Лабораторна робота №2. (2017)
Тема: *Основи Об'єктно – зорієнтоване програмування мовою С++.*
Класи. Протокол класу. Конструктори та деструктори.

Методичні вказівки

1. Основи об'єктно – зорієнтованого програмування мовою С++

Розвиток технологій програмування привело до виникнення об'єкто-зорієнтованого підходу до програмування (ОЗП). Об'єкто-зорієнтоване програмування (ООР, object-oriented programming) методологія програмування, заснована на представленні програми у вигляді сукупності об'єктів, кожен з який є реалізацією певного типу, що використовує механізм пересилання повідомлень і класи, організовані в ієрархію успадкування. Основні передумови виникнення ОЗП.

1. Зростання складності програм. Сучасні програми містять понад 10000 рядків коду. Налагодження та супровід таких програм стає дуже складним. Часто при зміні коду в одному місті програми виникають непередбачені помилки в інших місцях програми. ОЗП зводить програмування до побудови нових або використання готових класів. Кожен об'єкт є незалежним та проводить обмін даних з іншими об'єктами через відповідні інтерфейсні функції, таким чином, що зміна внутрішньої реалізації класу впливає тільки на роботу об'єктів даного класу;

2. ОЗП дозволяє зручно представляти в певному наближені об'єкти реального світу в термінах об'єктно-орієнтованої мови програмування (особливо використовуючи механізми наслідування та контейнеризації), що значно спрощує складання складних програм.

Центральний елемент ОЗП - абстракція. Дані за допомогою абстракції перетворюються в об'єкти, а послідовність обробки цих даних перетворюється на набір повідомлень, переданих між цими об'єктами. Кожен з об'єктів має своє власне унікальне поведінку. З об'єктами можна поводитися як із конкретними сутностями, які реагують на повідомлення, наказуючі їм виконати якісь дії.

ОЗП характеризується наступними принципами за Алану Кею(основоположником мови мови Smalltalk, якого вважають одним з «батьків-засновників» ООП, об'єкто-зорієнтованого підходу):

- все є об'єктом;
- обчислення здійснюються шляхом взаємодії (обміну даними) між об'єктами, при якому один об'єкт вимагає, щоб інший об'єкт виконав деяку дію; об'єкти взаємодіють, посилаючи й одержуючи повідомлення; повідомлення - це запит на виконання дії, доповнений набором аргументів, які можуть знадобитися при виконанні дії;
- кожен об'єкт має незалежну пам'ять, яка складається з інших об'єктів;
- кожен об'єкт є представником класу, який висловлює загальні властивості об'єктів даного типу;
- в класі задається функціональність (поведінка об'єкта); тим самим усі об'єкти, які є екземплярами одного класу, можуть виконувати одні і ті ж дії;
- класи організовані в єдину деревоподібну структуру з загальним корінням, звану ієрархією спадкування; пам'ять і поведінку, пов'язане з примірниками певного класу, автоматично доступні будь-якому класу, розташованому нижче в ієрархічному дереві.

Означення 1. Абстрагування (абстракція, abstraction) - метод розв'язання задачі, при якому об'єкти різного роду об'єднуються загальним поняттям (концепцією), а потім згруповано суті розглядаються як елементи єдиної категорії.

Абстрагування дозволяє відокремити логічний зміст фрагмента програми від проблеми його реалізації, розділивши зовнішнє опис (інтерфейс) об'єкта та його внутрішню організацію (реалізацію).

Означення 2. ***Інкапсуляція*** (encapsulation) - техніка, при якій не суттєва з точки зору інтерфейсу об'єкта інформація ховається всередині нього.

Означення 3. ***Успадкування*** (inheritance) - властивість об'єктів, за допомогою якого екземпляри класу отримують доступ до даних і методу класів-предків без їх повторного визначення.

Успадкування дозволяє різним типам даних спільно використовувати один і той самий код, приводячи до зменшення його розміру і підвищенню функціональності.

Означення 4. ***Поліморфізм*** (polymorphism) - властивість, що дозволяє використовувати один і той же інтерфейс для різних дій; поліморфної змінної, наприклад, може відповідати кілька різних методів.

Поліморфізм перекроює загальний код, який реалізує деякий інтерфейс, так, щоб задовольнити конкретним особливостей окремих типів даних.

Означення 5. ***Клас*** (class) - множина об'єктів, пов'язаних спільністю структури і поведінки; абстрактне опис даних і поведінки (методів) для сукупності схожих об'єктів, представники якої називаються екземплярами класу.

Означення 6. ***Об'єкт*** (object) - конкретна реалізація класу, що володіє характеристиками стану, поведінки та індивідуальності, синонім екземпляра.

Значно більш широкий список основних термінів ОЗП буде наведено пізніше. Слід мати на увазі, що в різних об'єктно-орієнтованих мовах для позначення одних і тих же концепцій ОЗП використовуються злегка відрізняються один від одного терміни.

Об'єктно-орієнтоване програмування сягає своїм корінням до створення мови програмування Симула в 1960-тих роках. На сьогодні багато із мов програмування (зокрема, C++, Java, C#, ActionScript , Python, PHP, Ruby та Objective-C) підтримують ОЗП. Мова С++ підтримує класичну форму ОЗП, вона активно використовується професійними програмістами.

2. Класи. Протокол класу. Конструктори та деструктори.

Клас – це створення нового типу (клас це – тип) розширення поняття структури мови С. Клас – тип даних, за допомогою якого крім набору типу даних можна задавати набір функцій для роботи з ними. Дані класу називаються *елементами даних* або *полями* (за аналогією з полями структури), а функції класу *методами*. Поля та методи називаються *елементами* класу. Опис класу:

class *ім'я_класу <: предки >****{***

<опис елементів класу> .. // Протокол визначення класу

} *<об'єкти класів > ;* *// Опис закінчується крапкою з коми*

Опис елементів класу – це визначення даних та опис(декларація) або визначення функцій. Опис елементів містить один або кілька *специфікацій доступу* до елементів, що задають за допомогою ключових слів *public, private* або *protected,* вони керують видимістю елементів класу. Елементи, описані після службового слова *private*, доступні тільки усередині класу. Цей вид доступу прийнятий за замовчуванням. Елементи, описані після службового слова *public*, доступні для функцій-елементів й інших функцій програми, у якій є представник класу. Елементи, описані після службового слова *protected,* доступні тільки усередині класу та при успадкуванні.

Елементи даних класів можуть мати будь-якого типу, крім типу цього ж класу (але можуть бути вказівник або посилання на цей клас). Ініціалізація полів при описі класу не допускається. Класи можуть бути глобальними (оголошеними поза будь-яким блоком) і локальними (оголошеними усередині блоку, наприклад, функції або іншого класу).

У класі крім звичайний методів(функцій) класу є деякі спеціальні функції, такі, як конструктори, деструктори, функції перетворення, операції.

Конструктор - функція з тим же ім'ям, що й сам клас без типу. В класі може бути декілька конструкторів. Деструктор - функція, ім'ям якої є ім'я класу із префікс-тильдою (~) без типів і без параметрів.

Кожен представник класу називається *об'єктом*.

Завдання до лабораторної роботи:

Розробити та реалізувати класи згідно варіантів. Передбачити введення початкових даних: з клавіатури, файлу та використовуючи датчик випадкових чисел. Написати тестові програми для кожної задачі.

Варіант	Завдання 1	Завдання 2	Завдання 3
1	Задача 1.1	Задача 2.1	Задача 3.1
2	Задача 1.2	Задача 2.2	Задача 3.2
3	Задача 1.3	Задача 2.3	Задача 3.3
4	Задача 1.4	Задача 2.4	Задача 3.4
5	Задача 1.5	Задача 2.5	Задача 3.5
6	Задача 1.6	Задача 2.6	Задача 3.6
7	Задача 1.7	Задача 2.7	Задача 3.7
8	Задача 1.8	Задача 2.8	Задача 3.8
9	Задача 1.9	Задача 2.9	Задача 3.9
10	Задача 1.10	Задача 2.10	Задача 3.10
11	Задача 1.1	Задача 2.6	Задача 3.8
12	Задача 1.2	Задача 2.7	Задача 3.9
13	Задача 1.3	Задача 2.8	Задача 3.10
14	Задача 1.4	Задача 2.9	Задача 3.1
15	Задача 1.5	Задача 2.10	Задача 3.2
16	Задача 1.6	Задача 2.1	Задача 3.3
17	Задача 1.7	Задача 2.2	Задача 3.4
18	Задача 1.8	Задача 2.3	Задача 3.5
19	Задача 1.9	Задача 2.4	Задача 3.6
20	Задача 1.10	Задача 2.5	Задача 3.7
21	Задача 1.1	Задача 2.6	Задача 3.8
22	Задача 1.2	Задача 2.7	Задача 3.9
23	Задача 1.3	Задача 2.8	Задача 3.10
24	Задача 1.4	Задача 2.9	Задача 3.1
25	Задача 1.5	Задача 2.10	Задача 3.2
26	Задача 1.6	Задача 2.1	Задача 3.3
27	Задача 1.7	Задача 2.2	Задача 3.4
28	Задача 1.8	Задача 2.3	Задача 3.5
29	Задача 1.9	Задача 2.4	Задача 3.6
30	Задача 1.10	Задача 2.5	Задача 3.7

Група задач 1.

Задача 1.1.

Створити тип даних - клас вектор, який має вказівник на **int**, число елементів і змінну стану. У класі визначити

- конструктор без параметрів(виділяє місце для одного елемента та інінціалізує його в нуль);
- конструктор з одним параметром - розмір вектора(виділяє місце та інінціалізує масив значенням нуль);
- конструктор із двома параметрами - розмір вектора та значення ініціалізації(виділяє місце (значення перший аргумент) та інінціалізує значенням другого аргументу).
- деструктор звільняє пам'ять.
- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент масиву;
- визначити функції друку, додавання, віднімання, які здійснюють ці арифметичні операції з даними цього класу, множення на ціле типу **short**;
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі масиву. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 1.2.

Створити тип даних - клас вектор, який має вказівник на **float**, число елементів і змінну стану. У класі визначити

- конструктор без параметрів(виділяє місце для одного елемента та інінціалізує його в нуль);
- конструктор з одним параметром - розмір вектора(виділяє місце та інінціалізує масив значенням нуль);
- конструктор із двома параметрами - розмір вектора та значення ініціалізації(виділяє місце (значення перший аргумент) та інінціалізує значенням другого аргументу).
- деструктор звільняє пам'ять.
- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент масиву;
- визначити функції друку, додавання, віднімання, які здійснюють ці арифметичні операції з даними цього класу, множення на ціле типу **long**;
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі масиву. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 1.3.

Створити тип даних - клас вектор, який має вказівник на **double**, число елементів і змінну стану. У класі визначити

- конструктор без параметрів(виділяє місце для одного елемента та інінціалізує його в нуль);
- конструктор з одним параметром - розмір вектора(виділяє місце та інінціалізує масив значенням нуль);
- конструктор із двома параметрами - розмір вектора та значення ініціалізації(виділяє місце (значення перший аргумент) та інінціалізує значенням другого аргументу).
- деструктор звільняє пам'ять.

- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент масиву;
- визначити функції друку, додавання, віднімання, які здійснюють ці арифметичні операції з даними цього класу, множення та ділення на скаляр типу **double**;
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі масиву. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 1.4.

Створити тип даних - клас вектор, який має вказівник на **short**, число елементів і змінну стану. У класі визначити

- конструктор без параметрів(виділяє місце для одного елемента та інінціалізує його в нуль);
- конструктор з одним параметром - розмір вектора(виділяє місце та інінціалізує масив значенням нуль);
- конструктор із двома параметрами - розмір вектора та значення ініціалізації(виділяє місце (значення перший аргумент) та інінціалізує значенням другого аргументу).
- деструктор звільняє пам'ять.
- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент масиву;
- визначити функції друку, додавання, віднімання, які здійснюють ці арифметичні операції з даними цього класу, множення на ціле типу **unsigned char**;
- визначити функції порівняння: більше, нерівно або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі масиву. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 1.5.

Створити тип даних - клас вектор, який має вказівник на **long**, число елементів і змінну стану. У класі визначити

- конструктор без параметрів(виділяє місце для одного елемента та інінціалізує його в нуль);
- конструктор з одним параметром - розмір вектора(виділяє місце та інінціалізує масив значенням нуль);
- конструктор із двома параметрами - розмір вектора та значення ініціалізації(виділяє місце (значення перший аргумент) та інінціалізує значенням другого аргументу);
- деструктор звільняє пам'ять;
- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент масиву;
- визначити функції друку, додавання, віднімання, які здійснюють ці арифметичні операції з даними цього класу, множення на ціле типу **unsigned int**;

- визначити функції порівняння: менше, нерівно або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі масиву. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 1.6.

Створити тип даних - клас вектор, який має поля x, y та z типу **float** і змінну стану. У класі визначити

- конструктор без параметрів(інінціалізує поля в нуль);
- конструктор з одним параметром типу **float** (інінціалізує поля x, y та z значенням параметру);
- конструктор з одним параметром вказівник на тип (інінціалізує поля x, y та z значенням масиву за вказівником, якщо вказівник NULL (nulptr) то встановити код помилки);
- деструктор із виведенням інформації про стан вектора;
- визначити функцію, яка присвоює полю x, y або z деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент з полів x, y та z;
- визначити функції друку, додавання, віднімання, векторний добуток які здійснюють ці арифметичні операції з даними цього класу;
- функцію ділення на ціле типу **short(**при діленні на 0 змінити стан, а ділення не виконувати**);**
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, діленні на 0, при передачі NULL (nulptr) в конструкторі із вказівником. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 1.7.

Створити тип даних - клас вектор, який має масив з трьох елементів типу **double** і змінну стану. У класі визначити

- конструктор без параметрів(інінціалізує поля в нуль);
- конструктор з одним параметром типу **double** (інінціалізує масив значенням параметру);
- конструктор з одним параметром вказівник на тип (інінціалізує масив класу значенням елементів масиву, який передається параметр за вказівником, якщо вказівник NULL (nulptr) то встановити код помилки);
- деструктор із виведенням інформації про стан вектора;
- визначити функцію, яка присвоює масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент з масиву;
- визначити функції друку, додавання, віднімання та векторного добутку які здійснюють ці арифметичні операції з даними цього класу;
- функцію ділення на ціле типу **short(**при діленні на 0 змінити стан, а ділення не виконувати**);**
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, діленні на 0, при передачі NULL (nulptr) в конструкторі із вказівником. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 1.8.

Створити тип даних - клас комплексне число, який має поля re та im типу **float** і поле змінну стану. У класі визначити

- конструктор без параметрів(інінціалізує поля в нуль);
- конструктор з одним параметром типу **float** (інінціалізує поля re та im значенням параметру);
- конструктор з одним параметром вказівник на тип (інінціалізує поля re та im значенням масиву за вказівником, якщо вказівник NULL (nulptr) то встановити код помилки);
- деструктор із виведенням інформації про стан вектора;
- визначити функцію, яка присвоює полю re та im деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент з полів re та im;
- визначити функції друку, додавання, віднімання, множення та ділення які здійснюють ці арифметичні операції з даними цього класу;
- функцію ділення на ціле типу **short(**при діленні на 0 змінити стан, а ділення не виконувати**)**;
- норми комплексного числа;
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, діленні на 0, при передачі NULL (nulptr) в конструкторі із вказівником. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 1.9.

Створити тип даних - клас вектор, який має вказівник на **unsigned int**, число елементів і змінну стану. У класі визначити

- конструктор без параметрів(виділяє місце для одного елемента та інінціалізує його в нуль);
- конструктор з одним параметром - розмір вектора(виділяє місце та інінціалізує масив значенням нуль);
- конструктор із двома параметрами - розмір вектора та значення ініціалізації(виділяє місце (значення перший аргумент) та інінціалізує значенням другого аргументу).
- деструктор звільняє пам'ять.
- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент масиву;
- визначити функції друку, додавання, віднімання, які здійснюють ці арифметичні операції з даними цього класу, множення на ціле типу **unsigned short**;
- визначити функції порівняння: більше, менше, нерівно або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі масиву, ініціалізація від'ємним числом. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 1.10.

Створити тип даних - клас множина з повторенням діапазону цілих чисел, який має вказівник на **unsigned int**, числа **beg** (початок) та **end**(кінець) діапазону типу **unsigned int** (**beg<end**) і змінну стану. У множині створюється масив розміром **size = end - beg ,** який кількість повторень елемента множини. У класі визначити

- конструктор без параметрів(виділяє місце для множини чисел **beg** = 0 до **end** =100 та інінціалізує його в нуль);

- конструктор з одним параметром - **end** (**beg** = 0, виділяє місце та інінціалізує масив значенням нуль);
- конструктор із двома параметрами **beg** та **end** (виділяє місце та інінціалізує масив значенням нуль);
- конструктор із трьома параметрами **beg, end** та **value**(виділяє місце та інінціалізує масив значенням **value**);
- деструктор звільняє пам'ять.
- визначити функцію, яка присвоює елементу множини деяке значення **value** (параметр за замовчуванням);
- функцію яка одержує кількість повторення елементу множини;
- визначити функції друку, об'єднання, перетину та різниці, які здійснюють ці арифметичні операції з даними цього класу;
- визначити функції порівняння: більше, менше, нерівно або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі множини, ініціалізація числом, яке є за межами діапазону множини. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Група задач 2

Задача 2.1.

Створити клас матриця. Даний клас містить вказівник на **int**, розміри рядків і стовпців та стан помилки. У класі визначити

- конструктор без параметрів(виділяє місце для матриці 3 на 3 елемента та інінціалізує його в нуль);
- конструктор з одним параметром – розмір **n** матриці (виділяє місце **n** на **n** та інінціалізує матрицю значенням нуль);
- конструктор із трьома розміри матриці (**n** , **m**) та значення ініціалізації **value** (виділяє місце перші аргументи та інінціалізує значенням третього аргументу - **value**).
- деструктор звільняє пам'ять.
- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент матриці за індексами i та j;
- визначити функції друку, додавання, множення, віднімання, які здійснюють ці арифметичні операції з даними цього класу;
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі матриці. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 2.2.

Створити клас матриця. Даний клас містить вказівник на **float**, розміри рядків і стовпців та стан помилки. У класі визначити

- конструктор без параметрів(виділяє місце для матриці 2 на 2 елемента та інінціалізує його в нуль);
- конструктор з одним параметром - розмір **n** матриці (виділяє місце для матриці **n** на **n** та інінціалізує матрицю значенням нуль);
- конструктор із трьома розміри матриці (**n** , **m**) та значення ініціалізації **value** (виділяє місце перші аргументи та інінціалізує значенням третього аргументу - **value**).
- деструктор звільняє пам'ять.

- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент матриці за індексами і та j;
- визначити функції друку, додавання, множення на скаляр, віднімання, які здійснюють ці арифметичні операції з даними цього класу;
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі матриці. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 2.3.

Створити клас матриця. Даний клас містить вказівник на **double**, розміри рядків і стовпців та стан помилки. У класі визначити

- конструктор без параметрів(виділяє місце для матриці 4 на 3 елемента та інінціалізує його в нуль);
- конструктор з одним параметром - розмір **n** матриці (виділяє місце для матриці **n** на **n** та інінціалізує матрицю значенням нуль);
- конструктор із трьома розміри матриці (**n** , **m**) та значення ініціалізації **value** (виділяє місце перші аргументи та інінціалізує значенням третього аргументу - **value**).
- деструктор звільняє пам'ять.
- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент матриці за індексами і та j;
- визначити функції друку, додавання, множення на скаляр типу **double**, віднімання, які здійснюють ці арифметичні операції з даними цього класу;
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі матриці. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 2.4.

Створити клас матриця. Даний клас містить вказівник на **short**, розмір рядків і стовпців та стан помилки. У класі визначити

- конструктор без параметрів(виділяє місце для матриці 4 на 4 елемента та інінціалізує його в нуль);
- конструктор з одним параметром – розмір **n** матриці (виділяє місце **n** на **n** та інінціалізує матрицю значенням нуль);
- конструктор із трьома параметрами - розміри матриці (**n** , **m**) та значення ініціалізації **value** (виділяє місце перші аргументи та інінціалізує значенням третього аргументу - **value**).
- деструктор звільняє пам'ять.
- визначити функцію, яка присвоює елементу масиву деяке значення **value** (параметр за замовчуванням);
- функцію яка одержує деякий елемент матриці за індексами і та j;
- визначити функції друку, додавання, множення, віднімання, які здійснюють ці арифметичні операції з даними цього класу;
- визначити функцію множення матриці на скаляр типу **short;**
- визначити функції порівняння: більше, менше або нерівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі матриці. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 2.5.

Створити клас матриця. Даний клас містить *вказівник на вказівник на* **int**, розміри рядків і стовпців та стан помилки. У класі визначити

- конструктор без параметрів(виділяє місце для матриці 3 на 3 елемента та інінціалізує його в нуль);
- конструктор з одним параметром – розмір **n** матриці (виділяє місце **n** на **n** та інінціалізує матрицю значенням нуль);
- конструктор із трьома параметрами розміри матриці (**n , m**) та значення ініціалізації **value** (виділяє місце перші аргументи та інінціалізує значенням третього аргументу - **value**).
- деструктор звільняє пам'ять.
- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент матриці за індексами i та j;
- визначити функції друку, додавання, множення, віднімання, які здійснюють ці арифметичні операції з даними цього класу;
- визначити функцію множення матриці на скаляр типу **short;**
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі матриці. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 2.6.

Створити клас матриця. Даний клас містить *вказівник на вказівник на* **double**, розміри рядків і стовпців та стан помилки. У класі визначити

- конструктор без параметрів(виділяє місце для матриці 2 на 2 елемента та інінціалізує його в нуль);
- конструктор з одним параметром – розмір **n** матриці (виділяє місце **n** на **n** та інінціалізує матрицю значенням нуль);
- конструктор із трьома параметрами розміри матриці (**n , m**) та значення ініціалізації **value** (виділяє місце перші аргументи та інінціалізує значенням третього аргументу - **value**).
- деструктор звільняє пам'ять.
- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент матриці за індексами i та j;
- визначити функції друку, додавання, множення, віднімання, які здійснюють ці арифметичні операції з даними цього класу;
- визначити функцію ділення матриці на скаляр типу **int (**у випадку якщо скаляр рівний нулю ділення не виконувати**);**
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі матриці, ділення на 0. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 2.7.

Створити клас матриця. Даний клас містить *вказівник на вказівник на* **long**, розміри рядків і стовпців та стан помилки. У класі визначити

- конструктор без параметрів(виділяє місце для матриці 5 на 5 елемента та інінціалізує його в нуль);
- конструктор з одним параметром – розмір **n** матриці (виділяє місце **n** на **n** та інінціалізує матрицю значенням нуль);
- конструктор із трьома параметрами розміри матриці (**n , m**) та значення ініціалізації **value** (виділяє місце перші аргументи та інінціалізує значенням третього аргументу - **value**).
- деструктор звільняє пам'ять.
- визначити функцію, яка присвоює елементу масиву деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент матриці за індексами i та j;
- визначити функції друку, додавання, множення, віднімання, які здійснюють ці арифметичні операції з даними цього класу;
- визначити функцію ділення матриці на скаляр типу **long (**у випадку якщо скаляр рівний нулю ділення не виконувати**);**
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі матриці, ділення на 0. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 2.8.

Створити два класи вектор та матриця. Даний клас вектор містить *вказівник на* **long** та розмір вектора, клас матриця містить *вказівник на* **створений клас вектор,** кількість векторів та стан помилки. У класах визначити

- конструктори без параметрів(для вектора виділяє місце для 5 елементів, для матриці виділяє місце для 5 векторів та інінціалізує елементи в нуль);
- конструктори з одним параметром – розмір **n** матриці (для вектора виділяє місце для **n** елементів, для матриці виділяє місце для **n** векторів та інінціалізує елементи в нуль);
- конструктор із трьома параметрами розміри матриці (**n , m**) та значення ініціалізації **value** (для вектора виділяє місце для **n** елементів, для матриці виділяє місце для **m** векторів та інінціалізує елементи значенням третього аргументу - **value**).
- деструктори звільняють пам'ять.

У класі матриці визначити

- визначити функцію, яка присвоює елементу матриці деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент матриці за індексами i та j;
- визначити функції друку, додавання, множення, віднімання, які здійснюють ці арифметичні операції з даними цього класу;
- визначити функцію ділення матриці на скаляр типу **long (**у випадку якщо скаляр рівний нулю ділення не виконувати**);**
- визначити функції порівняння: більше, менше або рівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі масивів, ділення на 0. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 2.9.

Створити два класи вектор та матриця. Даний клас вектор містить *вказівник на* **double** та розмір вектора, клас матриця містить *вказівник на* **створений клас вектор,** кількість векторів та стан помилки. У класах визначити

- конструктори без параметрів(для вектора виділяє місце для 5 елементів, для матриці виділяє місце для 5 векторів та інінціалізує елементи в нуль);
- конструктори з одним параметром – розмір **n** матриці (для вектора виділяє місце для **n** елементів, для матриці виділяє місце для **n** векторів та інінціалізує елементи в нуль);
- конструктор із трьома параметрами розміри матриці (**n** , **m**) та значення ініціалізації **value** (для вектора виділяє місце для **n** елементів, для матриці виділяє місце для **m** векторів та інінціалізує елементи значенням третього аргументу - **value**).
- деструктори звільняють пам'ять.

У класі матриці визначити

- визначити функцію, яка присвоює елементу матриці деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент матриці за індексами i та j;
- визначити функції друку, додавання, множення, віднімання, які здійснюють ці арифметичні операції з даними цього класу;
- визначити функцію ділення матриці на скаляр типу **int (**у випадку якщо скаляр рівний нулю ділення не виконувати**);**
- визначити функції порівняння: більше, менше або нерівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі масивів, ділення на 0. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Задача 2.10.

Створити два класи вектор та матриця. Даний клас вектор містить *вказівник на* **char** та розмір вектора, клас матриця містить *вказівник на* **створений клас вектор,** кількість векторів та стан помилки. У класах визначити

- конструктори без параметрів(для вектора виділяє місце для 5 елементів, для матриці виділяє місце для 5 векторів та інінціалізує елементи в '_' - символ пробіл);
- конструктори з одним параметром – розмір **n** матриці (для вектора виділяє місце для **n** елементів, для матриці виділяє місце для **n** векторів та інінціалізує елементи в '_' - символ пробіл);
- конструктор із трьома параметрами розміри матриці - **n** і **m** типу **unsigned int** та значення ініціалізації **value** типу **char**(для вектора виділяє місце для **n** елементів, для матриці виділяє місце для **m** векторів та інінціалізує елементи значенням третього аргументу - **value**).
- деструктори звільняють пам'ять.

У класі матриці визначити

- визначити функцію, яка присвоює елементу матриці деяке значення (параметр за замовчуванням);
- функцію яка одержує деякий елемент матриці за індексами i та j;
- визначити функції друку, кодування та декодування, які здійснюють ці по-бітові операції з даними цього класу;
- визначити функцію кодування матриці за скалярною маскою типу **unsigned char (**у випадку якщо скаляр рівний нулю кодування не виконувати**);**
- визначити функції порівняння: більше, менше або нерівно, які повертають ***true*** або ***false***.

У змінну стани встановлювати код помилки, коли не вистачає пам'яті, виходить за межі масивів, при кодуванні з маскою 0. Передбачити можливість підрахунку числа об'єктів даного типу. Перевірити роботу цього класу.

Група задач 3

Задача 3.1.

Створити клас типу - дата з полями: день (1-31), місяць (1-12), рік (ціле число). У класі визначити У класі визначити

- конструктори (не менше двох);
- функції-члени встановлення дня, місяця та року, функції встановлення полів класу повинні перевіряти коректність параметрів, що задаються;
- функції-члени одержання дня, місяця та року;
- дві функції-члени друку за шаблоном: "5 січня 2017 року" і "05.01.2017".

Перевірити роботу цього класу.

Задача 3.2.

Створити клас типу - - час із полями: година (0-23), хвилини (0-59), секунди (0-59). У класі визначити

- конструктори (не менше двох);
- функції-члени встановлення години, хвилини та секунди, функції встановлення полів класу повинні перевіряти коректність параметрів, що задаються;
- функції-члени одержання години, хвилини та секунди;
- дві функції-члени друку за шаблоном: "16 годин 18 хвилин 3 секунди " і "4 p.m. 18 хвилин 3 секунди "

Перевірити роботу цього класу.

Задача 3.3.

Створити клас типу – прямокутник (поля : висота, ширина, колір). У класі визначити

- конструктори (не менше двох);
- функції-члени обчислення площі, периметру;
- функції-члени встановлення висоти, ширини, кольору, функції встановлення полів класу повинні перевіряти коректність параметрів, що задаються;
- функції-члени що повертають значення полів;
- функцію друку.

Перевірити роботу цього класу.

Задача 3.4.

Створити клас типу – паралелограм (поля : основа, висота, бічна сторона, колір). У класі визначити

- конструктори (не менше двох);
- функції-члени обчислення площі, периметру;
- функції-члени встановлення основи, висоти, бічної сторони, кольору, функції встановлення полів класу повинні перевіряти коректність параметрів, що задаються;
- функції-члени що повертають значення полів;
- функцію друку.

Перевірити роботу цього класу.

Задача 3.5.

Створити клас типу – трикутник (поля : сторони, колір). У класі визначити

- конструктори (не менше двох);
- функції-члени обчислення площі, периметру;
- функції-члени встановлення значень сторін та кольору, функції встановлення полів класу повинні перевіряти коректність параметрів, що задаються;

- функції-члени що повертають значення полів;
- функцію друку.

Перевірити роботу цього класу.

Задача 3.6.

Створити клас типу – ромб (поля : сторона, діагональ, колір). У класі визначити

- конструктори (не менше двох);
- функції-члени обчислення площі, периметру;
- функції-члени встановлення значення сторони, діагоналі та кольору, функції встановлення полів класу повинні перевіряти коректність параметрів, що задаються;
- функції-члени що повертають значення полів;
- функцію друку.

Перевірити роботу цього класу.

Задача 3.7.

Створити клас типу гра в хрестики-нулики (поля: класу - масив з (3x3), стан, активний гравець). У класі визначити

- конструктор;
- функції-члени перевірки кінця гри, виведення повідомлень по стан гри, початку гри, ходів граків, тощо;

Перевірити роботу цього класу.

Задача 3.8.

Створити клас типу – круг (поля : радіус, колір). У класі визначити

- конструктори (не менше двох);
- функції-члени обчислення площі круга, довжини кола;
- функції-члени встановлення значення радіусу та кольору, функції встановлення полів класу повинні перевіряти коректність параметрів, що задаються;
- функції-члени що повертають значення полів;
- функцію друку.

Перевірити роботу цього класу.

Задача 3.9.

Створити клас типу – квадрат (поля : сторона, колір). У класі визначити

- конструктори (не менше двох);
- функції-члени обчислення площі, периметру, описаного та вписаного кола;
- функції-члени встановлення значення сторони та кольору, функції встановлення полів класу повинні перевіряти коректність параметрів, що задаються;
- функції-члени що повертають значення полів;
- функцію друку.

Перевірити роботу цього класу.

Задача 3.10.

Створити клас типу – куб (поля : сторона, колір). У класі визначити

- конструктори (не менше двох);
- функції-члени обчислення площі поверхні, об'єму, довжини діагоналі та об'єм вписаної сфери;
- функції-члени встановлення значення сторони та кольору, функції встановлення полів класу повинні перевіряти коректність параметрів, що задаються;
- функції-члени що повертають значення полів;

- функцію друку.

Перевірити роботу цього класу.