, boolalpha, dec, fixed, floatfield, hex, internal, left, oct, right, scientific, showbase, showpoint, showpos, skipws, unitbuf, uppercase.*** Ці значення використовуються для встановлення та скидання прапорців формату[11].

Прапорці форматування:

skipws - прогалинні символи ігноруються;

left - вирівнювання по лівому краю поля;

right - вирівнювання по правому краю поля;

internal - знак числа виводиться по лівому краю, число — по правому. Проміжок заповнюється символами fill (див. нижче);

dec - десяткова система числення;

oct - вісімкова система числення;

hex - шістнадцяткова система числення;

showbase - виводиться основа системи числення;

showpoint - виводиться десяткова крапка і дробова частина;

uppercase - виводяться символи верхнього регістру;

showpos - показує знак при виведенні додатних чисел;

scientific - дійсні числа у плаваючій формі;

fixed - дійсні числа у формі з фіксованою крапкою;

unitbuf - очищення буферів всіх потоків після кожного виведення;

stdio - очищення буферів потоків *stdout* і *stderr* після кожного виведення.

Для встановлення прапорці форматування використовується функція ***setf()***. Вона має наступний вигляд: ***fmtflags setf(****fmtflags прапорці****)***

Приклад:

```
#include <iostream>
using namespace std;
int main ()
{
cout.setf(ios::showpoint);
cout.setf(ios::showpos);
cout << 100.0; // Виводимо число +100.0
return 0;
}
```

Можна об’єднувати декілька прапорців в одній команді оператором «або»: ***cout.setf(ios::ishowpoint | ios::showpos);***

Функція ***unsetf().*** Скидає один чи декілька прапорців формату і має вигляд: ***void unsetf(****fmtflags прапорці****)***

Приклад**:**

```
#include <iostream>
using namespace std;
int main()
{
 cout.setf(ios::uppercase | ios::scientific);
 cout << 100.12; // Виводимо число 1.0012E+02
 cout.unsetf(ios::uppercase); // Скидаємо uppercase
 cout <<" \n" << 100.12; // Виводимо число 1.0012e+02
return 0;
}
```

Є функції встановлення ширина поля виведення ***width()***, точності ***precision()*** та символу-заповнювача ***fill()***. Без параметру – отримати значення, з параметром – встановити значення.

width(), width(x) – ширина поля виведення(загальна кількість символів);

precision(), ***precision(x)*** – точність(символи після коми);

fill(), fill(x) – символ-заповнювач.

Приклад:

```
#include <iostream>
using namespace std;
int main()
{
cout.precision(4); cout.width(10);
cout << 10.12345 << "\n"; // Виводить 10.12
cout.fill('*'); cout.width(12);
cout << 10.12345 << "\n"; // Виводить *******10.12
// Ширина поля виведення впливає і на рядки
cout.width(10);
cout << "Hi!" << "\n"; // Виводить *******Hi!
cout.width(20);
cout.setf(ios::left); // Вирівнювання по лівому краю
cout << 10.12345; // Виводить на екран 10.12**********
int i; cin >> i; return 0;
}
```

Використання маніпуляторів формату для форматування. Необхідно підключити заголовковий файл *<iomanip>.* Тут описані спеціальні функції, що називаються маніпуляторами (*manipulators*). Їх можна включати в оператори введення-виведення.

Стандартні маніпулятори: ***boolalpha***, ***dec***, ***endl***, ***ends***, ***fixed***, ***flush***, ***hex***, ***internal***, ***left***, ***noboolalpha***, ***noshowbase***, ***noshowpoint***, ***noshowpos***, ***noskipws***, ***nounitbuf***, ***nouppercase***, ***oct***, ***resetiosflags(fmtflags*** *f)*, ***right***, ***scientific***, ***setbase(int*** *base)*, ***setfill(int*** *Ch)*, ***setiosflags(fmtflags*** *f)*, ***setprecision(int*** *p)*, ***setw(int*** *W)*, ***showbase***, ***showpoint***, ***showpos***, ***skipws***, ***unitbuf***, ***uppercase***, ***ws***. [11]

Опис деяких маніпуляторів:

dec — десяткова система числення;

oct — вісімкова система числення;

hex — шістнадцяткова система числення;

ws — під час введення ігноруються прогалинні символи;

endl — переведення на новий рядок та очищення буферу;

ends — під час виведення виводиться нульовий символ (кінець рядка ‘\0’);

flush — очищення буферу під час виведення;

setbase(int n) — задає основу системи числення (*n = 8, 16, 10* чи *0*). *0* - основа за замовчуванням (десяткова, крім введення *8*- або*16*-кового числа);

resetiosflags(long) — скидає прапорці, що задані в параметрі;

setiosflags(long) — встановлює прапорці, що задані в параметрі;

setfill(int) — встановлює символ-заповнювач;

setprecision(int) — встановлює максимальну кількість цифр дробової частини для дійсних чисел з фіксованою крапкою (прапорець *fixed*) чи загальну

кількість значущих цифр для чисел з плаваючою крапкою (прапорець *scientific*);

setw(int) — встановлює максимальну ширину поля виведення.

Приклад:

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
  cout << hex << 100 << endl;
  cout << setfill('?') << setw(10) << 2343.0;
  int i; cin >> i;
  return 0;
}
```

Основи об'єктно-орієнтованого програмування

Клас – особливий тип даних, до складу якого входять не тільки данні, а й підпрограми для обробки цих даних (схожий на структуру чи суміш).

Об'єкт – змінна типу даних клас, представник класу.

Клас - це абстракція, загальне поняття, шаблон, уявлення, креслення речі. Об'єкт - це реальна конкретна річ, екземпляр класу.

Приклад:

Класи: собака, кішка, мишка, людина, автомобіль.

Об'єкти: Жучка, Мурка, Мікі-Маус, Петренко, ЗАЗ969М.

Клас є набором характеристик/властивостей (описують його стан) і дій/методів, які він може виконувати. Наприклад, — хай є клас Песик, до властивостей якого можна віднести «вік» і «кличку».

Екземпляр класу може виконувати дії, що описані у класі (Наприклад для класу «Песик»: гавкати, сидіти, лежати, їсти), а також містить значення властивостей класу (клас «Песик» має властивість «вік», а об'єкт «Мухтар» має значення цієї властивості — 2 роки). Крім властивостей і методів, об'єкт має власне ім'я, (не путати з властивістю «кличка»).

Наприклад: *МояСобака.кличка = "Мухтар"*

Що значить, що у об'єкта з іменем «МояСобака», є властивість кличка, і ця кличка — Мухтар.

А може бути і така ситуація:

ХозяїнСобаки.ім'я = "Василь"

ХозяїнСобаки.прізвище = "Іванов"

Тут об'єкт «ХозяїнСобаки», у котрого властивість «ім'я» має значення «Василь», а прізвище — «Іванов».

Подібним чином працюють і методи (дії) об'єктів.

Наприклад:

МояСобака.лежати()

МояСобака.сидіти()

Методи можуть бути з параметрами: *МояСобака.гавкати(3)* (команда гавкнути 3 рази).

Об'єкт завжди повинен бути створений. Клас існує як факт, тобто є завжди. Створення об'єкту виглядає в різних мовах програмування по різному, хоча частіше за все так: *МояСобака = new Песик().*

Так створюється новий екземпляр класу Песик. Іноді він може бути з параметром для додаткової інформації, що необхідна для об'єкту (залежить від реалізації класу): *МояСобака = new Песик("Мухтар").*

Клас повинен бути спочатку оголошений (описаний) перш ніж задавати змінні (об’єкти) цього класу.

Базові принципи ООП

- *пакетування (encapsulation),* це об'єднання в одному об'єкті даних та функцій, які працюють з цими даними. До деяких даних доступ може бути заборонений або обмежений;

- *наслідування (inheritance).* Нові класи можуть будуватися на основі вже існуючих, унаслідувавши їх властивості та методи;

- *поліморфізм (polymorphism)* - дозволяє використовувати одні і ті ж функції для вирішення різних задач;

- *передача повідомлень* - основна методологія побудування ООП. Такі програми є наборами об'єктів і передаванням повідомлень між цими об'єктами;

- *ієрархія класів* – розташування класів в залежності від того, який клас від яких похідний. З самого верху ієрархії основний клас, від якого походять інші класи;

- *перезавантаження функцій* – використовується для поліморфізму. Можна описати декілька однакових функцій але з різною кількістю аргументів, різним їх типом або різним значенням, що повертається;

- *перезавантаження операцій* – використовується для поліморфізму. Дозволяє перевизначити дію стандартних операцій (++, +, -, = і т.д.) над нестандартними даними. Виконується за допомогою функції(методу) у класі з ім'ям *operator<операція>()*. Приклад: *operator++().*

Опис класів та задавання об’єктів

<Оголошення класу>::= ***class*** <ім'я класу>***:***<список класів-батьків>
{
public: //доступне всім
<дані, методи>
protected: //доступне тільки нащадкам
<дані, методи>
private: //доступне тільки у класі
<дані, методи>
} <список змінних(об'єктів)> ***;***
<ім'я класу>::=<ідентифікатор>
Приклад оголошення класу:

```
class MyClass
{
```

```
public:
   MyClass(int=0);       //конструктор
   void SetA(int);       //публічний метод
   int GetA(void);       //публічний метод
   ~MyClass();           //деструктор
private:
   int A;                //приватна властивість
   double B,C;           //приватні властивості
protected:
   int F(int);           //захищений метод
};
```

Можна оголосити змінні цього класу: ***MyClass*** *MC,MC10[10], *Pmc;*

В даному прикладі: MC – статичний об'єкт класу ***MyClass***, *MC10* – масив із 10 статичних об'єктів класу ***MyClass***, *Pmc* - динамічний об'єкт класу ***MyClass*** (вказівник на об'єктів класу ***MyClass***).

Реалізацію методів можна робити або прямо у класі, або окремо. Необхідно реалізовувати всі методи, що описані в класі.

Приклад реалізації двох методів з вищеописаного класу:

```
void MyClass::SetA(int Value)
{
 if (Value>=0)       //перевірка на коректність даних
A=Value;
}
int MyClass::GetA(void)
{
return A;
}
```

Введення та виведення значень простих типів даних у Qt у візуальному режимі

Можна використовувати віджети: *label, lineEdit, textEdit, tableWidget* та ін.

Приклад:

```
float x; int a;
x=ui->lineEdit->text().toFloat(); //Введення дійсної значення до змінної x
a=ui->lineEdit_2->text().toInt();//Введення цілого значення до змінної a
//Виведення:
ui->label->setText(QString("%1").arg(x));
ui->label->setText(QString::number(x));
ui->lineEdit->setText(QString::number(a));
ui->textEdit->setText(QString("x=%1 Ом, %2").arg(x).arg(a));
ui->textEdit->append(QString("Текст %1").arg(x));
```

Приклад введення за допомогою діалогового вікна:

```
#include <QInputDialog>

float x;
bool ok;
```

*x=**QInputDialog**::**getDouble**(this,"Введення значень", "x=", 5, 0, 1000, 1, &ok);*

Результат подібного виклику вікна представлений на рисунку 1.1.

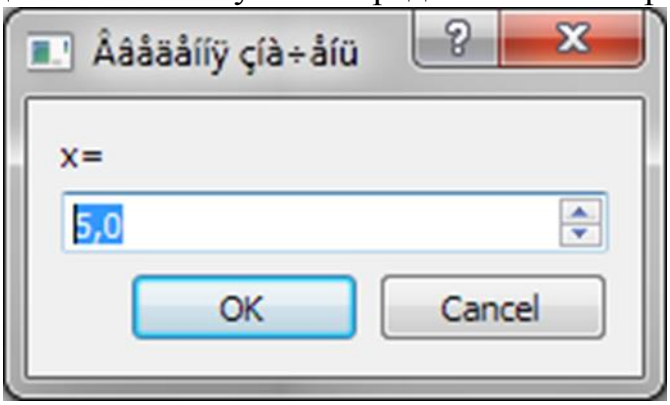


Рисунок 1.1 – Приклад діалогового вікна

Аналогічно можна викликати діалогові вікна ***getInt***, ***getText*** і.т.ін.

Для можливості вводити і виводити повідомлення з використанням кирилічних символів в усіх можливих ситуаціях потрібно підключити модуль *QTextCodec* і встановити кодування *CP1251*. Функції для встановлення кодування можна записати в конструктор головної форми програми.
Приклад:
***#include** <**QTextCodec**>*

***QTextCodec**::**setCodecForCStrings**(QTextCodec::codecForName("CP1251"));*
***QTextCodec**::**setCodecForTr**(QTextCodec::codecForName("CP1251"));*
***QTextCodec**::**setCodecForLocale**(QTextCodec::codecForName("CP1251"));*

У нових версіях Qt (починаючи з 5.2) локалізація здійснюється по іншому.

Приклад вирішення лабораторного завдання

Задача: *Написати програму для переведення номіналу опору резистора R у скорочену форму (Ом, кОм, МОм за правилом:*

$$R = \begin{cases} Ом, & \text{якщо } 0 \le R < 1000; \\ кОм, & \text{якщо } 1\,000 \le R < 1\,000\,000; \\ МОм, & \text{якщо } 1\,000\,000 \le R < 1\,000\,000\,000. \end{cases}$$

Блок схема програми зображена на рисунку 1.2.
Створюємо новий візуальний проект у Qt Creator з базовим класом QDialog. Конструюємо головне вікно нашої програми, наприклад як показано на рисунку 1.3. На формі розміщені віджети: *Label, Line Edit, Text Edit* та *Push Button*.
Вміст файлів проекту:

```
#ifndef DIALOG_H        //dialog.h
#define DIALOG_H
```

```
#include <QDialog>
namespace Ui {
    class Dialog;
}
class Dialog : public QDialog
{
    Q_OBJECT
public:
    explicit Dialog(QWidget *parent = 0);
    ~Dialog();
private slots:
    void on_pushButton_clicked();

private:
    Ui::Dialog *ui;
};
#endif // DIALOG_H
```

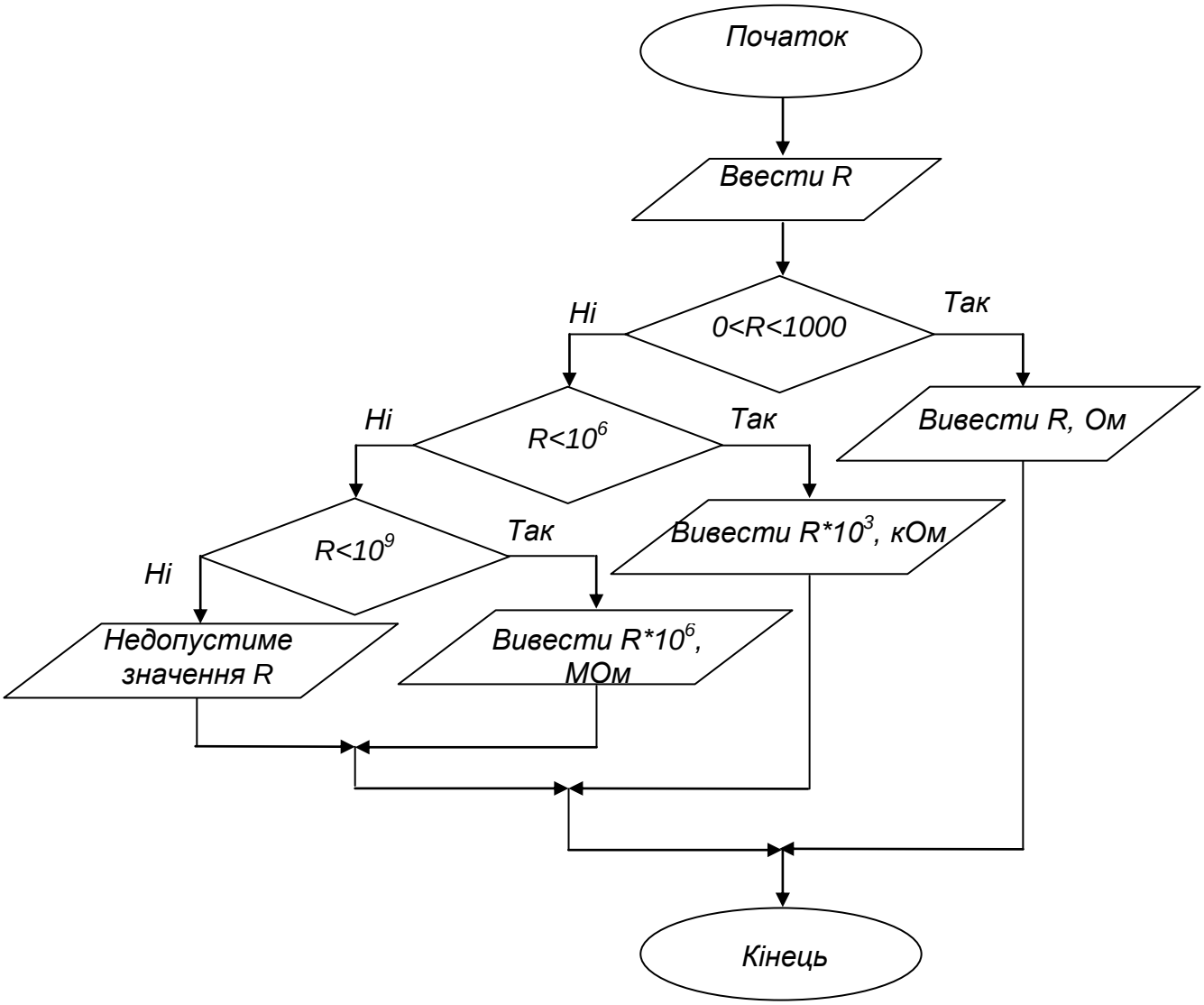


Рисунок 1.2 – Блок схема програми

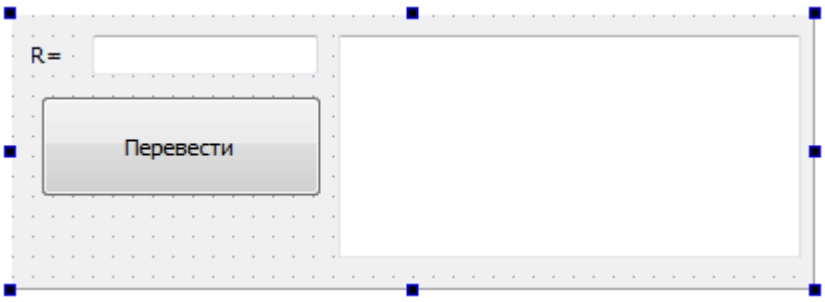


Рисунок 1.3 – Форма головного вікна програми

```
#include "dialog.h"          //dialog.cpp
#include "ui_dialog.h"
#include <QMessageBox>
#include <QTextCodec>
//------Задавання класу Rezistor--------
class Rezistor
{ public:
     Rezistor(double=0);          //конструктор
     void SetR(double);
     double GetR(void);
     QString Perevod(void);
     ~Rezistor();            //деструктор
private:
     double Opir;
protected:
bool Control(double);
};
//-----Задавання методів класу Rezistor----
Rezistor::Rezistor(double R)
{
Opir=R;
}
bool Rezistor::Control(double R)
{
if (R>=0&&R<=1e12) return true; else return false;
}
void Rezistor::SetR(double r)
{
if (Control(r)) Opir=r; else {
       QMessageBox msgBox;
       msgBox.setText("R<0 або R>1000000000000");
msgBox.exec(); }
}
double Rezistor::GetR(void)
{  return Opir;  }
QString Rezistor::Perevod(void)
{
if(Opir<1000)
```

```
  return QString("%L1 = %L2 Ом").arg(Opir).arg(Opir,
0,'g',3);
else  if(Opir<1e6)
     return QString("%L1 = %L2
кОм").arg(Opir).arg(Opir/1e3,0,'g',3);
   else if(Opir<1e9)
      return QString("%L1 = %L2
МОм").arg(Opir).arg(Opir/1e6,0,'g',3);  return "";
}
Rezistor::~Rezistor()
{   }
//-----кінець задавання методів класу Rezistor----
Dialog::Dialog(QWidget *parent) :
QDialog(parent),
ui(new Ui::Dialog)
{
ui->setupUi(this);
QTextCodec::setCodecForCStrings(QTextCodec::codecForName("CP1
251"));
QTextCodec::setCodecForTr(QTextCodec::codecForName("CP1251"))
;
QTextCodec::setCodecForLocale(QTextCodec::codecForName("CP125
1"));
}
Dialog::~Dialog()
{
delete ui;
}
void Dialog::on_pushButton_clicked()
{
Rezistor *R1;
R1 = new Rezistor();
R1->SetR(ui->lineEdit->text().toDouble());
ui->textEdit->append(R1->Perevod());
}
```

Результат роботи програми з тестами продемонстрований на рисунку 1.4

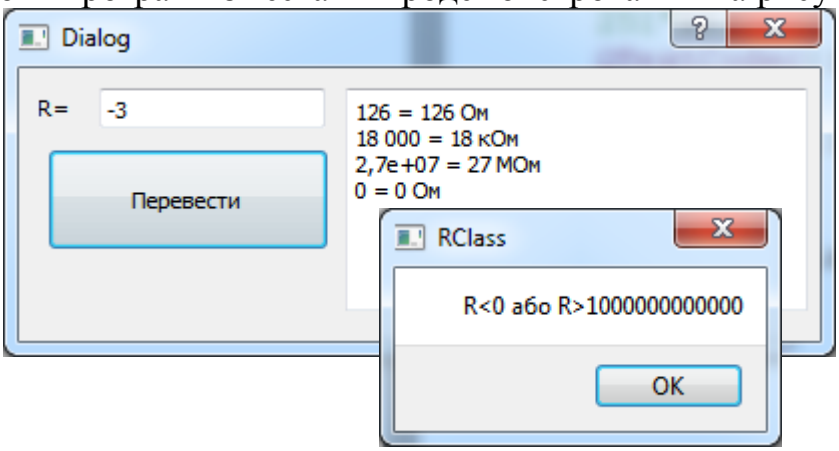


Рисунок 1.4 – Результат роботи програми

1.2 Методичні вказівки:

а) вивчити відмінності між мовами С та С++;

б) вивчити основні принципи ООП та вводу – виводу у візуальному режимі; розробити простий клас та використати його для вирішення свого варіанту задачі;

в) розробити візуальний інтерфейс (графічну форму) програми;

г) розробити алгоритми розв`язання задач свого варіанту, записавши їх у вигляді блок-схем, програма повинна правильно виконуватися за будь-яких припустимих значеннях початкових даних, передбачити усі можливі ситуації в програмі так, щоб уникнути входження в нескінчений цикл (зациклення); правильність роботи всіх гілок програми повинна бути перевірена на тестах;

д) забезпечити в програмі виведення коментарів, відповідних різним можливим ситуаціям.

1.3 Контрольні запитання:

1. Створення візуального проекту у Qt.
2. Введення та виведення у Qt значень змінних простих типів даних у візуальному режимі.
3. Поняття про ООП (класи, об’єкти, методи, властивості і т.д.).
4. Базові принципи ООП.
5. Опис класу у програмі та задавання об’єктів цього класу.
6. Відмінності С++ від С.
7. Старий та новий стиль С++.
8. Оператори введення-виведення <<, >>.
9. Форматування введення-виведення за допомогою членів класу **ios**. Прапорці форматування.
10. Форматування введення-виведення за допомогою маніпуляторів формату.

1.4 Варіанти завдань

Варіант 1

Задача 1: Дано дійсні числа *a,b,c,d*. Якщо $a \le b \le c \le d$, то кожне число замінити найбільшим з них; якщо $a>b>c>d$ то числа залишити без змін; в іншому випадку всі числа замінити їх квадратами.

Вказівки:

1. Значення *a,b,c,d* задати самостійно.
2. Виконати програму для трьох випадків:
   a) $a \le b \le c \le d$; b) $a>b>c>d$; c) $a<b>c<d$.
3. На екран вивести початкові та змінені значення *a,b,c,d*.

Задача 2: Написати програму для знаходження загального опору паралельного та послідовного з'єднання *n* резисторів.

Вказівки: Значення *n* та опори резисторів вводити з клавіатури.

Варіант 2

Задача 1: Дано дійсні числа *x,y*. Якщо *x,y* від`ємні, то кожне значення замінити його модулем; якщо від`ємне лише одне число, то обидва значення збільшити на *0.5*; якщо обидва значення невід`ємні і жодне з них не належить до відрізка [*0.5; 2.0*], то обидва значення зменшити у *10* разів; в інших випадках *x,y* залишити без змін.

Вказівки:

1. Значення *x,y* задавати самостійно.
2. Виконати програму для випадків:
   a) $x<0$ і $y<0$; b) $x<0, y>0$; або $x>0, y<0$; c) $0<x<0.5; y>0$;
   d) $0.5 \le x \le 2$ і $0.5 \le y \le 2$.
3. На екран вивести початкові дані *x,y* та змінені.

Задача 2: Метод послідовних наближень дозволяє вирахувати, для знаходження кореня п'ятого ступеня з додатного числа *a*, наближення:

$$x_{n+1}=\frac{4}{5}x_n+\frac{a}{5x_n^4}.$$

В даному випадку похибка $(n+1)$-го наближення не переважає $\frac{4}{5}a|x_{n+1}-x_n|$.

Написати програму для обчислення кореня п'ятого ступеня з числа *a* з точністю до 10^{-3}, враховуючи, що

$$x_0=\begin{cases}\min(2a;0.95), & \text{якщо } a \le 1\\ a/5, & \text{якщо } 1<a\le 25\\ a/25, & \text{якщо } a>25\end{cases}.$$

Вказівки: значення *a* задавати самостійно, розглянувши випадки: **a)** $a \le 1$; **b)** $1 \le a \le 25$; **c)** $a \ge 25$.

Варіант 3

Задача 1: Дано дійсні додатні числа *a,b,c,d*. З`ясувати, чи можливо прямокутник зі сторонами *a,b* вмістити усередині прямокутника зі сторонами *c,d*, щоб кожна зі сторін одного прямокутника була паралельна чи перпендикулярна кожній стороні другого прямокутника.

Вказівки:

1. Виконати програму, коли:
   a) $a<c, b>d$; b) $a>c, b<d$; c) $d>a>c, b<c$; d) $d>b>c, a<c$.
2. Значення *a,b,c,d* задавати самостійно.

Задача 2: Написати програму, яка обчислює границю послідовності $\{a_n\}$,

де $a_n = \dfrac{n}{\sqrt{n^2+1}+\sqrt{n^2-1}}$,

прийнявши за нього таке значення a_n, з яким $|a_{n+1}-a_n|<10^{-3}$

Вказівки: на екран вивести члени послідовності від a_1 до a_n та номер члена *n*, для якого справедлива показана вище рівність.

Варіант 4

Задача 1: Написати програму для переведення номіналу ємності конденсатора *C* у скорочену форму (пФ, нФ, мкФ, Ф) за правилом:

$$C=\begin{cases} пФ, & \text{якщо } 0<C<10^{-9};\\ нФ, & \text{якщо } 10^{-9}\le C<10^{-6};\\ мкФ, & \text{якщо } 10^{-6}\le C<1;\\ Ф, & \text{якщо } C\ge 1. \end{cases}$$

Вказівки: На екран вивести округлене значення ємності із відповідним позначенням, наприклад, якщо ввели 0.0047, вивести 4700 мкФ.

Задача 2: Корінь рівняння знаходиться послідовними наближеннями за формулою:

$$x_{n+1}=\frac{2-x_n^3}{5}$$

Написати програму, яка знаходить таке значення кореня, з яким різниця між двома сусідніми наближеннями не перебільшує 10^{-3}, виходячи із початкового наближення $x_0=1$.

Варіант 5

Задача 1: Якщо сума трьох різних дійсних чисел *x,y,z* менше за одиницю, то найменше з них замінити напівсумою двох інших; в іншому випадку замінити менше з *x,y* напівсумою двох, що залишилися.

Вказівки:

1. Виконати програму при: a) $x+y+z=1$; b) $x+y+z<1$.
2. Значення *x,y,z* задавати самостійно.

Задача 2. Дано додатні дійсні числа a,x,ε. В послідовності y_1,y_2, яка утворена за законом:

$$y_0=a,\ y_i=\frac{1}{2}\left(y_{i-1}+\frac{x}{y_{i-1}}\right),\ i=1,2,....$$

Знайти перший член y_n, для якого виконується нерівність $\left|y_n^2-y_{n-1}^2\right|<\varepsilon$.

Вказівки: значення a,x задавати самостійно. Прийняти $\varepsilon=10^{-3}$.

Варіант 6

Задача 1: Написати програму для переведення номіналу опору резистора R у скорочену форму (Ом, кОм, МОм, ГОм) за правилом:

$$R=\begin{cases} Ом, & \text{якщо } 0\le R<1000;\\ кОм, & \text{якщо } 1\,000\le R<1\,000\,000;\\ МОм, & \text{якщо } 1\,000\,000\le R<1\,000\,000\,000;\\ ГОм, & \text{якщо } R\ge 1\,000\,000\,000. \end{cases}$$

Вказівки: На екран вивести округлене значення опору із відповідним позначенням, наприклад, якщо ввели 5600, вивести 5.6 кОм.

Задача 2: Обчислити безкінцеву суму із заданою точністю $\varepsilon(\varepsilon>0)$, враховуючи, що потрібна точність досягнута, якщо наступний доданок виявився за модулем менший, ніж ε.

$$\sum_{i=1}^{\infty}\frac{(-1)^{i+1}}{i(i+1)(i+2)};$$

Вказівки: вважати $\varepsilon=10^{-3}$.

Варіант 7

Задача 1: Написати програму для визначення номеру телевізійного каналу за введеною несучою частотою зображення в першому діапазоні метрових хвиль. Розподіл каналів за частотою приведений в таблиці 1.4.

Таблиця 1.4 – Розподіл каналів за частотою

Діапазон хвиль	48–57	58–66	76–84	85–92	93–100	МГц
Номер каналу	1	2	3	4	5	N

Задача 2: Написати програму знаходження коефіцієнтів ряду згідно з формулою:

$$a_k=\frac{k^2+10}{k^3+1-k^2}\ ,$$ де $к=1,2,3$ – номер коефіцієнту ряду

Вказівки:

1. Знаходити коефіцієнти, доки $\left|a_{k+1}-a_k\right|<\varepsilon$.
2. Вивести на екран коефіцієнти з номерами, що діляться на 3 та ці номери.
3. Підібрати таке ε, з яким кількість членів (коефіцієнтів) ряду буде лежати в межах $30\le k\le 40$.

Варіант 8

Задача 1: Задано три дійсних числа, що є довжинами відрізків. Визначити, чи можливо побудувати трикутник з такими довжинами сторін і, якщо це можливо – визначити тип трикутника: рівносторонній (рівнобедрений), прямокутний чи інший трикутник.

Задача 2: Написати програму для знаходження загальної ємності паралельного та послідовного з'єднання *n* конденсаторів.

Вказівки: Значення *n* та ємності конденсаторів вводити з клавіатури.

Варіант 9

Задача 1: Написати програму для виводу на екран середньої температури пори року виходячи з таблиці 1.5.

Таблиця 1.5 – Температура пори року

Пора року	Весна	Літо	Осінь	Зима
tcep..	+10o	+25o	+5o	-15o

Вказівки: Пору року задавати за допомогою літер: *"B"* – весна, *"L"* – літо, *"O"* – осінь, *"Z"* – зима.

Задача 2: Вирахувати значення інтегралу $J_n=\frac{1}{e}\int_0^1 x^n e^x dx$ за наступною рекурсивною формулою: $J_n=1- nJ_{n-1}$, $n=1,2,\ldots$

Вказівки: Початкове наближення інтеграла прийняти рівним $J_0=1-1/e$, вирахування закінчити, коли поточне значення J_i за абсолютним значенням стане менше 10^{-4}

Варіант 10

Задача 1: Ввести *n* – ціле. Якщо $n\le 0$, то видати про це повідомлення і ввести нове значення *n*. Якщо $1\le n\le 5$, то обчислити $y=a^n+ln(b^2+c^4+d)$; попередньо здійснивши введення даних *a,b,c,d*. Вивести на екран *a,b,c,d,y*. Якщо $n>5$, обчислити $y=a^n+b+c+d$, попередньо здійснивши введення даних *a,b,c,d*. Вивести на екран *a,b,c,d,y*.

Вказівки: виконати програму при: a) $n\le 0$; b) $1\le n\le 5$; c) $n>5$.

Задача 2: Знайти границю послідовності C_i, коли *i* прямує від 0 до нескінченності, де $C_i=\frac{i^2}{\sqrt{i^4+1}}$, приймаючи за нього таке значення C_i, з яким $|C_i - C_{i-1}|<10^{-3}$.

Вказівки: На екран вивести всі значення C_i та номер останнього значення.

2 ЛАБОРАТОРНА РОБОТА №2. РОБОТА ІЗ МАСИВАМИ ЧИСЕЛ У ВІЗУАЛЬНОМУ РЕЖИМІ

Мета роботи: отримати практичні навички по роботі з масивами мовою С++ з використанням ООП; оволодіти навичками алгоритмізації та програмування структур із вкладеними циклами, способами введення та виведення одновимірних та багатовимірних масивів у візуальному режимі

2.1 Теоретичні відомості

У візуальному режимі елементи масив можна зберігати у чарунках віджету *tableWidget* і там же виконувати всі дії над ними. Також можна задати окремо змінну типу масив і з нею працювати, а *tableWidget* використовувати тільки для введення та виведення. Ще масив можна задати всередині класу, як властивість, і визначити методи для роботи з цим масивом.

Приклад введення та виведення одної комірки масиву з використанням *tableWidget:*

```
double a[6][8]; // Описання змінної-двомірного масиву
……...
ui->tableWidget->setRowCount(6); // Встановлення кількості рядків
ui->tableWidget->setColumnCount(8); // Встановлення к-ті стовпчиків
// Зчитування з комірки 2, 3.
a[2][3]=ui->tableWidget->item(2,3)->text().toDouble();
// Запис у комірку 0, 0 числа 123:
QTableWidgetItem *ptw=0;
ptw=new QTableWidgetItem(QString::number(123));
ui->tableWidget->setItem(0,0,ptw);
```

У прикладі вище, перш ніж записати у комірку, створюється об’єкт комірки класу *QTableWidgetItem.*

За необхідністю, у візуальному режимі не обов’язково використовувати лише *Table Widget* для роботи з масивами, можна скористатися і іншими віджетами, наприклад: *Label, Line Edit, Text Edit,* але організація введення та виведення масиву в такому варіанті буде значно складнішою, ніж у *tableWidget*.

Для роботи з одновимірними масивами (векторами-рядками та векторами-стовпчиками) також можна використовувати *Table Widget*. Але в цьому випадку один із вимірів залишається фіксованим і дорівнює одиниці. Тобто реально задаємо двомірний масив розмірністю 1xN або Nx1.

Не існує простих методів візуалізації для роботи з масивами, у яких кількість вимірів 3 і більше. Хоча можна використати все ті ж віджети *tableWidget.,* розташувавши на формі їх у кількості, що дорівнює третьому виміру (для трьохвимірних масивів). Можна також перемикати показ віджетів за номером третього виміру, виводячи поточну площину у одному *tableWidget,* або створити декілька вкладок і перемикатися між ними, тощо.

Приклад вирішення задачі з використанням масиву, що розташований у глобальній змінній, введення та виведення відбувається за допомогою *tableWidget.*

Задача: *Написати програму транспонування матриці* $A=(a_{ij})$, *де* $i=1,2,...n$, $j=1,2,...m$ *для довільних значень* n,m. *Операція транспонування полягає в заміні рядків матриці стовпчиками (*i*-й рядок заміняється на* j*-й стовпчик).*

Блок схема програми зображена на рисунку 2.1.

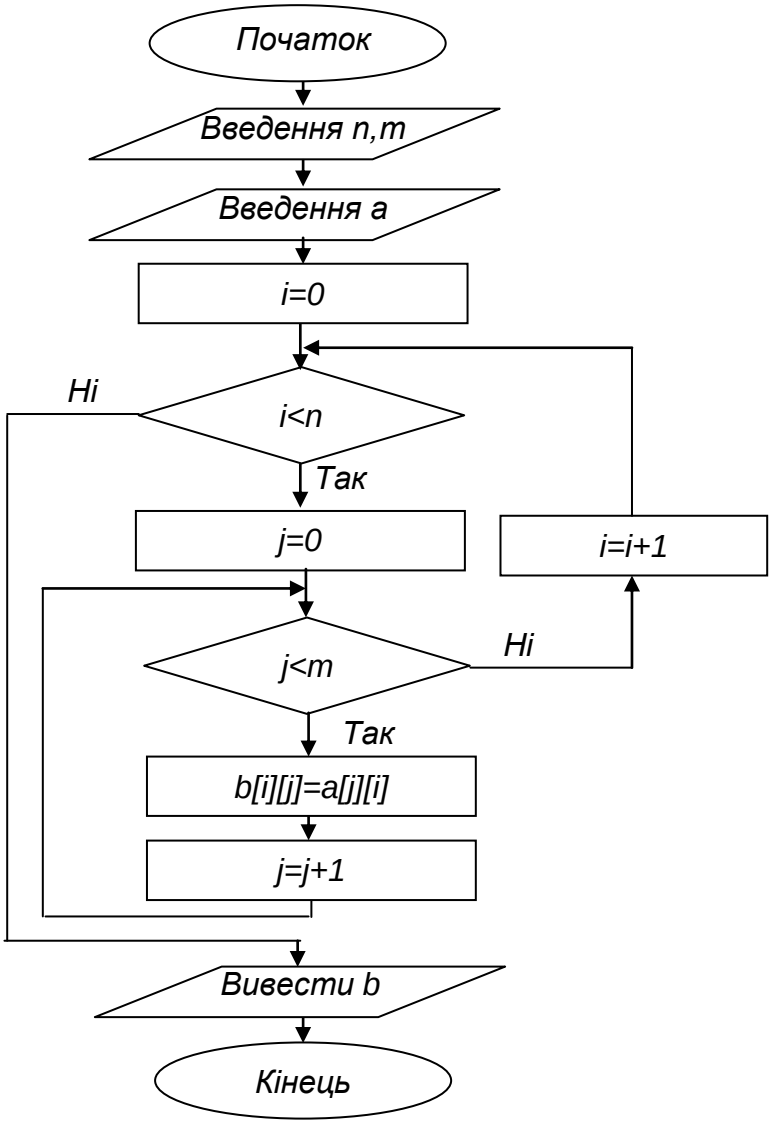


Рисунок 2.1 – Блок схема програми

Створюємо новий візуальний проект у Qt Creator з базовим класом QDialog. Конструюємо головне вікно нашої програми, наприклад як показано на рисунку 2.2. На формі розміщені по 2 віджети: *Label, Line Edit, Table Widget* та *Push Button*.

Вміст файлу dialog.h:

```
#ifndef DIALOG_H            //dialog.h
#define DIALOG_H
```

```
#include <QDialog>
namespace Ui {
    class Dialog;
}
class Dialog : public QDialog
{
    Q_OBJECT
public:
    explicit Dialog(QWidget *parent = 0);
    ~Dialog();
private slots:
    void on_pushButton_clicked();
    void on_pushButton_2_clicked();
private:
    Ui::Dialog *ui;
};
#endif // DIALOG_H
```

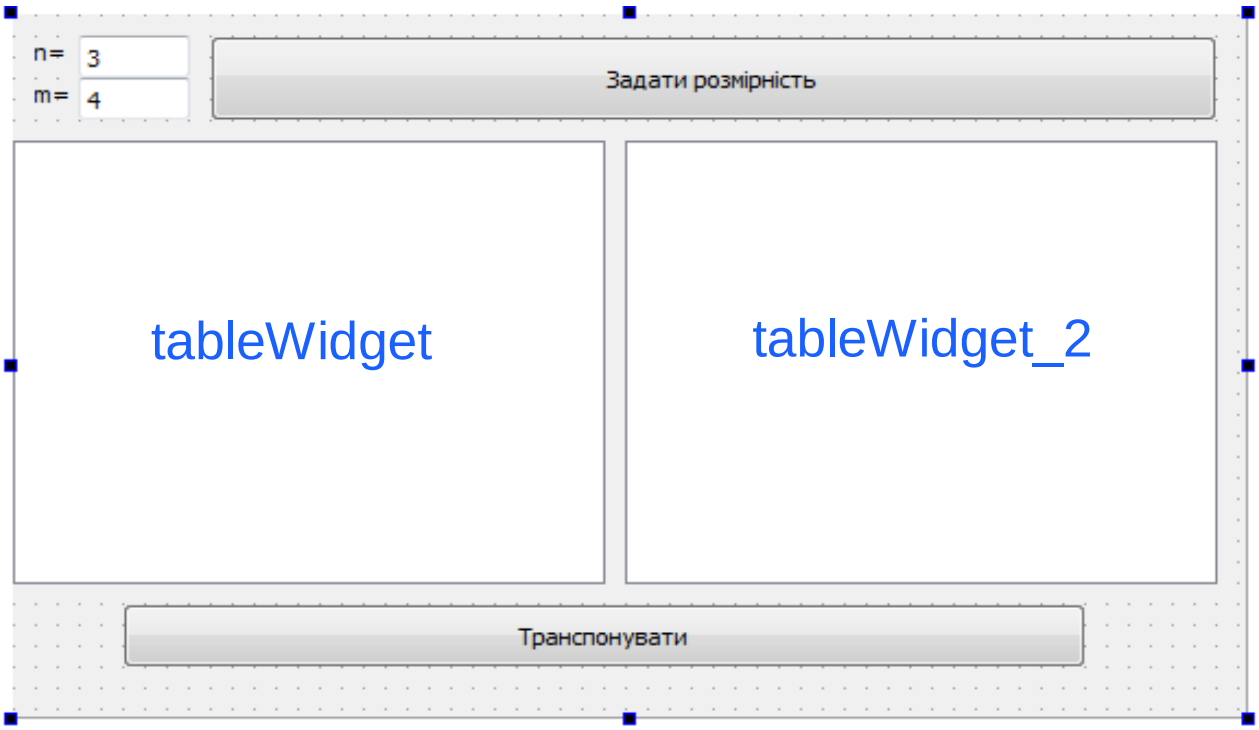


Рисунок 2.2 – Форма головного вікна програми

Вміст файлу dialog.cpp:

```
#include "dialog.h"                //dialog.cpp
#include "ui_dialog.h"
Dialog::Dialog(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::Dialog)
{
    ui->setupUi(this);
}
Dialog::~Dialog()
```

```
{
    delete ui;
}
double a[10][10],b[10][10];
int n,m;
void Dialog::on_pushButton_clicked()
{   n=ui->lineEdit->text().toInt();
    m=ui->lineEdit_2->text().toInt();
    ui->tableWidget->setRowCount(n);
    ui->tableWidget->setColumnCount(m);
    ui->tableWidget_2->setRowCount(m);
    ui->tableWidget_2->setColumnCount(n);
    for(int i=0;i<n;i++)
      for(int j=0;j<m;j++) {
         QTableWidgetItem *ptw=0;
         ptw=new QTableWidgetItem("");
         ui->tableWidget->setItem(i,j,ptw);
                        }
}
void Dialog::on_pushButton_2_clicked()
{
    for(int i=0;i<n;i++)
        for(int j=0;j<m;j++)
          a[i][j]=ui->tableWidget->item(i,j)-
>text().toDouble();
       for(int i=0;i<m;i++)
        for(int j=0;j<n;j++)
          {
            b[i][j]=a[j][i];
            QTableWidgetItem *ptw=0;
            ptw=new
QTableWidgetItem(QString::number(b[i][j]));
            ui->tableWidget_2->setItem(i,j,ptw);
           }
}
```

Вміст файлу dialog.cpp для випадку вирішення вищенаведеної задачі шляхом створення свого класу Masiv:

```
#include "dialog.h"                  //dialog.cpp
#include "ui_dialog.h"
Dialog::Dialog(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::Dialog)
{
    ui->setupUi(this);
}
Dialog::~Dialog()
{
    delete ui;
```

```
}

class Masiv //Задавання нашого класу

{
 public:      Masiv(int N, int M);
                 void InputMas(int i, int j, double ms);
                 double OutputMas(int i, int j);
                 void SetN(int);
                 int GetN();
                 void SetM(int);
                 int GetM();
                 void Transponired(Masiv);
                 ~Masiv();
 private:    double mas[100][100]; //безпосередньо сам масив
                 int n,m;          //поточний розмір масиву
};
Masiv::Masiv(int N, int M)          //Реалізація конструктора
{
   if(N>0&&N<=100) n=N; else n=1;
   if(M>0&&M<=100) m=M; else m=1;
}
            //Реалізація методу задавання кількості рядків
void Masiv::SetN(int N)
{  if(N>0&&N<=100) n=N;
}
            //Реалізація методу задавання кількості стовпчиків
void Masiv::SetM(int M)
{  if(M>0&&M<=100) m=M;
}
            //Реалізація методу отримання кількості рядків
int Masiv::GetN() { return n; }
            //Реалізація методу отримання кількості стовпчиків
int Masiv::GetM() { return m; }
            //Реалізація метода отримання елемента масиву
double Masiv::OutputMas(int i,int j)
{
  if(i<GetN()&&j<GetM())return mas[i][j];else return 0;
}
            //Реалізація метода запису елемента масиву
void Masiv::InputMas(int i,int j,double ms)
{
  if(i<GetN()&&j<GetM()) mas[i][j]=ms;
}
            //Реалізація метода транспонування масиву
void Masiv::Transponired(Masiv x)
{
   for(int i=0;i<GetN();i++)
```

```
      for(int j=0;j<GetM();j++)
                  mas[i][j]=x.mas[j][i];
 }
Masiv::~Masiv()          //Реалізація деструктора
{
}

Masiv a(3,4), b(4,3); //Задавання двох об'єктів нашого класу

void Dialog::on_pushButton_clicked()
{ a.SetN(ui->lineEdit->text().toInt());
  a.SetM(ui->lineEdit_2->text().toInt());
  b.SetM(ui->lineEdit->text().toInt());
  b.SetN(ui->lineEdit_2->text().toInt());
  ui->tableWidget->setRowCount(a.GetN());
  ui->tableWidget->setColumnCount(a.GetM());
  ui->tableWidget_2->setRowCount(b.GetN());
  ui->tableWidget_2->setColumnCount(b.GetM());
  for(int i=0;i<a.GetN();i++)
     for(int j=0;j<a.GetM();j++) {
        QTableWidgetItem *ptw=0;
        ptw=new QTableWidgetItem("");
        ui->tableWidget->setItem(i,j,ptw); }
}
void Dialog::on_pushButton_2_clicked()
{
    for(int i=0;i<a.GetN();i++)     //Введення "a"
      for(int j=0;j<a.GetM();j++)
         a.InputMas(i,j,ui->tableWidget->item(i,j)-
>text().toDouble());
    b.Transponired(a);              //Транспонування
    for(int i=0;i<b.GetN();i++)     //Виведення "b"
       for(int j=0;j<b.GetM();j++)
         {
           QTableWidgetItem *ptw=0;
           ptw=new
QTableWidgetItem(QString("%1").arg(b.OutputMas(i,j)));
           ui->tableWidget_2->setItem(i,j,ptw);
         }
}
```

2.2 Методичні вказівки

а) вивчити правила організації масивів та синтаксис описання в програмі одномірних та двомірних масивів(матриць);

б) вивчити особливості введення та виведення масивів у візуальному режимі;

в)розробити візуальний інтерфейс для забезпеченні вирішення задачі свого варіанту;

г)на основі знань з ООП розробити клас та використати його для роботи з масивами;

д) розробити алгоритм вирішення задач свого варіанту, записавши їх у вигляді блок-схем;

е) забезпечити в програмі виведення коментарів та природне завершення програми в випадках можливого зациклення та зависання програми.

2.3 Контрольні запитання

1. Робота з масивами у візуальному режимі.
2. Введення масиву за допомогою *tableWidget*.
3. Виведення масиву за допомогою *tableWidget*.
4. Масиви в ООП. Приклад класу для роботи з масивом.
5. Особливості введення та виведення одновимірних масивів та масивів, з кількістю вимірів більше двох.
6. Властивості віджету *tableWidget*, їх встановлення у тексті програми та у конструкторі форм.

2.4 Варіанти завдань

Варіант 1

Задача1: Помножити матрицю A_{MxN}, на вектор R, з розмірністю n за формулою:

$$U_i = \sum_{j=1}^{n} a_{ij} * r_j \quad \text{, де } i=1,2,...m,\ j=1,2,...n.$$

Вказівки: на екран вивести попередню матрицю A, вектор R (задати самостійно) та результуючий вектор U.

Задача2: Дано натуральне число n, цілочислова квадратна матриця A порядку n. Отримати $b_1...b_n$, де b_i – це найменше із значень елементів, які знаходяться на початку i-го рядка матриці до елемента, який належить до головної діагоналі включно.

Вказівки: матрицю задавати самостійно.

Варіант 2

Задача1: Знайти скласти та відняти дві матриці A і B з однаковою розмірністю mxn за формулами:

$c_{ij}=a_{ij}+b_{ij}$; $d_{ij}=a_{ij}-b_{ij}$; де $i=1,2,....m,\ j=1,2,....n$.

Вказівки: на екран вивести попередні матриці (задавати самостійно) та результуючі.

Задача2: Дана цілочислова квадратна матриця порядку *4*. Знайти найменше із значень елементів стовпця, який володіє найбільшою сумою по модулю елементів. Якщо таких стовпчиків декілька, то взяти перший з них.

Вказівки: попередню матрицю задати самостійно.

Варіант 3

Задача1: Написати програму транспонування матриці $A=(a_{ij})$, де $i=1,2,...n,\ j=1,2,...n$ для довільного значення n. Операція транспонування полягає в заміні рядків матриці стовпчиками (i-й рядок заміняється на j-й стовпчик).

Вказівки: на екран вивести попередню матрицю та результуючу (попередню матрицю задавати самостійно).

Задача2: Дано натуральне n, цілочислова квадратна матриця порядку n. Отримати $b_1....b_n$, де b_i – це сума елементів, які знаходяться після першого від`ємного елемента в $i$-му рядку (якщо всі елементи рядка невід`ємні, то прийняти $b_i=100$).

Вказівки: попередню матрицю задавати самостійно.

Варіант 4

Задача1: Помножити матрицю A з розмірністю mxn на матрицю B з розмірністю nxl за формулою

$$c_{kj}= \sum_{i=1}^{n} (a_{ki} * b_{ij}) \text{ , де } j=1,2,...l,\ k=1,2,...m.$$

Отримана матриця має розмірність mxl.

Вказівки: на екран вивести матриці A,B (задавати самостійно) та C.

Задача2: Дана дійсна матриця із розмірністю mxn. Отримати послідовність $b_1...b_n$, де b_i – це найбільше із значень елементів i-го рядка.

Вказівки: попередню матрицю задавати самостійно.

Варіант 5

Задача1: У заданій дійсній матриці із розмірністю *3*x*4* поміняти місцями рядок, який містить елемент із найбільшим значенням, із рядком, який містить елемент із найменшим значенням. Припускається, що такі елементи єдині.

Вказівки: попередню матрицю задавати самостійно.

Задача2: Дана дійсна матриця із розмірністю mxn. Отримати послідовність $b_1...b_n$, де b_i – це добуток квадратів тих елементів i-го рядка, модулі яких належать до відрізку [*1*, *1.5*].

Вказівки: попередню матрицю задавати самостійно.

Варіант 6

Задача1: В даній дійсній квадратній матриці із розмірністю n знайти суму елементів рядка, в якій знаходиться елемент із найменшим значенням. Припускається, що такий елемент єдиний.

Вказівки: попередню матрицю задавати самостійно.

Задача2: Дана дійсна матриця із розмірністю mxn. Отримати послідовність $b_1....b_n$, де b_i – це число від`ємних елементів в i-му рядку.

Вказівки: попередню матрицю задавати самостійно.

Варіант 7

Задача 1: Написати програму для арифметичних дій над матрицею $A=[a_{ij}]$ в залежності від нажатої клавіші:

$$[A]=\begin{cases}[A]+x, & \text{якщо '+';}\\ [A]-x, & \text{якщо '–';}\\ [A]*x, & \text{якщо '*'.}\end{cases} \quad i=1,2,3...n;\ j=1,2,3...m.$$

Вказівки: m, n та початкову матрицю задати самостійно, x– вводити з клавіатури після натискання ‘+’, ’-’ або ‘*’.

Задача 2: Задане натуральне число k, цілочислова матриця порядку k. Отримати вектор-стовбець $c_1,c_2...c_i...c_k$, де c_i –сума від’ємних елементів i-го рядка.

Вказівки: значення k та початкову матрицю задати самостійно.

Варіант 8

Задача 1: Заданий двомірний масив $A=(a_{ij})$, де $i=1,2…k$, $j=1,2…f$, елементами якого є цілі числа, які складаються з будь якої кількості цифр. Написати програму для складання матриці, елементами якої будуть числа , які дорівнюють кількості цифр в однійменній комірці в масиві A.

Вказівки: початкову матрицю задати самостійно, на екран вивести початкову та результуючу матриці.

Задача 2: впорядкувати послідовність $c_1…c_n$, яка складається з дійсних чисел в порядку зменшення. Дробові числа округлити до найближчого цілого числа.

Вказівки: початкову послідовність задати самостійно, на екран вивести початкову та результуючу послідовність.

Варіант 9

Задача 1: Заданий двомірний масив $A=(a_{ij})$, де $i=1,2…n$, $j=1,2…m$,. Сформувати одномірний масив B, що складається з від'ємних елементів масиву A, та знайти їх суму.

Задача 2: Задане натуральне число k, цілочисловий одномірний масив порядку k. Поміняти у масиві максимальний елемент з першим, а мінімальний з останнім.

Вказівки: Значення k та початковий масив задати самостійно.

Варіант 10

Задача 1: Заданий двомірний масив $A=(a_{ij})$, де $i=1,2…n$, $j=1,2…m$, елементами якого є цілі числа. Впорядкувати інформацію в масиві в порядку зростання.

Вказівки:

1. Початковий масив задати самостійно.
2. На екран вивести початковий та результуючий масиви.

Задача 2: В одномірному масиві $u_1…u_n$, що складається з величин напруги джерел живлення, знайти кількість стандартних величин напруги: 1.5В, 3В, 4.5В, 6В, 9В, 12В.

Вказівки: початковий масив задати самостійно.

3 ЛАБОРАТОРНА РОБОТА №3. ПРОГРАМУВАННЯ З ВИКОРИСТАННЯМ ПІДПРОГРАМ КОРИСТУВАЧА

Мета роботи: отримання навичок з алгоритмізації і програмування задач з використанням підпрограм та звертання до них у ООП; навичок з вибору параметрів підпрограм; навичок по створенню та роботою методів класів ООП.

3.1 Теоретичні відомості

У С++ є відмінності в роботах з підпрограмами, у порівнянні з мовою С:

- ключове слово void у круглих дужках під час опису підпрограми, якщо не треба формальних параметрів, не обов'язкове (еквівалентно int m(); і int m(void));

- всі функції повинні мати прототипи (попередній опис), якщо викликаються раніше, ніж описані.

- якщо функція повертає тип, відмінний від void, то оператор return обов’язково повинен бути і мати значення відповідного типу;

- можна оголошувати (описувати) локальні змінні не тільки на початку блоку, а в будь якому місці.

Підпрограми у ООП використовуються в якості методів у класах. Можна використовувати як методи, що повертають результат у місце виклику(функції), так і методи, що не повертають результат у місце виклику (процедури). Також можуть бути методи з параметрами та без параметрів. Все аналогічно як у звичайних підпрограм. Викликаються методи шляхом звертання до об’єкта відповідного класу (ім’я об’єкта) з вказуванням через крапку(для статичного об’єкта) або через тире і знак більше (для динамічного об’єкта) імені метода, що викликається і пустих круглих дужок, якщо метод без параметрів, або в дужках вказуються фактичні параметри, якщо метод з параметрами.

Для статичного об’єкт, як і для звичайної змінної, місце у пам’яті відділяється в межах сегменту коду програми під час компіляції(в середині файлу, що виконується). Динамічний же об’єкт описується як вказівник (аналогічно як і динамічна змінна) і місце у пам’яті для нього відділяється під час запуску програми на виконання в момент створення динамічного об’єкта відповідною командою (явний чи опосередкований виклик метода-конструктора об’єкта) у вільній оперативній пам’яті комп’ютера.

Приклад задавання статичних об’єктів класу *Rezistor*, що описаний в першій лабораторній роботі (стор. 16) та виклик відповідних методів*:*

```
Rezistor R1, R2;                    // Опис статичних об’єктів
R1.SetR(12);                        // Виклик методу статичного об’єкту
ui->textEdit->append(R1.Perevod());  // Виклик методу статичного об’єкту в якості параметра іншого методу динамічного об’єкту ui.
```

Приклад задавання динамічних об’єктів класу *Rezistor:*

```
Rezistor *R1, *R2;                  // Опис динамічних об’єктів
R1 = new Rezistor();                // Створюємо об’єкт R1, конструктор
                                    //без параметру
```

*R2 = **new Rezistor**(1000); // Створюємо об'єкт R2, конструктор з*
// параметром
*R1->**SetR**(12); // Виклик методу динамічного об'єкту*
*ui->**textEdit**->**append**(R1->**Perevod**());// Виклик методу динамічного об'єкту в якості параметра іншого методу динамічного об'єкту **ui.***

Більш докладніше про підпрограми та їх параметри можна почитати у методичних вказівках попереднього семестру у третій лабораторній роботі[15].

Приклад вирішення лабораторного завдання

Задача. Задані дійсні числа a_1,a_2,a_3; b_1,b_2,b_3; c_1,c_2,c_3. Отримати:

$$e=\begin{cases}\max(b_1,b_2,b_3)+\max(c_1,c_2,c_3), & \text{якщо } |\max(a_1,a_2,a_3)|>10,\\ 1+[\max(\sum_{i=1}^{3}a_i,\sum_{i=1}^{3}b_i,\sum_{i=1}^{3}c_i)]^2, & - \text{ в іншому випадку.}\end{cases}$$

Вказівки: попередні дані задавати самостійно; знаходження max оформити у вигляді функції, а обчислення $\sum$ та введення a_i ,$b_{i,}$, c_i– у вигляді аналога процедур.

Блок схема програми зображена на рисунках 3.1, 3.2.

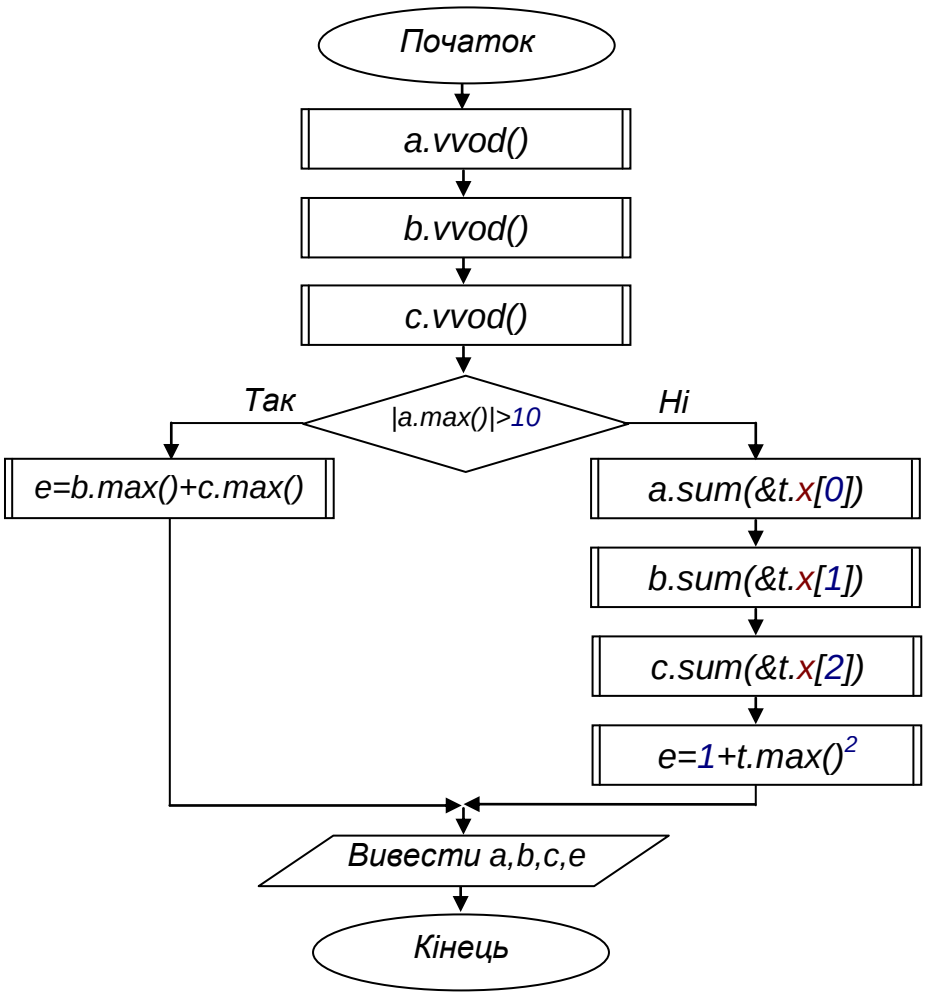


Рисунок 3.1 – Блок схема основної програми

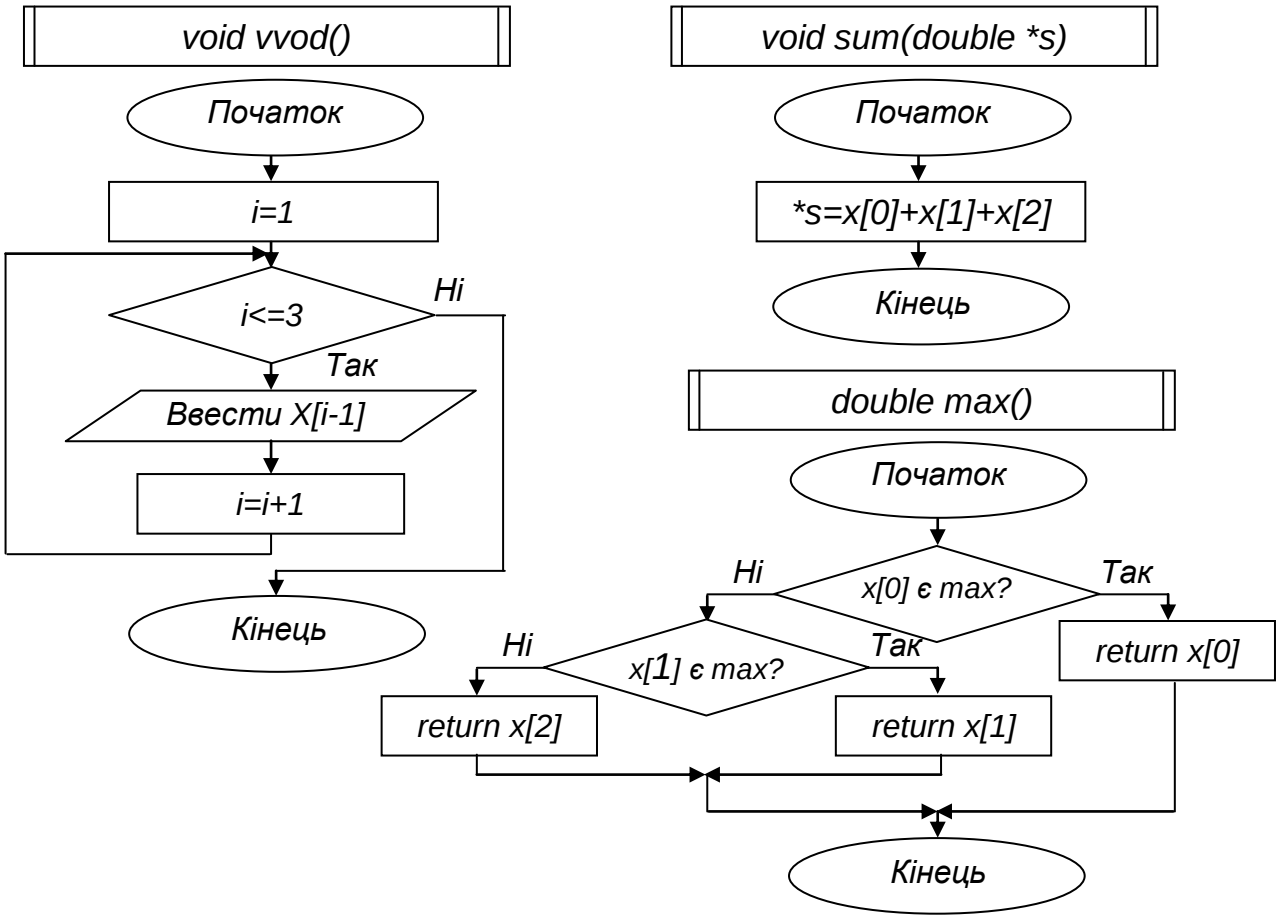


Рисунок 3.2 – Блок схеми підпрограм

Створюємо новий візуальний проект у Qt Creator з базовим класом QDialog. Конструюємо головне вікно нашої програми, наприклад як показано на рисунку 3.3. На формі розміщені 2 віджети: *Text Edit* та *Push Button*.

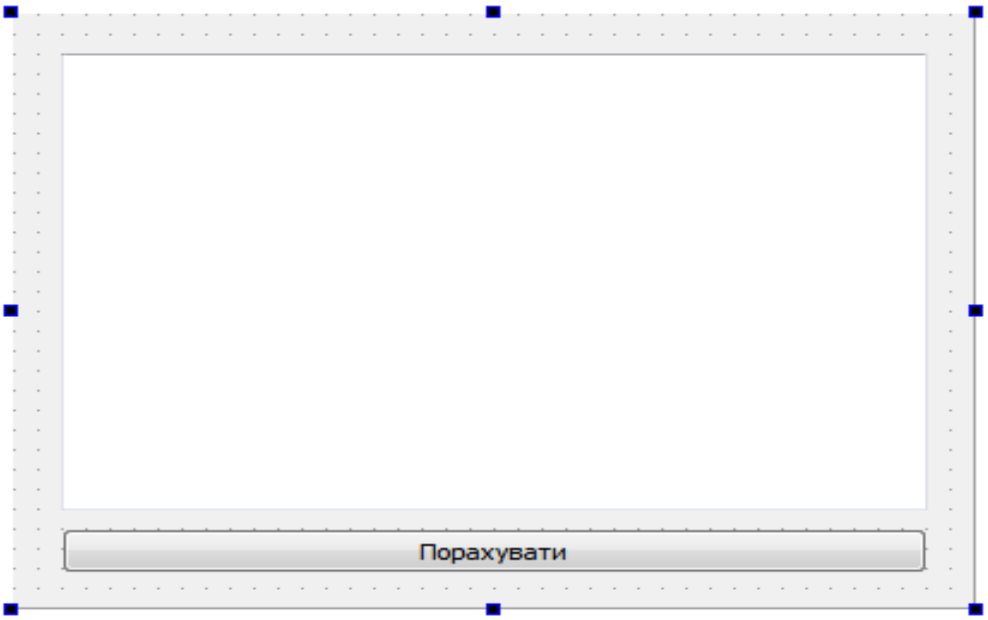


Рисунок 3.3 – Форма головного вікна програми

Реалізуємо клас з методами: знаходження максимального значення, введення масиву та обчислення суми масиву. В даному прикладі опис класу та його реалізація здійснюються в одному файлі. Файл dialog.h залишаємо без змін.

Вміст файлу dialog.cpp:

```
#include "dialog.h"
#include "ui_dialog.h"
#include <QTextCodec>
#include <QInputDialog>
#include <math.h>

Dialog::Dialog(QWidget *parent) :
        QDialog(parent), ui(new Ui::Dialog)
{
ui->setupUi(this);
QTextCodec::setCodecForCStrings(QTextCodec::codecForName("CP1251"));
QTextCodec::setCodecForTr(QTextCodec::codecForName("CP1251"));
QTextCodec::setCodecForLocale(QTextCodec::codecForName("CP1251"));
}

Dialog::~Dialog() { delete ui; }

class abc                          \\Початок опису нашого класу
{
public:
     double x[3];                        \\Три змінні
     QString name;                       \\Ім'я змінних

     abc(QString Nm="") {name=Nm;}  \\Конструктор
     ~abc() { }                          \\Деструктор

void vvod()                        \\Метод(процедура)введення
  {
    bool ok;
    for(int i=1; i<=3;i++)
      x[i-1]=QInputDialog::getDouble(0,"Введення значень "+name,
        QString("%1%2=").arg(name).arg(i), 3.8, -100, 1000, 5, &ok);
    }

void sum(double *s)            \\Метод(процедура)сумування
{
  *s=x[0]+x[1]+x[2];
 }

double max()              \\Метод(функція)знаходження максимального знач.
{
  if(x[0]>=x[1]&&x[0]>=x[2])return x[0]; else
    if(x[1]>=x[0]&&x[1]>=x[2])return x[1]; else return x[2];
 }
```

```
};                                  \\Кінець опису нашого класу

abc a("a"), b("b"), c("c");         \\Задавання статичних об'єктів нашого класу

void Dialog::on_pushButton_clicked() \\ Підпрограма обробки натискання кнопки
{
double e; abc t("t");
a.vvod();  b.vvod();  c.vvod();
if (fabs(a.max())>10) e=b.max()+c.max();
  else {
          a.sum(&t.x[0]); b.sum(&t.x[1]); c.sum(&t.x[2]);
          e=1+pow(t.max(),2);
        }
ui->textEdit->append(QString("a1=%1, a2=%2, a3=%3").arg(a.x[0]).arg(a.x[1]).
arg(a.x[2]));
ui->textEdit->append(QString("b1=%1, b2=%2, b3=%3").arg(b.x[0]).arg(b.x[1]).
arg(b.x[2]));
ui->textEdit->append(QString("c1=%1, c2=%2, c3=%3").arg(c.x[0]).arg(c.x[1]).
arg(c.x[2]));
ui->textEdit->append(QString("e=%1").arg(e));
 }
```

3.2 Методичні вказівки:

а) вивчити правила запису підпрограм різних типів та засобів звертання до них;

б) вивчити засоби передачі параметрів до підпрограми ;

в) вивчити особливості використання підпрограм у С++;

г) вивчити особливості задавання та виклику на виконання методів у класах при використанні ООП;

д)розробити візуальний інтерфейс для забезпеченні вирішення задачі свого варіанту;

е)на основі знань з ООП розробити клас та використати його для вирішення свого варіанту завдання;

ж) розробити алгоритми розв`язання задач свого варіанту, записавши їх у вигляді блок-схем;

з) підготувати текстовий варіант програми та попередніх даних.

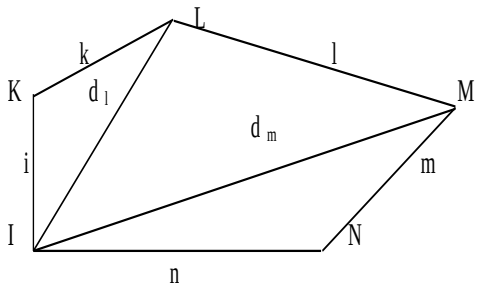
3.3 Контрольні запитання

1. Відмінності у підпрограмах мови С та С++.
2. Використання підпрограм в ООП.
3. Поняття про методи класу. Методи-функції, методи-підпрограми.
4. Формальні та фактичні параметри у методах.
5. Перезавантаження методів.
6. Виклик на виконання методів у статичних та динамічних об’єктах.

3.4 Варіанти завдань

Варіант 1

Задача1: Написати програму знаходження кутів багатокутника *IKLMN* зі сторонами *i,k,l,m,n* та діагоналями d_l d_m.



Вказівки: кут *A* трикутника виражається через його сторони *a,b,c* за формулою: $A=2\text{arctg}\frac{(p-b)(p-c)}{p(p-a)}$, де $p=\frac{a+b+c}{2}$ – напівпериметр. Попередні дані задавати самостійно.

Задача2: Обчислити наступні функції: $x=\frac{a^b+c^d}{a^c+b^d}$;$y=(x+7)^3+(x-7)^5$;

$w=\ln(x^a+y^b)$; $v=x^{\sin b}+(8-x)^{\cos b}$;

Вказівки: попередні дані задавати самостійно; операцію піднесення до ступеня оформити у вигляді функції.

Варіант 2

Задача1: Написати програму знаходження наближеного значення інтеграла $I=\int_{10}^{20}\frac{dx}{x}$, використовуючи наступну наближену формулу:

$$I=\int_{x_0}^{x_n}\mathrm{f}(x)dx\approx\sum_{i=0}^{n-1}\frac{h}{2}[\mathrm{f}(x_i)+\mathrm{f}(x_{i+1})],\ \text{де } h=x_{i+1}-x_i$$

Вказівки: прийняти *h=0.01;* знаходження підінтегральної функції оформити у вигляді підпрограми.

Задача2: Дано дійсні додатні числа *a,b,c*. Обчислити:

$$\frac{\max(c,a+b)+\max(a,b+c)}{1+\max(a+bc,b+ac,15)}.$$

Вказівки: попередні дані задавати самостійно; знаходження *max* оформити у вигляді аналога процедури.

Варіант 3

Задача1: Обчислити визначений інтеграл:

$I=\int_b^a \sqrt{2x+1}dx$, за формулою: $I=\frac{b-a}{b}[\mathrm{f}(a)+4\mathrm{f}(\frac{b+a}{b})+\mathrm{f}(b)]$.

Вказівки: попередні дані задавати самостійно; знаходження підінтегральної функції оформити у вигляді підпрограми.

Задача2: Обчислити величину:

$y=\frac{2\,\mathrm{th}(1/2)-3\,\mathrm{th}(t-1/10)}{5-\mathrm{th}(4t-1)}$, де значення гіперболічного тангенса обчислюється за формулою:

$$th(x)=\frac{\sum_{k=0}^{5} x^{2k+1}/(2k+1)!}{\sum_{k=1}^{5} x^{2k}/(2k)!}.$$

Вказівки: обчислення значень факторіала та th*(x)* оформити у вигляді аналога процедури; значення *t* задавати самостійно.

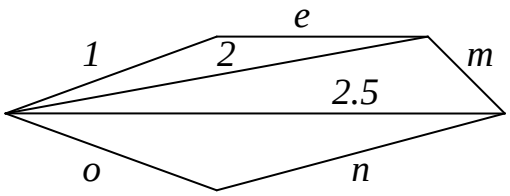
Варіант 4

Задача1: Задані дійсні числа $a_1,a_2,a_3;\ b_1,b_2,b_3;\ c_1,c_2,c_3$. Отримати:

$$e=\begin{cases}\min(b_1,b_2,b_3)+\min(c_1,c_2,c_3), & \text{якщо } |\min(a_1,a_2,a_3)|>10,\\ 1+[\min(\sum_{i=1}^{3}a_i,\sum_{i=1}^{3}b_i,\sum_{i=1}^{3}c_i)]^2, & -\text{ в іншому випадку.}\end{cases}$$

Вказівки: попередні дані задавати самостійно; знаходження min оформити у вигляді функції, а обчислення $\sum$ та введення a_i ,b_i, c_i– у вигляді аналога процедур.

Задача2: Задані дійсні числа *e,m,n,o*. Знайти площу п`ятикутника:



Вказівки: значення *e,m,n,o* задавати самостійно; знаходження площі трикутника за формулою Герона $S=\sqrt{p(p-a)(p-b)(p-c)}$, де *p=(a+b+c)/2* оформити у вигляді підпрограми.

Варіант 5

Задача1: По заданим дійсним числам $a_0, a_1, \ldots a_{10}; b_0, b_1, \ldots b_{10}; c_0, c_1, \ldots c_{10}; x, y, z$ обчислити величину: $\frac{(a_0x^{10}+a_1x^9+\cdots+a_{10})^2-(b_0y^{10}+b_1y^9+\cdots+b_{10})}{c_0(x+y)^{10}+c_1(x+y)^9+\ldots.+c_{10}}$

Вказівки: попередні дані задавати самостійно; введення елементів векторів задати у вигляді процедури; знаходження ступеневого багаточлена

задати у вигляді функції, яка реалізує схему Горнера: $p=a_0x^n+ a_1x^{n-1} +...+ a_{n-1}x+a_n=(...((a_0x+a_1)x+a_2)x+...+a_{n-1})x+a_n));$

Задача 2: Створити підпрограми для обчислення опору паралельного та послідовного з'єднання *n* резисторів. В якості параметрів підпрограм використати кількість резисторів і масив з номіналів опорів цих резисторів.

Вказівки: За допомогою створених підпрограм порахувати паралельне та послідовне з'єднання наступних резисторів:

*а)*1.5, 330, 200, 1500; *б)* 2200, 390, 1000, 4700, 56000, 750; *в)*8200, 430, 150;

Варіант 6

Задача1: Дано дійсні числа $x_1,y_1,....x_{10},y_{10}.$ Знайти периметр десятикутника, вершини якого мають відповідні координати: $(x_1,y_1),....(x_{10},y_{10})$.

Вказівки: операцію знаходження довжини між двома точками, які задані своїми координатами, оформити у вигляді процедури.

Задача 2: Створити підпрограму для переведення числа із двійкової до десяткової системи числення.

Вказівки: двійкове число задавати як одномірний масив.

Варіант 7

Задача 1: Вирахувати вираз:

$$k=\begin{cases} 2\cdot x^5+\sum\limits_{i=0}^{X}(2\cdot i+\dfrac{y}{i+1}), & \text{якщо } x\le 10; \\ y^{20}-x^6-\sum\limits_{i=10}^{X}(8\cdot i-\dfrac{x}{i}), & \text{якщо } x>10. \end{cases}$$

Вказівки: вирахування суми оформити у вигляді підпрограми; значення *x*, *y* вводити з клавіатури ($x>0$).

Задача 2:

Вирахувати величину $g=\dfrac{2\cdot \mathrm{k}(x,y,z)+\mathrm{k}^2(x,y,z)}{(\mathrm{k}(x,y,z)-\mathrm{k}^2(x,y,z))^2}$, де

$$\mathrm{k}(a,b,c)=\begin{cases} \dfrac{a\cdot b\cdot c}{a-b-c}, & \text{якщо } a>b>c; \\ \dfrac{a\cdot b\cdot c}{a+b+c}, & \text{в інших випадках}. \end{cases}$$

Вказівки: вирахування k*(a, b, c)* оформити у вигляді аналога процедури; значення *x, y, z* вводити з клавіатури.

Варіант 8

Задача 1: Вирахувати вираз: $\mathrm{p}=\dfrac{2\cdot \mathrm{x}^5+4\cdot \mathrm{x}^{30}+8\cdot \mathrm{y}!}{2\cdot \mathrm{y}+\mathrm{y}^{15}-25\cdot \mathrm{x}!}$.

Вказівки: піднесення до ступеня та знаходження факторіалу оформити у вигляді підпрограми; значення *x* та *y* вводити з клавіатури.

Задача 2: Вирахувати вираз: $y = \frac{2 \cdot \mathrm{th}(\frac{1}{2}) - 25 \cdot \mathrm{th}(t - \frac{1}{5})}{5 - \mathrm{th}(3 \cdot t - 2)}$, де значення гіперболічного тангенсу вирахувати по формулі:

$$\mathrm{th}(x) \approx \frac{\sum_{k=0}^{6} \frac{x^{2k+1}}{(2k+1)!}}{\sum_{k=1}^{6} \frac{x^{2k}}{2k!}}.$$

Вказівки: вираховування значення факторіалу оформити у вигляді підпрограми; вираховування значення *th(x)* оформити у вигляді аналога процедури; значення *t* вводити з клавіатури.

Варіант 9

Задача 1: Дано дійсні числа *a, b*. Знайти: *u=max(a, b), v=max(ab, a+b), c=max(*u^5+v^5*, 256).*

Вказівки: попередні дані задавати самостійно; знаходження *max* оформити у вигляді підпрограми.

Задача2: Змінній *t* присвоїти значення *1*, якщо рівняння $x^2+6.2x+a^2=0$ та $x^2+ax+b-1=0$ мають дійсні корені та в той-же час обидва корені першого рівняння лежать між коренями другого, і присвоїти значення *0* в усіх інших випадках.

Вказівки: *a,b* задавати самостійно так, щоб перевірити усі можливі випадки. Вирішення будь-якого квадратного рівняння оформити у вигляді підпрограми-процедури.

Варіант 10

Задача 1: Задані два комплексні числа (a+ib) і (c+id) і тип операції (додавання, множення, віднімання, ділення). Створити функції для виконання арифметичних операцій над комплексними числами і вирахувати значення модулів заданих комплексних чисел.

Вказівки: *a, b, c* та *d* вводити з клавіатури.

Задача2: Дано дійсні числа *s,t*. Отримати g*(1.2, s)* + g*(t, s)* − g*(2s−1, st)*, де

$$g(a,b) = \frac{a^2 + b^2}{a^2 + 2ab + 3b^2 + 4}$$

Вказівки: попередні дані задавати самостійно; знаходження функції *g(a, b)* оформити у вигляді підпрограми.

4 ЛАБОРАТОРНА РОБОТА №4. ОБРОБКА СИМВОЛЬНИХ ДАНИХ

Мета роботи: отримання навичок з алгоритмізації та програмування задач, які оброблюють символьні дані; освоїти введення та виведення символьних даних, їх оброблення; навчитися використовувати стандартні функції оброблення символьних даних у С++ та візуальному режимі з використанням ООП.

4.1 Теоретичні відомості

У С++ можна працювати як зі звичайні рядки з мови С (масив із символів), так і з об'єкти класу string. Також використовуються і специфічні рядкові класи системи програмування, наприклад QString у Qt.

Об’єкти класу string.

У Qt з класом ***string*** можна працювати в області імен ***std***:

using namespace std;

<Задавання рядка>::= ***string*** <ім'я рядку> | ***string*** <ім'я рядку>("Рядок ініціалізації") | ***string*** <ім'я рядку> ="Рядок ініціалізації " | ***string*** <ім'я рядку>(<рядок>) | ***string*** <ім'я рядку>(<рядок С>, n)

В кінці не містять нульового символу.

Приклад:

string s;

string s1("Приклад рядка");

string s2="Ще один рядок";

s, s1, s2 об'єкти класу string

Введення-виведення рядків.

Консольний режим:

- операції << і >> для роботи з потоками.
- функція getline(<потік>,<рядок>).

Візуальний режим:

Введення з lineEdit: *s=ui->lineEdit->text().toStdString();*

Виведення у textEdit: *ui->textEdit->append(QString().fromStdString(s));*

Аналогічно і з іншими об'єктами QString.

Операції для роботи з рядками:

=	-	присвоєння,
+	-	об'єднання (додавання),
==	-	дорівнює,
!=	-	не дорівнює,
<	-	менше,
<=	-	менше чи дорівнює,
>	-	більше чи дорівнює,
>=	-	більше чи дорівнює,
[ ]	-	індексація,
<<	-	виведення,
>>	-	введення,
+=	-	додавання в кінець рядка.

Присвоювати у рядок можна символ, рядкову константу або змінну.

Приклад:

s='A';

s="Рядок символів";

s1=s2;

Особливості роботи з рядками у С++:

- розмір рядка встановлюється автоматично так щоб помістилися всі символи;
- адреса першого символу не відповідає адресі рядка. (&s[0] не дорівнює s);
- отримати доступ до символу у рядку крім [i] можна і за допомогою методу at(i).

Приклад: *s[4]==s.at(4);*

Методи для роботи з рядками.

assign - присвоєння частини одного рядка іншому.

assign(<рядок>); (те ж саме що і =);

assign(<рядок>,<позиція>, <к-ть символів>);

assign(<рядок С>, <к-ть символів>).

Приклад:

s.assign(s1); // те ж саме що і s=s1;

s.assign(s1,5,3); // присвоює у s 3 символи з s1 починаючи з 6-го символу

append - додавання частини одного рядка до іншого.

append(<рядок>); (те ж саме що і +=);

append(<рядок>,<позиція>, <к-ть символів>);

append(<рядок С>, <к-ть символів>).

Приклад:

s.append(s1); // те ж саме що і s+=s1 (s=s+s1);

s.append(s1,5,3); // додає до s 3 символи з s1 починаючи з 6-го символу

insert – вставляє частину одного рядка до іншого.

insert(<позиція>, <рядок>);

insert(<поз1>,<рядок>, <поз2>, <к-ть символів>);

insert (<рядок С>, <позиція>, <к-ть символів>).

Приклад:

s.insert(5,s1); // вставляє у s з 5-ї позиції рядок s1;

s.insert(5,s1,6,3); // вставляє у s з 5-ї позиції 3 символи рядку s1 починаючи з 6-го символу

erase – видалення частини рядка.

erase(<позиція>=0, <к-ть символів>=max);

Видаляє з рядка <к-ть символів> елементів, починаючи з позиції <позиція>. Якщо не вказана <к-ть символів>, видаляється увесь залишок рядка.

Приклад:

s.erase(5,4); // видаляється з s з 5-ї позиції 4 символи;

s.erase(5); // видаляється з s кінець рядка починаючи з 5-ї позиції

s.erase(); // видаляється весь рядок s (очищується). Аналог: s.clear();

replace – заміна частини рядка.

replace(<позиція>, <к-ть символів>, <рядок>);

replace (<поз1>, <к-ть симв1>, <рядок>, <поз2>, <к-ть симв2>);

replace (<поз1>, <к-ть симв1>, <рядок С>, <к-ть симв2>).

Приклад:

s.replace(5, 3, s1, 4, 2); // заміняє у s з 5-ї позиції 3 символи на 2 символи з рядка s1 з 4-ї позиції

swap – обмін вмістом двох рядків.

swap(<рядок>);

Приклад: *s.swap(s1); // вміст s і s1міняється місцями*

substr – отримання частини рядка.

substr(<позиція>=0,<к-ть символів>=max);

Приклад:

s1=s.substr(6,3); // у s1записується 3 символи з s починаючи з 6-ї позиції.

s1=s.substr(6); // у s1записується кінець рядка s починаючи з 6-ї позиції.

c_str – повертає рядок **C**(масив символів з \0 у кінці).

c_str();

Приклад: *s.c_str(); // повертається рядок мови С еквівалентний s*

data – повертає рядок **C**(масив символів) але без \0 у кінці.

data();

Приклад: *s.data(); // повертається рядок мови С без \0 у кінці, еквівалентний s*

copy – копію у рядок **C**(масив символів) частину рядка, повертає к-ть скопійованих символів, \0 в кінець не записується.

copy(<рядок С>, <к-ть символів>, <позиція>=0);

Приклад:

n=s.copy(sc,7,5); // копіює у sc 7 символів рядка s з 5-ї позиції. У n записується 7;

n=s.copy(sc,7); // копіює у sc 7 символів початку рядка s. У n записується 7.

Пошук у рядку

Існує багато методів пошуку, основні: **find, rfind, find_first_of, find_last_of, find_first_not_of, find_last_not_of.**

find(<підрядок>|<символ>, <позиція>= 0) - шукає саме ліве входження рядка <підрядок> або символу, починаючи з позиції <позиція>, і повертає знайдену позицію або max, якщо підрядок чи символ не знайдений.

rfind(<підрядок>|<символ>, <позиція>= max) - шукає саме праве входження рядка <підрядок> або символу, починаючи з позиції <позиція>, і повертає знайдену позицію або max, якщо підрядок чи символ не знайдений.

find_first_of(<підрядок>|<символ>, <позиція>= 0) - шукає саме ліве входження будь-якого символу з рядку <підрядок> або одного символу, починаючи з позиції <позиція>, і повертає знайдену позицію або max, якщо підрядок чи символ не знайдений.

find_last_of(<підрядок>|<символ>, <позиція>= 0) - шукає саме праве входження будь-якого символу з рядку <підрядок> або одного символу, починаючи з позиції <позиція>, і повертає знайдену позицію або max, якщо підрядок чи символ не знайдений.

find_first_not_of(<підрядок>|<символ>, <позиція>= 0) - шукає саму ліву позиці, починаючи з позиції <позиція>, для якої жоден символ з рядку <підрядок> або один символ не співпадає з початковим рядком чи символом початкового рядку.

compare – порівняння частин рядків.

compare(<рядок>);

compare(<позиція>, <к-ть символів>, <рядок>);

compare(<поз1>, <к-ть сим1>, <рядок>, <поз2>, <к-ть сим2>);

Повертають значення, менше 0, якщо початковий рядок менше <рядок>, дорівнює нулю, якщо рядки одинакові, і більше нуля — якщо початковий рядок більше. Працює аналогічно функції strstr з бібліотеки C.

Отримання характеристик рядків.

size() - кількість елементів у рядку.

length() - кількість елементів у рядку.

max_size() – максимально можлива довжина рядку (max).

capacity() - об'єм пам'яті, що займає рядок.

empty() - true, якщо рядок пустий.

Приклад:

Задача: Написати програму, яка:

а) створює каталог виробів, які зберігаються на складі. Кожен запис каталогу є рядок і містить шифр деталі, кількість та місце знаходження. По введеному шифру ЕОМ повідомляє про кількість виробів на складі та їх місцезнаходження;

б) вносить зміни про кількість деталей, які знаходяться на складі, під час їх видачі покупцю.

Блок схема програми зображена на рисунку 4.1.

(Буде пізніше)

Рисунок 4.1 – Блок схема програми

Створюємо новий візуальний проект у Qt Creator з базовим класом QDialog. Конструюємо головне вікно нашої програми, наприклад як показано на рисунку 4.2. На формі розміщені 3 віджети: *Label,* 2 віджети *LineEdit,* та *PushButton*, віджет *textEdit*

label

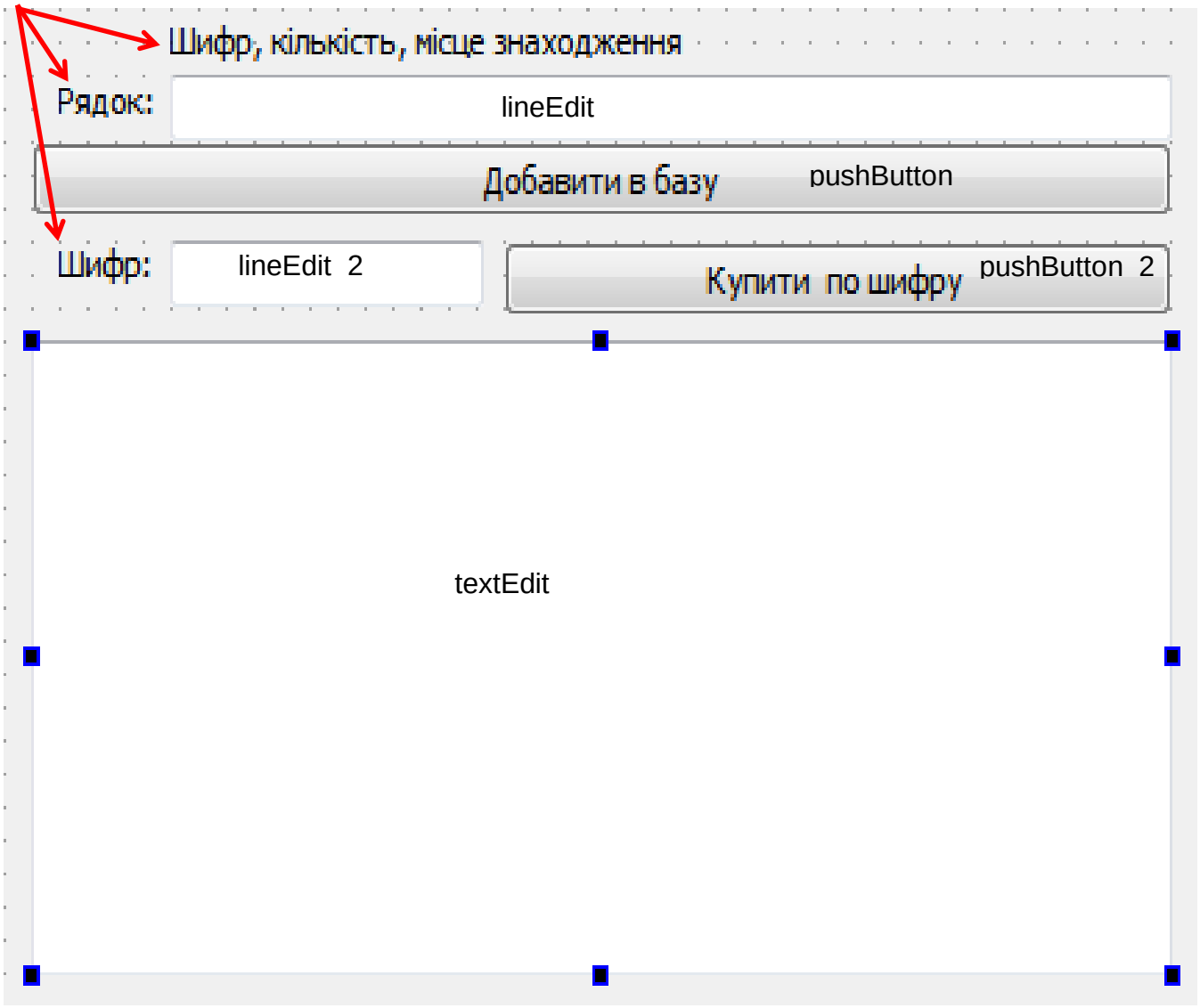


Рисунок 4.2 – Форма головного вікна програми

В прикладі опис класу та його реалізація здійснюються в одному файлі.

Вміст файлу dialog.h:

```
#include "dialog.h"
#include "ui_dialog.h"
#include <QTextCodec>
#include <QInputDialog>
using namespace std;
Dialog::Dialog(QWidget *parent) :
QDialog(parent), ui(new Ui::Dialog)
{
  ui->setupUi(this);
 QTextCodec::setCodecForCStrings(QTextCodec::codecForName("CP1251"));
 QTextCodec::setCodecForTr(QTextCodec::codecForName("CP1251"));
 QTextCodec::setCodecForLocale(QTextCodec::codecForName("CP1251"));
```

```
}
Dialog::~Dialog() { delete ui; }

string s[100];
int n=0;

void Dialog::on_pushButton_clicked()
{
  if (ui->lineEdit->text().isEmpty()!=true)
    if (n<100)
     {
       s[n++]=ui->lineEdit->text().toStdString();
       ui->textEdit->append(QString().number(n)+". "+
                                        QString().fromStdString(s[n-1]));
       ui->lineEdit->clear();
      }
   else ui->textEdit->append("База переповнена!");
}

void Dialog::on_pushButton_2_clicked()
 {
   string shifr, result;
   char res[100];
   int k1,k2,nom;
   long ekil;
   bool nema=true, ok;
   long kil;
   shifr=ui->lineEdit_2->text().toStdString();
   for(int i=0; i<n&&nema; i++)
    {
         if ( s[i].compare(0, k1=s[i].find(','), shifr)==0 )
         {
         nema=false;
         nom=i;
         k2=s[i].find(',',k1+1);
         ekil=atol(s[i].substr(k1+1,k2-k1-1).c_str());
         result="На складі є "+s[i].substr(k1+1, k2-k1-1)+
          " деталей шифру \""+shifr+"\", знаходяться: "+
                                 s[i].substr(k2+1);
         ui->textEdit->append(QString().fromStdString(result));
          }
   }
   if (nema) ui->textEdit->append("На складі нема деталі
             шифру "+QString().fromStdString('\"'+shifr+'\"'));
  else
   {
    kil=QInputDialog::getInt(0,"Видача зі складу деталей",
```

```
        "Взяти "+QString().fromStdString('\"'+shifr+'\"')+
        " із "+QString().number(ekil)+" :", 1, 0, ekil, 0, &ok);
    itoa(ekil-kil, res, 10);
    s[nom].replace(k1+1, k2-k1-1, res);
    ui->textEdit->append(QString().number(nom+1)+
                         ". "+QString().fromStdString(s[nom]));
    }
}
```

4.2 Методичні вказівки

а) вивчити правила запису символьних даних (констант, змінних, масивів, рядків) та опис способу їх введення та виведення у Сі++;

б) ознайомитися з основними стандартними функціями, які дозволяють оброблювати символьні рядки. Звернути увагу на виклик підпрограм та на типи формальних параметрів;

в) розробити візуальний інтерфейс для забезпеченні вирішення задачі свого варіанту;

г) на основі знань з ООП розробити свій клас або обрати один із стандартних класів по роботі з рядками та використати його для вирішення свого варіанту завдання;

д) розробити алгоритми розв`язання задач свого варіанта, записавши їх у вигляді блок-схем;

4.3 Контрольні запитання

1. Робота з рядками у С++.
2. Об’єкти класу **string**.
3. Введення та виведення рядків у С++ та візуальному режимі.
4. Операції для роботи з рядками у С++.
5. Методи для роботи з рядками.
6. Методи для пошуку у рядку.
7. Методи для отримання характеристик рядків.

4.4 Варіанти завдань

Варіант1

Задача1: Написати програму, яка реалізує епізод казки: ЕОМ питає, куди бажає піти герой (наліво, направо чи прямо), та виводить на екран, що його чекає в кожному випадку.

Вказівки: відповідь ЕОМ присвоїти символьній(рядковій) змінній та вивести на екран; текст питань та відповідей ЕОМ задавати самостійно.

Задача2: Задати інформацію для групи студентів у вигляді одномірного рядкового масиву, кожний елемент якого містить наступні данні: прізвище,

ім`я, по батькові, рік народження. Визначити середній вік студентів та прізвище самого старшого.

Вказівки: результат присвоїти двом рядковим змінним та вивести на екран. (Наприклад: середній вік – 18 років. Старший – Іванов – 20 років)

Варіант 2

Задача1: У кіоску продається газета, яка коштує 1гр. та журнал, який коштує 2гр. Написати програму, яка питає про бажання покупця (журнал чи газета), приймає гроші (сума грошей вводиться з клавіатури) та виводить на екран належну здачу.

Задача2: Задати інформацію про результати сесії для групи студентів у вигляді одномірного масиву, кожний елемент якого містить інформацію про одного студента (прізвище та п`ять оцінок через кому).

Обчислити середній бал кожного студента та створити новий рядковий масив, кожний елемент якого містить інформацію у вигляді: середній бал Іванова - 4.2.

Варіант 3

Задача1: У масиві із n рядків, кожна з яких містить ім`я та прізвище, знайти тих, кого звуть Вася.

Задача 2: В одномірному масиві із *n* рядків, кожний з яких містить номінал резистора, перевести його зі скороченого вигляду у повний (МОм = $x10^6$, кОм = $x10^3$ і т.д. Наприклад: ‘3,3 ГОм’ = ‘3300000000 Ом’), а також знайти максимальний і мінімальний опір.

Вказівки: рядки та *n* задати самостійно.

Варіант 4

Задача1: Визначити, чи входить слово ”*день*” у рядок, який складається з декількох слів. Якщо так, то отримати новий рядок, який відповідає старому, але з якого виключені всі слова “*день*”.

Задача2: Задати рядковий масив, кожний елемент якого містить наступну інформацію: прізвище складальника деякого виробу даного цеху та число виробів, які той зібрав кожного дня за тиждень (через кому). Написати програму, яка видавала б значення рядкової змінної (яка визначена у програмі), що містить прізвище складальника, загальну кількість виробів за тиждень та середнє число виробів, що складалися за день (для кожного складальника). Визначити та вивести на екран прізвище того працівника, який досягнув найвищої продуктивності праці.

Варіант 5

Задача1: Написати програму, яка питає ім`я, порівнює з тими, що вона має як елементи символьного масиву та вітає або повідомляє, що “не знайома”.

Задача2: Написати програму, яка виконує наступні фінансово-економічні розрахунки. Дані у вигляді місячної заробітної плати робочих зі спеціальностями різних категорій занести в одномірний масив рядків, кожен з

яких (рядок) містить: прізвище, категорію, місячний заробіток, номер цеху. Обчислити загальну суму виплат за місяць по всіх категоріях, по категоріях окремо, відсоток по категоріях від загальної суми, середню місячну заробітну плату по категоріях.

Варіант 6

Задача 1: Розробити програму, котра в текстовому рядку замінює будь-яку кількість однакових символів, що йдуть один за одним підряд на один такий же символ та цифру, яка відповідає кількості видалених символів (Наприклад:
‘*1CABk3KKK111DeFf0100fk0cccccc’=’1CABk3K212DeFf0101fk0c5*’).

Вказівки: початковий рядок вводити з клавіатури; отриманий рядок виводити під початковим рядком.

Задача 2: В одномірному масиві із *n* рядків, кожний з яких містить прізвище та ім’я через пробіл, знайти та вивести прізвища тільки тих, у кого ім’я складається з 4 літер.

Варіант 7

Задача1: Дано масив з *n* рядків. В кожному рядку замінити всі знаки оклику крапками.

Задача2: Написати програму, яка:

а) створює каталог виробів, які зберігаються на складі. Кожен запис каталогу є рядок і містить шифр деталі, кількість та місце знаходження. По введеному шифру ЕОМ повідомляє про кількість виробів на складі та їх місцезнаходження;

б) вносить зміни про кількість деталей, які знаходяться на складі, під час їх видачі покупцю.

Варіант 8

Задача 1: В масиві, який складається з рядків, кожен з яких містить прізвище, ім’я та по-батькові через пробіли, знайти й вивести на екран тих, у кого прізвище Іванов чи Іванова.

Вказівки: початковий масив задати самостійно.

Задача 2: В символьному рядку видалити всі непарні числа, а парні числа взяти в круглі дужки (наприклад: початковий рядок ‘*Abc18Cd056k-!B,10*’, результуючий рядок :*’Abc(8)Cd(0)(6)k-!B,(0)’*).

Варіант 9

Задача 1: Розробити програму, котра в текстовому рядку замінює будь-яку кількість прогалин, що йдуть підряд на одну прогалину і переводить всі великі літери, крім першої, у маленькі, а цифри видаляє. (Наприклад: ‘*Я1СА ВкЗльК1De 0100fk0 cccc’=’Яса bкльk1de fk ccc*’).

Вказівки: початковий рядок вводити з клавіатури; отриманий рядок виводити під початковим рядком.

Задача 2: В одномірному масиві із *n* рядків, кожний з яких містить номінал котушки індуктивності, перевести його зі скороченого вигляду у повний (мкГн = x10^{-6}, мГн = x10^{-3} і т.д. Наприклад: ‘120 мкГн’ = ‘0.00012 Гн’), а також знайти суму всіх індуктивностей у масиві.

Варіант 10

Задача 1: В заданому рядку знайти найкоротше і найдовше слово та вказати номери позицій, з яких вони починаються.

Вказівки: початковий рядок задати самостійно.

Задача2: У продажу книг у книжковій крамниці приймає участь ЕОМ. Написати програму, яка питає у покупця назву книги, яку той бажає купити. Якщо книга є у продажу, то повідомляє, скільки та коштує, приймає гроші (сума грошей вводиться з клавіатури). Далі ЕОМ визначає належну здачу (якщо грошей внесено більше); виводить на екран “Дякую”, якщо здача не потрібна; або виводить повідомлення про недостачу коштів.

Вказівки: попередні дані задавати у вигляді масиву рядків, кожний з яких містить наступну інформацію: прізвище автора, назву, ціну.

5 ЛАБОРАТОРНА РОБОТА №5. РОБОТА ЗІ СТРУКТУРАМИ ТА ФАЙЛАМИ

Мета роботи: отримання навичок з алгоритмізації та програмування задач з використанням файлових структур даних; освоїти проектування структури файлу, виведення даних до файлу та читання даних з файлу; отримання навичок з організації введення/виведення значень структурних типів даних; опанування практичними навичками програмування задач з використанням структур у С++ та візуальному режимі з використанням ООП.

5.1 Теоретичні відомості

У С++ працювати з файлами можна як у класичному С, так і з використанням класів та об’єктів для роботи з потоками.

Потоки для роботи з файлами.

Крім стандартних потоків введення-виведення (cin, cout, cerr, clog) можна задати свої потоки і зв'язати їх з будь яким файлом.

Працювати з потоками(файлами) можна як в послідовному доступі (текстові файли, читання/запис по черзі) так і в прямому доступі (двійкові файли, читання/запис з будь якого місця).

Все що раніше ми розглядали для потокового введення/виведення з клавіатури/монітору справедливо і для потокового введення/виведення з будь якого файлу.

Існують три класи для роботи з файлами:

ifstream — клас вхідних файлових потоків;

ofstream — клас вихідних файлових потоків;

fstream — клас двонаправлених файлових потоків.

Ці класи є похідними від класів *istream, ostream* і *iostream* відповідно, тому вони наслідують перезавантажені операції << і >>, прапорці форматування, маніпулятори, методи, стани потоків і т. д.

Використання файлів у програмі передбачає наступні дії:

- створення потоку;
- відкривання потоку і зв’язування його з файлом;
- обмін даними(введення/виведення);
- знищення потоку;
- закривання файлу.

Кожний клас файлових потоків містить конструктори, за допомогою яких можна створити об’єкти цих класів різними способами.

Створення файлових потоків.

Необхідно підключити заготовочний файл та задати простір імен:

#include <fstream>

using namespace std;

Конструктори без параметрів створюють об’єкт відповідного класу але не зв’язують його з файлом:

ifstream <ім'я потоку>;

ofstream <ім'я потоку>;

fstream <ім'я потоку>;

Конструктори з параметрами створюють об’єкт відповідного класу, відкривають файл у відповідному режимі з вказаним ім’ям і зв’язують файл з об’єктом:

ifstream <ім'я потоку>(<ім'я файлу>, <режим>= ios::in);

ofstream <ім'я потоку>(<ім'я файлу>,<режим>= ios::out | ios::trunc);

fstream <ім'я потоку>(<ім'я файлу>,<режим>= ios::in |ios::out);

Приклад:

```
#include <fstream>
using namespace std;
.....
string filename(“d://123.dat");
.....
ifstream in_stream;
ofstream out_stream("1.txt");
fstream inout(filename.c_str(),ios::out);
```

in_stream, out_stream, inout - об'єкти-потоки файлового введення виведення класу fstream.

(Далі буде.)

5.2 Методичні вказівки

а) вивчити основну термінологію, яка пов`язана з файловими структурами даних: файл та його структура, фізичний та логічний записи, методи доступу, поточний вказівник файлу у С++;

б) вивчити стандартні функції, які забезпечують основні операції з бінарними та тестовими файлами у С++ та візуальному режимі;

в) засвоїти правила роботи зі структурами та сумішами у С++

г) розробити візуальний інтерфейс для забезпеченні вирішення задачі свого варіанту;

д)на основі знань з ООП розробити клас та використати його для вирішення свого варіанту завдання;

е) розробити алгоритми розв`язання задач свого варіанта, записавши їх у вигляді блок-схем;

ж) підготувати текстовий варіант програми та попередніх даних. Провести відлагодження програми, перевіривши всі можливі ситуації.

5.3 Контрольні запитання

1..Робота з файлами у С++.

2..Потоки для роботи з файлами.

3..Створення файлових потоків.

4.Відкривання файлів.

5..Читання та запис файлів.

6. Введення та виведення потоків.
7. Методи читання потоків.
8. Методи запису потоків.
9. Помилки потоків.
10. Методи отримання стану потоків.

5.4 Варіанти завдань

Варіант1

Задача А: Створити файл, який містить інформацію про особисту колекцію книголюба. Структура запису: шифр книги, автор, рік видання, місцезнаходження (номер стелажу, шафи та т.і.). Кількість записів довільна.

Задача В: Написати програму, яка видає наступну інформацію:

— місцезнаходження книги автора *A* назви *B*. Значення *A,B* ввести з клавіатури;

— список книг автора *C*, які знаходяться в колекції;

— кількість книг видання *X* року, які знаходяться в колекції.

Варіант 2

Задача А: Створити файл-довідник, який містить дані про біполярні транзистори. Структура запису: марка, провідність (n-p-n, p-n-p), максимальний струм колектора, максимальна напруга колектор-емітер, мінімальний і максимальний коефіцієнти підсилення (h21e), максимальна робоча частота. Кількість записів довільна.

Задача В: Написати програму, яка дозволяє шукати у довіднику:

— всю інформацію по введеній марці транзистора з клавіатури;

— по введеному з клавіатури струму, напрузі і коефіцієнту підсилення видати всі підходящі транзистори;

— видати всі комплементарні пари транзисторів (у яких параметри однакові, а провідність різна).

Варіант 3

Задача А: Утворити файл, який містить інформацію про співробітників університету. Структура запису: прізвище працюючого, назва відділу, рік народження, стаж роботи, посада, оклад. Кількість записів довільна.

Задача В: Написати програму, яка видає дозволяє отримати наступну інформацію:

— список працівників пенсійного віку на сьогоднішній день з зазначенням стажу роботи;

— середній стаж працюючих у відділі *X*.

Варіант 4

Задача А: Утворити файл, який містить інформацію про пацієнтів дитячої клініки. Структура запису: прізвище пацієнта, стать, вік, місце проживання (місто), діагноз. Кількість записів довільна.

Задача В: Написати програму, яка видає наступну інформацію:

— кількість пацієнтів, які прибули до клініки з іншого міста;

— список пацієнтів старших *X* років з діагнозом *Y*. Значення *X,Y* ввести з клавіатури.

Варіант 5

Задача А: Утворити файл, який містить інформацію про здачу студентами сесії. Структура запису: індекс групи, прізвище студента, оцінки з п`яти екзаменів та п`яти заліків (“*З*” – зараховано, “*Н*” – не зараховано). Кількість записів довільна.

Задача В: Написати програму, яка видає наступну інформацію:

— прізвища невстигаючих студентів з вказівкою індексів груп та кількостей заборгованостей;

— середній бал, отриманий кожним студентом групи *X*, та всією групою в цілому.

Варіант 6

Задача А: Утворити файл, який містить інформацію про асортимент взуття в крамниці фірми. Структура запису: артикул, назва, кількість, ціна однієї пари. Кількість записів довільна. Артикул починається з літери *Ж* для жіночого взуття, *Ч* – чоловічого, *Д* – дитячого.

Задача В: Написати програму, яка видає наступну інформацію:

— про наявність та ціну взуття артикула *X*;

— асортиментний список жіночого взуття з вказівкою назви та кількості пар кожної моделі, яка є у продажу.

Варіант 7

Задача А: Створити файл-довідник, який містить дані про напівпровідникові діоди. Структура запису: марка, максимальний струм, максимальна зворотна напруга, падіння напруги у відкритому стані, максимальна робоча частота. Кількість записів довільна.

Задача В: Написати програму, яка дозволяє шукати у довіднику:

— всю інформацію по введеній марці діода з клавіатури;

— по введеному з клавіатури струму, зворотній напрузі і частоті видати всі підходящі діоди;

— видати всю інформацію про діоди з падінням напруги у відкритому стані менше, ніж введено з клавіатури.

Варіант 8

Задача А: Створити файл, який містить значення функції sin(x), cos(x), tg(x) коли *x* змінюється від 0 до 314 з кроком 0.5.

Задача В: Написати програму, яка у файлі, шукає від’ємні елементи, і коли вони є, то виводить їх на екран. Коли від’ємних елементів немає, на екран вивести перший та останній елементи.

Варіант 9

Задача А: Створити файл, який містить інформацію про наявність квитків і рейсів Аерофлоту. Структура запису: номер рейсу, пункт призначення, час вильоту, час прибуття, кількість вільних місць у салоні. Кількість записів довільна.

Задача В: Написати програму, яка видає інформацію наступного типу:

— час відправлення літаків у місто *X*;

— наявність вільних місць на рейс у місто *X* ;з часом відправлення *Y*.

Вказівки: значення *X,Y* вводиться по запиту з клавіатури.

Варіант 10

Задача А: Написати програму яка створює файл, що містить інформацію про розклад телепрограм на день. Структура запису: назва програми, час початку програми, час закінчення програми.

Задача В: Написати програму, яка дозволяє отримати наступну інформацію:

— усю програму телепередач на день;

— по введеному з клавіатури часу видати назву програми, котра буде транслюватися в цей час;

— назву самої довгої та самої короткої (за тривалістю) телепрограми.

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Р.Лафорте. Объектно-ориентированное программирование в C++, 4-е изд. — Питер, — 2004. — 923с.
2. Макс Шлее. Qt4.5. Профессиональное программирование на C++. —СПб.: БХВ-Петербург, 2010. — 896 е.: ил. + DVD — (В подлиннике)
3. Бланшет Ж., Саммерфилд М. Qt 4: программирование GUI на C++. Пер. с англ. 2-е изд., доп. -М.: КУДИЦ-ПРЕСС, 2008. - 736 с.
4. Скляров В. А. Язык C++ и объектно-ориентированное программирование. — Минск.: Выш. шк., 1997. — 478 с.
5. Земсков Ю.В. Программирование на C++ с использованием библиотеки Qt 4.- Волгоград, 2007 г.
6. Секунов Н. Ю. Программирование на C++ в Linux. — СПб.: БХВ-Петербург, 2004. — 368 с.: ил.
7. Прата С. Язык программирования C++. Лекции и упражнения. Учебник: Пер. с англ./Стивсн Прата - СПб.: ООО «ДиаСофтЮП», 2005. - 1104 с.
8. Мэйерс С. Эффективное использование C++. 55 верных способов улучшить структуру и код ваших программ - М.: ДМК Пресс, 2006. - 300 с.: ил.
9. C/C++. Программирование на языке высокого уровня / Т. А. Павловская. — СПб.: Питер, 2003. — 461 е.: ил.
10. Шилдт Г. Самоучитель C++: Пер. с англ. — 3-е изд. — СПб.: БХВ-Петербург, 2003. — 688 с.
11. Шилдт, Герберт. Полный справочник по C++, 4-е издание. Пер. с англ. — М. : Издательский дом "Вильяме", 2006. — 800 с. : ил.
12. Архангельский А. Я. C++Builder 6. Справочное пособие. Книга 1. Язык C++. –М.: Бином-Пресс, 2002 г. — 544с.: ил.
13. Архангельский А.Я. C++Builder 6. Справочное пособие. Книга 2. Классы и компоненты. –М.: Бином-Пресс, 2002 г. – 528 с: ил.
14. Культин Н. Б. C++ Builder в задачах и примерах. — СПб.: БХВ-Петербург, 2005. – 336 с: ил.
15. Програмування та алгоритмічні мови. Методичні вказівки до виконання лабораторних робіт для студентів напрямку підготовки 0908 ”Електроніка” по спеціальності 6.090803 “Електронні системи”. Частина 2. Мова програмування “С”/ Укл. Ревко А.С., Гордієнко В.В. –Чернігів: ЧДТУ, 2007. – 60 с.

ДОДАТОК А – РОБОТА В СЕРЕДОВИЩІ QT CREATOR

Інтегроване середовище (Integrated Development Environment – IDE) – це комплекс програм, куди входять текстовий редактора, підсистема роботи з файлами, систему допомоги, вбудований відлагоджувач, підсистема керування компілюванням і редагуванням зв’язків, компілятор і редактор зв’язків, бібліотеки заготовочних файлів (з розширенням *.h) а також програми-утіліти. IDE дає можливість отримати файл, що може працювати на комп’ютері (EXE-файл), не використовуючи інші програми.

ДОДАТОК Б – ПРИКЛАД ТИТУЛЬНОГО ЛИСТА ЗВІТУ ПО ЦИКЛУ ЛАБОРАТОРНИХ РОБІТ

Міністерство освіти і науки України
Чернігівський національний технологічний університет

Кафедра промислової електроніки

ЗВІТ

про виконання циклу лабораторних робіт
з курсу ”**Інформатика**”

Виконав: студент групи (*шифр групи*)
(П.І.Б. студента)

Перевірив: (*П.І.Б. викладача*)

Чернігів (*рік*)