**Ministry of education and science of the Kyrgyz Republic**

**Kyrgyz State Technical University named after I.Razzakov**

**Faculty of Information Technologies**

**Department of Software of Computer Systems**

**Major: 710400 «Software Engineering»**

Report

Discipline: «**Object-Oriented Design**»

Software requirements

Done by**:** student of the SE(eng)- 1- 21

Muratbek kyzy Eliza

Checked by: Musabaev E.B.

Bishkek 2024

**Task1**

**Class Timer**

class Timer {

private:

int seconds;

public:

**Constructors-** Timer() : seconds(0) {}

Timer(int sec) : seconds(sec) {}

Timer(int min, int sec) : seconds(min \* 60 + sec) {}

**Destructor-** ~Timer() {

cout << "Таймер stop" << endl;

}

**Functional requirements**

1. The Timer class should have three overloaded constructors.Sorting records in the table in alphabetical order by the destination point.
2. The countdown should continue until it reaches 0.
3. When the countdown reaches 0, destructor display a message

**Task 2**

**Class Nomenclature**

class Nomenclature {

private:

string Product;

short int Price;

float RozNas;

int Quantity;

public:

Nomenclature(string product, short int price, float roznas, int quantity)

: Product(product), Price(price), RozNas(roznas), Quantity(quantity) {}

~Nomenclature() {

cout << "Товар удален" << endl;

}

**Functional requirements**

1. The Nomenclature class should have a constructor that initializes the object with a product name, wholesale price, retail markup, and quantity.Users should be able to add or remove student records dynamically.
2. The class should provide a function to calculate the net income for selling the product.
3. The class should provide a function to display all information about the product

**Task 3**

class Soft {

private:

string program;

string developer;

double size;

string date;

chrono::system\_clock::time\_point licenseDate;

public:

// Исправленный конструктор с правильным порядком инициализации

Soft(string prog, string dev, double size, string date, chrono::system\_clock::time\_point license)

: program(prog), developer(dev), size(size), date(date), licenseDate(license) {}

~Soft() {

cout << "Информация удалена!" << endl;

}

**Functional requirements**

1. The Soft class should have a constructor that initializes the object with a program name, developer, size, and license expiration date.Simulate a T-shaped sorting node with a main track and a side track.
2. The class should provide a function to calculate the number of days until the license expiration date.
3. The class should provide a function to display all information about the installed software

**Nonfunctional requirements**

1. Response Time: The program should quickly calculate results,
2. Availability-is the system available only when connected to the internet or does it also work offline
3. Usability-The interface must be simple and understandable
4. OS Windows 7 and higher
5. Processor intel core i5 and higher

**Answers to questions:**

1. **Что такое конструктор? Как он вызывается на выполнение?**

**Конструктор** - это метод класса, выполняющийся автоматически в момент создания объекта. Конструктор предназначен для инициализации объекта

* 1. Имя конструктора совпадает с именем класса.
  2. Конструктор вызывается автоматически при создании объекта

1. **Особенности конструктора, отличающие его от других методов класса?**

**Особенности конструктора:**

1. Имя конструктора совпадает с именем класса.
2. У конструкторов не существует возвращаемых значений.
3. Отсутствие типа возвращаемого значения. Нельзя получить указатель на конструктор.
4. Если пользователь не указал ни одного конструктора, компилятор создает его автоматически.
5. Конструктор не наследуется.
6. Конструктор можно перегружать по разным аргументам.
7. Существует конструктор по умолчанию (без аргументов) и конструктор с аргументами.
8. **Что такое конструктор по умолчанию? В каком случае следует его использовать?**

**Конструктор по умолчанию** - это конструктор, который может быть вызван без передачи аргументов.

Используется когда есть необходимость инициализации по умолчанию

1. **Способы инициализации в конструкторе по умолчанию? Какой способ инициализации предпочтительнее?**

1. Инициализация полей объекта с помощью списка инициализации:

2. Инициализация полей объекта с помощью присвоения им значений в теле конструктора.

Второй способ не содержит ошибок, но он не рекомендуется.

Причины:

1. Инициализация полей должна происходить до начала исполнения тела конструктора. Чтобы в теле конструктора можно было производить более сложные действия, чем обычная инициализация.
2. Список инициализации - это единственный способ задать первоначальные значения константам и ссылкам.

Так что первый способ предпочтительнее

1. **Что такое конструктор с аргументами? Когда используют конструктор с аргументами?**

**Конструктор с аргументами** – это конструктор, имеющий хотя бы один аргумент, может инициализировать поля значениями, переданными ему в качестве аргументов. К тому же этот конструктор значительно упрощает код программы.

Когда объект требуется инициализировать с конкретными значениями

Когда вы хотите, чтобы пользователь мог указать значения при создании объекта

1. **Какие преимущества у конструктора с аргументами перед методом Set()?**
   1. Инициализация при создании объекта
   2. Гарантия корректности состояния
   3. Конструктор с аргументами может быть более эффективным с точки зрения производительности
2. **Что такое деструктор? Для чего его используют?**

**Деструктор** – это метод, автоматически вызываемый при уничтожении объекта.

Деструктор используется для освобождения ресурсов, которые были выделены во время жизни объекта.

1. **Особенности деструктора**

Особенности деструктора:

1. Имеет имя, совпадающее с именем конструктора, а, следовательно, и класса.
2. Предваряется символом тильда.
3. Не возвращает значения.
4. Не имеет аргументов
5. **В какой последовательности выполняется конструкторы, в какой деструкторы классов?**

Конструкторы базовых классов вызываются в порядке их объявления. Деструкторы вызываются в обратном порядке.

1. **Можно ли перегружать конструктор?**

Да, можно. Рассмотрим два основных способа перегрузкиконструктора:

**способ 1**. Использование в одной программе обоих типов конструктора.

**Способ 2**. Для перегрузки используются конструкторы с одним именем, но отличающиеся количеством аргументов

1. **Что такое конструктор копирования?**

**Конструктор копирования –** это конструктор, с помощью которого возможно сделать третий способ инициализации полей объекта. При этом способе для инициализации полей объекта используются поля уже существующего объекта.