Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра Информатики

Дисциплина «Программирование»

**ОТЧЕТ**

к лабораторной работе №4

на тему:

**«Конструкторы. Статические члены класса.**

**Шаблон проектирования Singleton.»**

БГУИР 6-05-0612-02 34

|  |
| --- |
| Выполнил студент группы 353505  Хорошко Кирилл Николаевич |
|  |
| (дата, подпись студента) |
| Проверил ассистент каф. Информатики  РОМАНЮК Максим Валерьевич |
|  |
| (дата, подпись преподавателя) |

Минск 2024

# 1 Индивидуальное задание

Задание 1. Вариант 6.Спроектировать классы для выбранной предметной области: класс-контейнер, управляющий контейнеризируемым классом, и контейнеризируемый класс. Предусмотреть у классов наличие полей, методов и свойств. Предусмотреть перегрузку методов. Предусмотреть наличие статических методов. Применить к классу-контейнеру шаблон проектирования Singleton. Предметная область: Отдел кадров-Норма выработки. В классе хранить информацию о наименовании предприятия, числе работников, норму выработки часов в месяц (одинаковая для всех работников - класс Норма выработки), оплату за час, подоходный налог. Реализовать метод для подсчета общей выплаты по подоходному налогу. Реализовать возможность изменения (увеличения и уменьшения) нормы выработки.

# 2 Выполнение работы

Для выполнения данной лабороторной работы первым делом был создан проект, в этом проекте содержится 3 файла .cs. Каждый файл содержал свой класс, один из них содержал функцию Main. Ниже представлен код файла, который содержал класс Norm\_work. В этом классе предусматривалась перегрузка конструктора, был определен метод get\_norm\_work\_ocklock(), который возвращал значение поля norm\_work\_oklock, который устанавливал норму выработки часов в месяц, а также метод get\_norm\_work\_ocklock(), который позволял установить значение norm\_work\_oklock.

class Norm\_work

{

public Norm\_work() { }

public Norm\_work(int norm\_work\_oklock)

{

this.norm\_work\_oklock = norm\_work\_oklock;

}

private int norm\_work\_oklock;

public void set\_norm\_vork\_oklock(int norm\_work\_oklock)

{

this.norm\_work\_oklock = norm\_work\_oklock;

}

public int get\_norm\_work\_ocklock()

{

return norm\_work\_oklock;

}

}

После этого был создан класс-контейнер с названием “Station”. Он содержит поля “Name”, “NumberOfSeats”, “NumberOfSoldTickets” и “ticket”. Поле “ticket” является объектом класса “Ticket”. Так же для реализации шаблона Singleton был создан приватный конструктор класса, инициализирующий все поля, и пустой объект класса. Для применения этого конструктора был создан статический метод “GetStation”, который проверяет, созданный в классе объект на “ null ”. В случае, если поле “null”, то создается новый объект класса и возвращается ссылка на него. Иначе – только возвращается ссылка. То есть можно создать только 1 независимый объект класса, а все остальные лишь будут указывать на ту же область памяти. Для изменения цены билета было создано свойство “TicketCost”, которое обращается к соответствующему свойству класса “Ticket”. Так же был создан метод “CostOfUnsoldTickets”, вычисляющий стоимость непроданных бтлетов.

После этого был создан класс-контейнер с названием “Department”. Он содержит все поля, требуемые в условии задания. Так же для реализации шаблона Singleton был создан приватный конструктор класса, инициализирующий все поля, и пустой объект класса. Для применения этого конструктора был создан статический метод “GetStation”, который проверяет, созданный в классе объект на “ null ”. В случае, если поле “null”, то создается новый объект класса и возвращается ссылка на него. Иначе – возвращается ссылка на этот объект. Можно создать только 1 независимый объект класса. Для подсчета был реализован метод Sum\_Tax();

class Department

{

private static Department department;

public Department() { }

public Department(string name)

{

this.name = name;

}

public Department(string name, int amount\_worker)

{

this.name = name;

this.amount\_worker = amount\_worker;

}

public Department(string name, int amount\_worker, int norm\_work\_okclock)

{

this.name = name;

this.amount\_worker = amount\_worker;

this.norm\_work.set\_norm\_vork\_oklock(norm\_work\_okclock);

}

public Department(string name, int amount\_worker, int norm\_work\_okclock, int payment\_ocklock)

{

this.name = name;

this.amount\_worker = amount\_worker;

this.norm\_work.set\_norm\_vork\_oklock(norm\_work\_okclock);

this.payment\_oklock = payment\_ocklock;

}

public Department(string name, int amount\_worker, int norm\_work\_okclock, int payment\_ocklock, int tax)

{

this.name = name;

this.amount\_worker = amount\_worker;

this.norm\_work.set\_norm\_vork\_oklock(norm\_work\_okclock);

this.payment\_oklock = payment\_ocklock;

this.tax = tax;

}

public string name;

public int amount\_worker;

public Norm\_work norm\_work = new Norm\_work();

public int payment\_oklock;

private int Tax;

public int tax

{

get

{

return Tax;

}

set

{

if (value < 0 || value > 100)

{

Console.WriteLine("Вы ввели неверное значение. Значение по умолчанию 10%");

Tax = 10;

}

else

{

Tax = value;

}

}

}

public Department GetDepartment()

{

if (department == null)

{

department = new Department(this.name, amount\_worker, norm\_work.get\_norm\_work\_ocklock() , payment\_oklock, Tax);

}

return department;

}

public double Sum\_tax()

{

GetDepartment();

return (double)Department.department.amount\_worker \* Department.department.payment\_oklock \* Department.department.Tax / 100;

}

}

Основной код программы:

class Program

{

static void Main(string[] args)

{

char t = '1';

while (t != '0')

{

Department company = new Department();

Console.WriteLine("Введите имя компании");

company.name = Console.ReadLine();

Console.WriteLine("Введите число работников");

company.amount\_worker = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Введите норму выработки часов");

company.norm\_work.set\_norm\_vork\_oklock(Convert.ToInt32(Console.ReadLine()));

Console.WriteLine("Введите оплату за час");

company.payment\_oklock = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Введите подоходный налов в %");

company.tax = Convert.ToInt32(Console.ReadLine());

Console.Write("Общая выплата по подоходному налогу : ");

Console.WriteLine(company.Sum\_tax());

Console.WriteLine("Введите не 0, если хотите продолжить");

t = Convert.ToChar(Console.ReadLine());

}

}

}

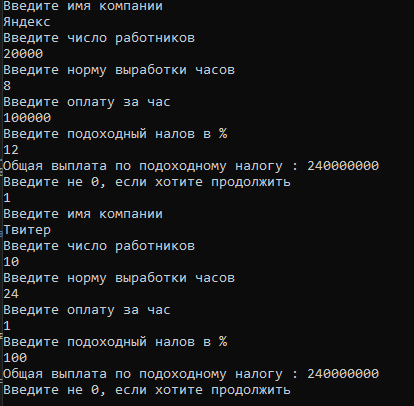


Рисунок 1 — Работа программы

Как можно заметить из рисунка 1, программа отобразила общую выплату подоходного налога одну и ту же, что указывает на соблюдение Singleton.

# Вывод

В ходе лабораторной работы я углубил свои знания в теми создания классов. Научился создавать конструкторы и описывать свойства классов. Изучил основу работы с шаблоном проектирования Singleton.