**Федеральное государственное автономное образовательное учреждение высшего образования   
"Национальный исследовательский университет   
"Высшая школа экономики"**

*Факультет социально-экономических и компьютерных наук*

Панфилов Даниил Сергеевич

**ЛАБОРАТОРНАЯ РАБОТА №9**

*Классы и объекты*

студента образовательной программы «Разработка информационных систем для бизнеса» по направлению подготовки *38.03.05 Бизнес-информатика*

Руководитель

ученая степень, ученое

звание, должность

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

И.О. Найданов

Пермь, 2023 год

**Оглавление**

[1. Постановка задачи. 3](#_Toc158378928)

[1.2. Общая постановка задачи. 3](#_Toc158378929)

[1.3. Частная подстановка задачи. 4](#_Toc158378930)

[2. Диаграмма классов для каждой части работы. 5](#_Toc158378931)

[3. Программа: 5](#_Toc158378932)

[4. Тестирование: 6](#_Toc158378933)

[5. Анализ покрытия кода тестами из Visual Studio 2022: 6](#_Toc158378934)

# Постановка задачи:

## Общая постановка задачи.

**Часть 1:**

1. Реализовать (в отдельном файле) определение нового класса/типа данных (закрытые атрибуты, свойства, конструкторы, инициализация и вывод атрибутов). Необходимо реализовать не менее 3 конструкторов: без параметров, с параметрами и конструктор копирования.
2. Для демонстрации работы с объектами написать главную функцию, в которой создаются объекты класса и выводится информация, которая содержится в атрибутах.
3. Написать функцию, выполняющую указанное в варианте действие. Рассмотреть два варианта реализации функции:
   1. статическую функцию;
   2. метод класса.
4. В основной программе продемонстрировать работу функций, объяснить разницу между статической и нестатической функциями.
5. Используя статическую компоненту класса подсчитать количество созданных в программе объектов.

**Часть 2:**

1. Добавить к реализованному классу указанные в варианте перегруженные операции.
2. Добавить метод public override bool Equals(object obj) {} для сравнения двух объектов реализованного класса (без этого метода не будут работать unit-тесты).
3. Написать демонстрационную программу, в которой создаются объекты пользовательских классов и выполняются указанные операции.

**Часть 3:**

1. Реализовать класс-коллекцию (в отдельном файле), полем которого является **одномерный массив** (**не использовать стандартные коллекции C#!)** из элементов заданного в варианте типа. Например, для класса Money нужно создать класс MoneyArray следующим образом:

class MoneyArray

{

Money[] arr;//одномерный массив элементов типа Money

. . . .

}

В классе реализовать:

* конструктор без параметров;
* конструктор с параметрами, заполняющий элементы случайными значениями;
* конструктор с параметрами, позволяющий заполнить массив элементами, заданными пользователем с клавиатуры;
* конструктор копирования, позволяющий создать копию коллекции, которая передается в конструктор как параметр, д.б. реализовано глубокое копирование.
* метод для просмотра элементов массива.

1. Реализовать индексатор для доступа к элементам коллекции. Предусмотреть проверку при выходе индекса за пределы массива и обработку этой ситуации с помощью **исключительных ситуаций.**
2. Написать демонстрационную программу, позволяющую создать массив разными способами (ручной ввод и случайная генерация) и распечатать элементы массива. Создать новую коллекцию на основе существующей, показать, что выполнено глубокое копирование.
3. В демонстрационной программе показать 4 варианта работы индексатора (запись объекта и получение объекта с существующим индексом, запись объекта и получение объекта с несуществующим индексом).
4. Написать функцию **в классе Program** для выполнения указанного в варианте задания (**использовать индексатор** и, если необходимо, перегрузить нужные для выполнения задачи операции), т.е. функция должна перебирать массив arr и обрабатывать его элементы, используя перегруженные операции пользовательского класса.
5. Подсчитать количество созданных объектов и созданных коллекций.

## Частная подстановка задачи.

**Часть 1:**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название класса** | **Атрибуты (поля)** | **Методы** |
| 2 | DialClock | int hours,  int minutes  (часы и минуты) | Вычислить угол между часовой и минутной стрелкой (в градусах) объекта DialClock.  Результат должен быть типа double, допустимо округление до 0,0001. |

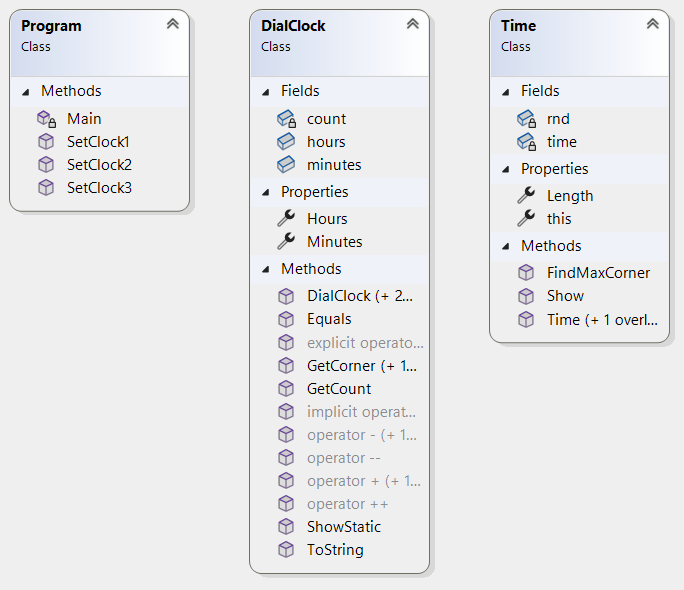
**Часть 2:**

|  |  |  |
| --- | --- | --- |
| **№** | **Название класса** | **Методы** |
| 2 | DialClock | Унарные операции:  ++ добавление минуты к объекту типа DialClock (учесть, что минут может быть от 0 до 59, а часов – от 0 до 23 включительно);  -- вычитание минуты из объекта типа DialClock (учесть те же ограничения).  Операции приведения типа:  bool (явная) – результатом является true, если угол между часовой и минутной стрелкой кратен 2,5, иначе – false;  int (неявная) – результатом является количество пройденных делений минутной стрелкой от начального положения обеих стрелок (когда обе стрелки указывают на число 12 циферблата часов) до текущего включительно.  Бинарные операции:  + DialClock dc, целое число (минуты) (лево- и правосторонние операции). Результат должен быть типа DialClock;  - DialClock dc, целое число (минуты) (лево- и правосторонние операции). Результат должен быть типа DialClock. |

**Часть 3:**

|  |  |  |
| --- | --- | --- |
| **№** | **Название класса** | **Задание** |
| 2 | DialClockArray | Найти объект DialClock с максимальным углом между часовой и минутной стрелкой. |

# Диаграмма классов для каждой части работы:



***Рисунок 2 – Диаграмма классов для класса Program, DialClock и Time.***

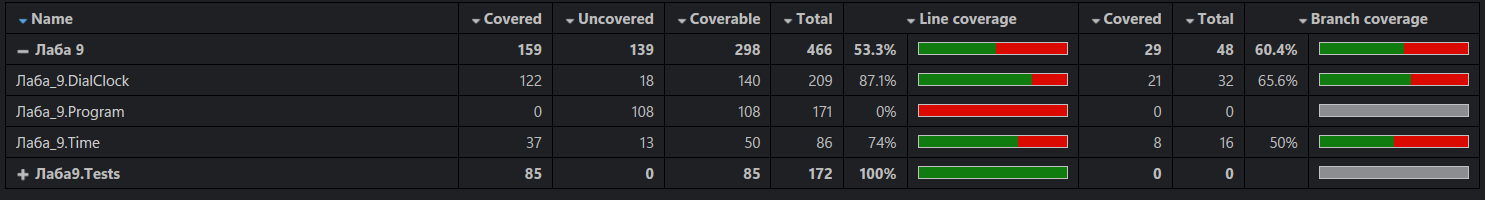
# Программа:

Ссылка на Git репозиторий: <https://github.com/hse-ris-23/9-D0ne1>

# Тестирование:

Тесты находятся в том же репозитории (<https://github.com/hse-ris-23/9-D0ne1>), в папке Лаба9.Tests, имя файла: UnitTest1.cs

# Анализ покрытия кода тестами из Visual Studio 2022:



***Рисунок 5 – Покрытия тестами программу, используя Fine Code Coverage.***