**2 лабораторная**

***Тема: Создание базовых объектов.***

**Общие требования к выполнению работы**

1. Создать проект консольной программы.
2. Создать базовый класс согласно заданию. Все данные класса описать в закрытой секции, все методы класса реализовать как открытые. Создать конструктор по умолчанию и конструктор инициализации. Создать деструктор.
3. Вне класса должна быть создана дружественная функция, предназначенная для вывода данных объекта на экран.
4. Для задания значений уникальных номеров использовать или статические члены класса или особую инициализацию членов класса.
5. Установка значений переменных должна иметь проверку на реальный диапазон значений.
6. В головной функции создать несколько одиночных экземпляров базового класса. Обработать все экземпляры базового класса с помощью заданного метода дружественного класса. Вывести содержимое переменных экземпляров классов на экран до и после этой обработки.
7. В головной функции создать массив экземпляров базового класса. Обработать все элементы массива с помощью заданного метода дружественного класса. Вывести содержимое переменных элементов массива на экран до и после этой обработки.

**Варианты**

1. Создать класс «Студент» со следующими данными: id – уникальный номер, name – ФИО, rate – рейтинг. Создать дружественный класс «Декан», в котором реализовать метод установки рейтинга студента.
2. Создать класс «Деталь» со следующими данными: id – уникальный номер, name – наименование, mass – вес. Создать дружественный класс «Контроль», в котором реализовать метод установки веса детали.
3. Создать класс «Сотрудник» со следующими данными: id – уникальный номер, name – ФИО, salary – оклад. Создать дружественный класс «Бухгалтер», в котором реализовать метод установки оклада сотрудника.
4. Создать класс «Автомобиль» со следующими данными: id – уникальный номер, name – ФИО владельца, mileage – пробег. Создать дружественный класс «СТО», в котором реализовать метод установки пробега автомобиля.
5. Создать класс «Товар» со следующими данными: id – уникальный номер, name – название, cost – стоимость. Создать дружественный класс «Продавец», в котором реализовать метод установки стоимости товара.
6. Создать класс «Задание» со следующими данными: id – уникальный номер, name – название, result – процент выполнения. Создать дружественный класс «Проверяющий», в котором реализовать метод установки процента выполнения задания.
7. Создать класс «Школьник» со следующими данными: id – уникальный номер, name – ФИО, missed – количество пропущенных занятий. Создать дружественный класс «Учитель», в котором реализовать метод установки количества пропущенных занятий.
8. Создать класс «Фрукт» со следующими данными: id – уникальный номер, name – наименование, price – цена. Создать дружественный класс «Менеджер», в котором реализовать метод установки цены фрукта.
9. Создать класс «Котик» со следующими данными: id – уникальный номер, name – ФИО, diagnosis – диагноз. Создать дружественный класс «Ветеринар», в котором реализовать метод установки диагноза котика.
10. Создать класс «Мотоцикл» со следующими данными: id – уникальный номер, name – ФИО владельца, mileage – пробег. Создать дружественный класс «Владелец», в котором реализовать метод установки пробега мотоцикла.
11. Создать класс «Одежда» со следующими данными: id – уникальный номер, name – название, cost – стоимость. Создать дружественный класс «Товаровед», в котором реализовать метод установки стоимости одежду.
12. Создать класс «Экзамен» со следующими данными: id – уникальный номер, name – название, mark – оценка. Создать дружественный класс «Экзаменатор», в котором реализовать метод установки оценки за экзамен.
13. Создать класс «Death Stranding» со следующими данными: id – уникальный номер, time – время, проведенное за игрой, level – уровень сложности. Создать дружественный класс «Геймер», в котором реализовать метод установки времени, проведенного за игрой.
14. Создать класс «Сериал» со следующими данными: id – уникальный номер, name – название, review – отзыв. Создать дружественный класс «Зритель», в котором реализовать метод добавления отзыва на сериал.
15. Создать класс «Олимпиада» со следующими данными: id – уникальный номер, name – ФИО участника, place – место. Создать дружественный класс «Судья», в котором реализовать метод установки места, которое участник занял в олимпиаде.

**3 лабораторная**

***Тема: Перегрузка функций и операторов.******Динамическое создание объектов и массивов объектов.***

**Общие требования к выполнению работы**

1. Создать проект консольной программы.
2. Создать базовый класс согласно заданию. Все данные класса описать в закрытой секции, все методы класса реализовать как открытые.
3. Создать конструктор по умолчанию, конструктор инициализации и конструктор копирования. Конструкторы должны выделять память для данных динамически с использованием оператора new.
4. Создать деструктор, который будет освобождать динамически выделенную память с помощью оператора delete.
5. Создать в классе методы доступа к одиночному элементу данных объекта (по чтению и записи), отображения данных на экран, преобразования данных объекта (например, сортировки).
6. Перегрузить операторы-члены класса (согласно варианту):

-- присвоение =;

-- индексация элемента [];

-- инициализация ();

-- унарные арифметические (++);

-- бинарные арифметические (+ или ‑ ) и сравнения (> или < или ==), причем только для однотипных операндов;

1. Перегрузить дружественные функции-операторы (согласно варианту)::

-- унарные арифметические (--);

-- бинарные арифметические (+ или ‑ ) и сравнения (> или < или ==), причем для разных типов операндов;

-- вывода в поток (<<).

1. В головной функции создать несколько экземпляров базового класса. Провести вызовы перегруженных функций и операторов. Вывести результаты на экран.
2. В головной функции динамически создать и проинициализировать экземпляр базового класса. Динамически создать его копию. Выполнить преобразование данных экземпляра базового класса. Вывести содержимое обоих экземпляров на экран. Удалить экземпляры.
3. В головной функции динамически создать массив экземпляров класса. Каждый из них проинициализировать. Выполнить доступ к элементам данных экземпляров классов. Удалить массивы.

**Варианты:**

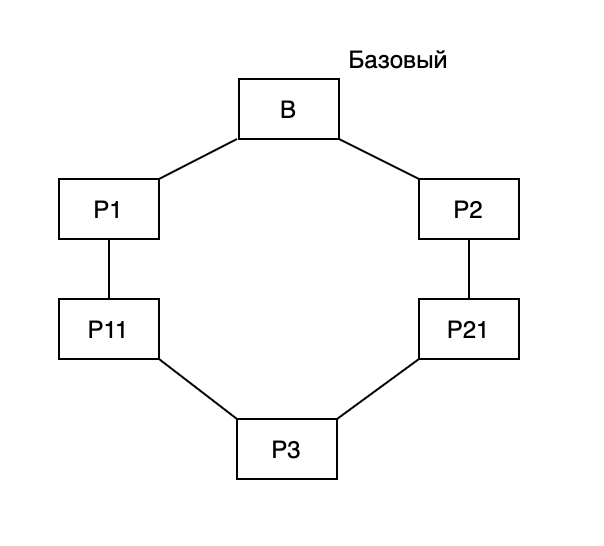
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Класс** | **Операторы-члены класса** | **Дружественные функции-операторы** | **Сортировка** |
| **1** | Двумерный массив (матрица) с элементами типа char. | присвоение =;  индексация элемента []; | унарные арифметические (--); |  |
| **2** | Двумерный массив (матрица) с элементами типа int. | инициализация ();  перемножение матриц & | бинарные арифметические (+ и ‑ ) , причем для разных типов операндов; |  |
| **3** | Двумерный массив (матрица) с элементами типа double | бинарные арифметические (+ и ‑ ) , причем только для однотипных операндов; | вывода в поток (<<) |  |
| **4** | Строка | сравнения (> и <),  сложение строки и объекта += | унарные арифметические (--); | Пузырьком по убыванию |
| **5** | Одномерный массив с элементами типа int | индексация элемента [];  унарные арифметические (++); | бинарные арифметические (+ и ‑ ) , причем для разных типов операндов; | Шелла по убыванию |
| **6** | Одномерный массив с элементами типа double. | присвоение =;  инициализация (); | унарные арифметические (--); | Быстрая по убыванию |
| **7** | Двумерный массив (матрица) с элементами типа char. | бинарные арифметические (+) и сравнения (> и <), причем только для однотипных операндов; | вывода в поток (<<) |  |
| **8** | Двумерный массив (матрица) с элементами типа int. | бинарные арифметические (-) и сравнения (==), причем только для однотипных операндов; | вывода в поток (<<) |  |
| **9** | Двумерный массив (матрица) с элементами типа double | индексация элемента [];  сложение матриц + | вывода в поток (<<) |  |
| **10** | Строка | присвоение =;  проверка на сколько одна строка больше другой - | сравнения (> и < и ), причем для разных типов операндов; | Пузырьком по возрастанию |
| **11** | Одномерный массив с элементами типа int | присвоение =;  объединение массивов + | унарные арифметические (--); | Шелла по возрастанию |
| **12** | Одномерный массив с элементами типа double. | пересечение массивов &  унарные арифметические (++); | бинарные арифметические (+ и ‑ ) , причем для разных типов операндов; | Быстрая по возрастанию |
| **13** | Двумерный массив (матрица) с элементами типа char. | бинарные арифметические (+ и ‑ ) , причем только для однотипных операндов; | вывода в поток (<<) |  |
| **14** | Двумерный массив (матрица) с элементами типа int. | сравнения (> и < и ==), | унарные арифметические (--); |  |
| **15** | Двумерный массив (матрица) с элементами типа double | индексация элемента [];  унарные арифметические (++); | бинарные арифметические (+ и ‑ ) , причем для разных типов операндов; |  |
| **16** | Строка | проверка на сколько одна строка больше другой –  инициализация (); | унарные арифметические (--); | Пузырьком по убыванию |
| **17** | Одномерный массив с элементами типа int | бинарные арифметические (+) и сравнения (> и <), причем только для однотипных операндов; | вывода в поток (<<) | Шелла по убыванию |
| **18** | Одномерный массив с элементами типа double. | бинарные арифметические (-) и сравнения (==), причем только для однотипных операндов; | вывода в поток (<<) | Быстрая по убыванию |
| **19** | Двумерный массив (матрица) с элементами типа char. | индексация элемента [];  инициализация (); | вывода в поток (<<) |  |
| **20** | Двумерный массив (матрица) с элементами типа int. | сложение матриц +  унарные арифметические (++); | сравнения (> и < и ), причем для разных типов операндов; |  |
| **21** | Двумерный массив (матрица) с элементами типа double | присвоение =;  индексация элемента []; | унарные арифметические (--); |  |
| **22** | Строка | сложение строки и объекта +=  возвращение подстроки [] | бинарные арифметические (+ и ‑ ) , причем для разных типов операндов; | Пузырьком по возрастанию |
| **23** | Одномерный массив с элементами типа int | бинарные арифметические (+ и ‑ ) , причем только для однотипных операндов; | вывода в поток (<<) | Шелла по возрастанию |
| **24** | Одномерный массив с элементами типа double. | сравнения (> и < и ==), | унарные арифметические (--); | Быстрая по возрастанию |
| **25** | Двумерный массив (матрица) с элементами типа char. | индексация элемента [];  унарные арифметические (++); | бинарные арифметические (+ и ‑ ) , причем для разных типов операндов; |  |
| **26** | Двумерный массив (матрица) с элементами типа int. | перемножение матриц &  инициализация (); | унарные арифметические (--); |  |
| **27** | Двумерный массив (матрица) с элементами типа double | бинарные арифметические (+) и сравнения (> и <) | вывода в поток (<<) |  |
| **28** | Строка | бинарные арифметические (-) и сравнения (==), | вывода в поток (<<) | Пузырьком по убыванию |
| **29** | Одномерный массив с элементами типа int | индексация элемента [];  пересечение массивов & | вывода в поток (<<) | Шелла по убыванию |
| **30** | Одномерный массив с элементами типа double. | Объединение массивов +  унарные арифметические (++); | сравнения (> и < и ), причем для разных типов операндов; | Быстрая по убыванию |

**4 лабораторная**

***Тема: Наследование***

**Общие требования к выполнению работы**

1. По полученному базовому классу создать классы наследников по двум разным ветвям наследования (B←P1←P11 и B←P2←P21):
2. во всех классах должны быть свои данные (характеристики объектов), расположенные в private или protected секциях;
3. во всех классах создать конструкторы инициализации объектов для всех классов (не забыть про передачу параметров в конструкции базовых классов);
4. во всех классах создать деструкторы;
5. во всех классах создать метод вывода объекта на экран (всех его параметров);
6. остальные методы создавать по необходимости.
7. Создать класс с множественным наследованием (P11 ← P3 → P21). Создать в этом классе выше описанные функции (в коде функций не использовать доступ к данным через оператор разрешения зоны видимости ::).



1. В головной функции создать экземпляр класса P3. Вывести содержимое экземпляра класса P3 на экран. Проверить, что получится, если не будет использовано виртуальное наследование.
2. Все возможные данные вводятся с клавиатуры с проверкой на корректность. Данные для объектов вводятся в main и передаются в объекты классов через конструкторы. Память выделять динамически.

|  |  |
| --- | --- |
| **№ задания** | **Задание к лабораторной работе** |
| **1** | Растение |
| **2** | Транспорт |
| **3** | Литература |
| **4** | Работа (отдел кадров) |
| **5** | Продукты |
| **6** | Хоккей |
| **7** | Техника (бытовая) |
| **8** | Кинематограф |
| **9** | Мебель |
| **10** | Одежда |
| **11** | Кино |
| **12** | Животные |
| **13** | История |
| **14** | Программирование |
| **15** | Мультфильмы |
| **16** | Администрирование |
| **17** | Смешарики |
| **18** | Дизайн |
| **19** | Кулинария |
| **20** | Футбол |
| **21** | Казино |
| **22** | Волейбол |
| **23** | Настольные игры |
| **24** | Родословная |
| **25** | Вирус |
| **26** | Гриб |
| **27** | Завод |
| **28** | Фигура |
| **29** | Дом |
| **30** | Книга |

**5 лабораторная**

***Тема: Виртуальные функции и абстрактные классы***

**Общие требования к выполнению работы**

1. Базовый класс должен быть абстрактным.
2. По полученному базовому классу создать классы наследников по двум разным ветвям наследования (B←P1←P11 и B←P2):
   1. во всех классах должны быть свои данные (характеристики объектов), расположенные в private или protected секциях;
   2. во всех классах создать конструкторы инициализации объектов для всех классов (не забыть про передачу параметров в конструкции базовых классов);
   3. во всех классах создать деструкторы;
   4. остальные методы создавать по необходимости.
3. Создать в базовом классе чисто виртуальные функции расчета (например, расчет площади фигуры и т.п.) и вывода объекта на экран (всех его параметров). Выполнить реализацию этих виртуальных функций в классах наследниках.
4. Задать в базовом классе деструктор как виртуальный.
5. В головной функции динамически создать массив указателей на базовый класс. Заполнить массив указателями на динамически создаваемые объекты производных классов (P1, P11, P2). Для каждого элемента массива проверить работу виртуальных функций. Удалить из памяти динамически выделенные объекты.
6. Всё взаимодействие с экземплярами производных классов производить через указатель на базовый класс.
7. Сгенерировать диаграмму классов

**Варианты:**

Варианты остаются с 4 лабораторной работы.