**Лабораторна робота №12**

**Спадкування властивостей класів**

**Мета роботи**

Ознайомитись з властивостями процесу спадкування та отримати практичні навички у створенні похідних від базового класів.

**Хід роботи**

1) Ознайомитися з методичними вказівками до лабораторної роботи та темою "Спадкування";

2) розробити схеми алгоритмів методів похідного класу згідно із завданням за варіантом;

3) за схемами алгоритмів виконати оголошення та визначення похідного від створеного у лабораторній роботі №11 класу і його членів;

4) розробити схему алгоритму програми, що демонструє роботу зі сформованим класом;

5) за схемою алгоритму написати програму на мові С++;

6) зробити висновки;

7) підготувати звіт про виконання лабораторної роботи, що включає наступні пункти: номер, тема, мета та хід лабораторної роботи, завдання, схема алгоритму програми, лістинг програми з коментуванням кожної інструкції, результат роботи програми (скріншот), висновки.

**Теоретичні відомості та рекомендації до виконання**

Спадкування – це одне з базисних понять об'єктно-орієнтованого програмування. Воно засновується на використанні вже розроблених класів та для створення більш повних чи спеціалізованих програмних об'єктів.

Основне призначення спадкування – спрощення написання програм, завдяки застосуванню вже перевірених функцій класу без їх повторного оголошення та визначення через створення копії об'єкта у новому класі.

Використання спадкування приводить до зменшення програмного коду. При виконанні завдання до лабораторної роботи студент повинен засвоїти базові принципи побудови похідних класів та навчитися працювати з їх членами із основної програми.

Виконання лабораторної роботи починається ознайомленням з теоретичними відомостями про основи побудови похідних класів та методичними вказівками. Засвоївши їх, студент приступає до розробки класу згідно із завданням за варіантом. Базовим класом для створюваного обирається клас з лабораторної роботи №11.

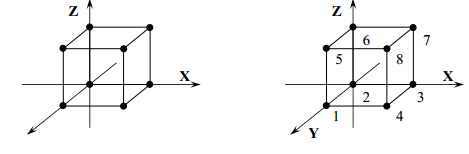
Після розробки класу студент переходить до створення схеми алгоритму програми, що демонструє можливості як базового, так і похідного класів. За схемою алгоритму виконується написання програми на мові С++.

У висновках до звіту студент повинен розкрити призначення теми "Спадкування властивостей класів". Висновок має бути поданий в формі пояснення сфери застосування вивченого матеріалу згідно із розумінням його студентом.

**Завдання для самостійного виконання**

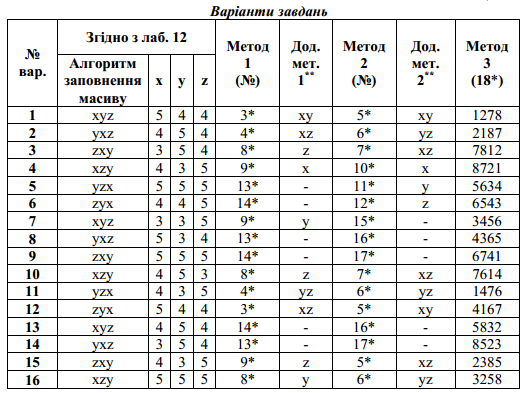
Розробити схему алгоритму та програму, що демонструє роботу з похідним від розробленого у лабораторній роботі №11 класом, оголошеним та визначеним згідно із завданням за варіантом. Клас повинен включати змінну, яка є прапорцем заповнення масиву, три методи, вказані у таблиці варіантів (табл. 11.1), метод поелементного заповнення тривимірного масиву та метод тасування елементів у двовимірному масиві.

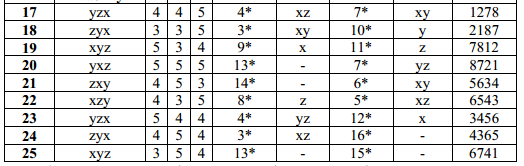
Тривимірний масив повинен бути побудований згідно з рис. 12.1.



**Рис. 12.1 Схематичне відображення тривимірного масиву**

Таблиця 12.1

****

****

\* в цих полях вказані номери методів, що наведені нижче:

1 – метод тасування елементів масиву;

2 – метод поелементного заповнення масиву;

3/4 – метод пошуку найбільшого/найменшого значення вектора;

5/6 – метод розміщення елементів вектора в порядку зростання/спадання;

7 – метод пошуку середнього арифметичного значення елементів вектора;

8/9 – метод пошуку найбільшого/найменшого значення матриці;

10/11 – метод розміщення елементів матриці у порядку зростання/спадання;

12 – метод пошуку середнього арифметичного значення елементів матриці;

13/14 – метод пошуку найбільшого/найменшого значення тривимірного масиву;

15/16 – метод розміщення елементів тривимірного масиву у порядку зростання/зменшення;

17 – метод пошуку середнього арифметичного значення елементів тривимірного масиву;

18 – метод виводу елементів зрізу тривимірного масиву у вигляді матриці.

\*\* у додаткових до методів полях вказано для векторів – координати матриці, у якій повинен знаходитися перший елемент вектора, для матриць – вісь, яка є перпендикулярною до неї.

**Вказівки до виконання завдання**

Завдання до лабораторної роботи включає розробку прапорця, трьох методів згідно з таблицею варіантів та базових методів, що мають бути присутні у всіх завданнях.

***Прапорець заповнення массиву*** – це змінна типу **bool**. Вона має значення 1, якщо масив було заповнено після запуску програми та 0, – якщо масив ще не заповнено. Ця змінна повинна опитуватись у всіх методах похідного класу для прийняття рішення про виконання операцій з масивом (якщо змінна має значення 0 – користувачу виводиться повідомлення про потрібність заповнення масиву. Програмний код функцій при цьому не виконується!).

Наприклад, якщо прапорець назвати flag, то для його перевірки у методах необхідно використати наступний код:

**if**(flag == 1) //*якщо прапорець має значення "1"*

{

<*діїзгідно з сутністю метода*>;

}

**else** //*якщо прапорець має значення "0"*

{

printf("Arrey did not fiilled!\r\n"); //*вивід константного рядка*

}

Базовими функціями, що будуть входити до похідного классу, є метод тасування елементів двовимірного масиву та метод поелементного заповнення масиву**:**

***1. Метод тасування елементів масиву.*** Для того, щоб можна було перевірити роботу методів похідного классу, значення елементів тривимірного масиву повинні бути записані випадковим чином. Щоб заповнити тривимірний масив випадковими значеннями, використовуйте одномірний масив, який заповнюється завдяки функції з лабораторної роботи №9 (повинна бути включена до методів похідного класу):

//*підключення бібліотек для реалізаціїфункцій заповнення масиву*

//*невпорядкованими значеннями вектора*

#**include**<algorithm>

#**include**<functional>

#**include**<vector>

**using namespace** std; //*застосування простору імен std*

//*оголошення масиву з 125 елементів (відповідає тривимірному масивові 5х5х5)*

**int** mas[125];

//*оголошення та визначення функції, що виконує заповнення масиву змішаними*

//*значеннями*

**void** RandomShuffle(**int** x, **int** n)

{

**const int** VECTOR\_SIZE = 125 ; //*число елементів вектора*

//*створення вектора*

**typedef** vector<**int**> IntVector ;

**typedef** IntVector::iterator IntVectorIt ;

//*оголошення та визначення змінноїперебору індексів елементів масиву*

**int** g=0;

IntVector Numbers(VECTOR\_SIZE) ;

IntVectorIt start, end, it ;

**for** (**int** i=0; i<125; i++) //*заповнення вектора значеннями*

{

Numbers[i] = x;

x=x+n;

}

//*визначення меж вектора*

start = Numbers.begin();

end = Numbers.end();

//*виклик функції, що перемішує значення вектора*

random\_shuffle(start, end);

//*заповнення масиву значеннями вектора*

**for**(it = start; it != end; it++)

{

mas[g]=\*it;

g++;

}

}

Після виконання цієї функції масив mas[125] буде заповнено числами від "x" до "x+124·n", де х – це початкове значення, а n – крок зміни значення.

Елементи цього масиву можна використати для послідовного заповнення тривимірного масиву, який було оголошено у базовому класі.

***2. Метод поелементного заповнення масиву.*** Ця функція повинна мати наступний формат оголошення:

**void** SetArrValue(**char** x, **char** y, **char** z, **int** n);

де х – номер елемента за віссю Х;

у – номер елемента за віссю Y;

z – номер елемента за віссю Z;

n – значення, яке буде присвоєно елементові масиву.

Завдяки цій функції є можливість змінювати значення елементів тривимірного масиву без використання імені масиву.

***3/4. Метод пошуку найбільшого/найменшого значення вектора.*** Ці функції повинні мати наступний формат оголошення:

**int** VectorMaxValue (**char** x, **char** y);

**int** VectorMinValue (**char** x, **char** y);

В цих функціях х та у – координати першого елемента вектора (згідно із таблицею може бути xy, xz, yz). Перший елемент завжди знаходиться на граничній, нульовій площині тривимірного масиву. Так, для вектора, що має початок у х = 3; у = 2, початковим елементом за z буде нульовий. Таким чином, координати першого елемента вектора будуть наступними (3, 2, 0).

Значення, які повертають ці функції, відповідно найбільший та найменший елементи вектора.

***5/6. Метод розміщення елементів вектора у порядку зростання/спадання.*** Функції повинні мати наступний формат оголошення:

**void** VectorMinToMax (**char** x, **char** y);

**void** VectorMaxToMin (**char** x, **char** y);

В цих функціях х та у – координати першого елемента вектора (див. метод 3/4).

Після виконання цього методу масив M[x], який треба буде додатково оголосити у похідному класі, повинен бути заповнений елементами з вказаного вектора згідно із завданням (у порядку зростання чи у порядку спадання).

***7. Метод пошуку середнього арифметичного значення елементів вектора.*** Функція повинна мати наступний формат оголошення:

**int** VectorMidValue (**char** x, **char** y);

В цій функції х та у – координати першого елемента вектора (див. метод 3/4). Функція повертає середнє арифметичне значення елементів вектора.

***8/9. Метод пошуку найбільшого/найменшого значення матриці.***

Функції повинні мати наступний формат оголошення:

**int** MatrixMaxValue (**char** x);

**int** MatrixMinValue (**char** x);

В цих функціях х – номер матриці двовимірного масиву. Наприклад, якщо сказано, що матриця має номер 3 по осі Х (згідно із варіантом може бути x, y, z), то вона паралельна площині YZ, а її елементи відповідають масивові Arr[2][y][z].

Значення, які повертають ці функції, відповідно найбільший та найменший елементи матриці.

***10/11. Метод розміщення елементів матриці у порядку зростання/спадання.*** Функції повинні мати наступний формат оголошення:

**void** MatrixMinToMax (**char** x);

**void** MatrixMaxToMin (**char** x);

В цих функціях х – номер матриці двовимірного масиву (див. метод 8/9).

Після виконання цього методу масив А[x][y], який треба буде додатково оголосити у похідному класі, повинен бути заповнений елементами з вказаного вектору згідно із завданням.

***12. Метод пошуку середнього арифметичного значення елементів матриці.*** Функція повинна мати наступний формат оголошення:

**int** MatrixMidValue (**char** x);

В цій функції х – номер матриці двовимірного масиву (див. метод 8/9).

Функція повертає середнє арифметичне значення елементів матриці.

***13/14. Метод пошуку найбільшого/найменшого значення тривимірного массиву.*** Функції повинні мати наступний формат оголошення:

**int** CubeMaxValue ();

**int** CubeMinValue ();

Значення, які повертають ці функції, відповідно найбільший та найменший елементи масиву.

***15/16. Метод розміщення елементів тривимірного масиву у порядку зростання/спадання.*** Функції повинні мати наступний формат оголошення:

**void** CubeMinToMax ();

**void** CubeMaxToMin ();

Після виконання цього методу масив B[x][y][z], який треба буде додатково оголосити у похідному класі, повинен бути заповнений значеннями елементів з тривимірного масиву Arr[x][y][z] згідно із завданням (новий масив заповнюється відповідно до алгоритму за завданням).

***17. Метод пошуку середнього арифметичного значення елементів тривимірного масиву.*** Функція повинна мати наступний формат оголошення:

**int** CubeMidValue ();

Функція повертає середнє арифметичне значення елементів масиву.

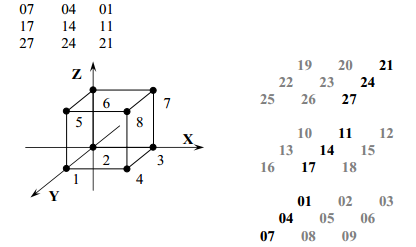
***18. Метод виводу елементів зрізу тривимірного масиву у вигляді матриці.*** Функція повинна мати наступний формат оголошення:

**void** ShowMatrix();

Функція вичитує значення елементів зрізу згідно з варіантом і розташовує їх у масиві С[x][у], який треба буде додатково оголосити у похідному класі.

Зчитування значень елементів проводиться наступним чином: записана у таблиці варіантів комбінація з чотирьох цифр є верхівками чотирикутного зрізу тривимірного массиву, причому перша цифра показує розташування початкового елемента для зчитування, а перші дві цифри у сукупності – рядок, по якому має проводитися зчитування.

Наприклад, комбінація "1278" для тривимірного масиву 3х3х3, наведеного на рис. 12.2, означає, що першим елементом матриці повинно бути значення "7", а другим по лінії за рисунком – "4". Вся матриця буде мати вигляд:



***Рис. 12.2. Відображення матриці, відповідноїдо зрізу "1278"***

Основна програма має відображати результати роботи всіх методів, причому спочатку на консоль повинен бути виведен заповнений випадковими значеннями масив за допомогою методу ShowCube() базового класу.

**Питання для підготовки до захисту лабораторної роботи**

1) Що таке спадкування?

2) У чому полягає призначення спадкування?

3) Які методи доступу до базового класу ви знаєте?

4) У чому полягає принцип створення похідних класів?

5) Яким чином виконується виклик методів базового класу через похідний?