**Государственное образовательное учреждение**

**среднего профессионально образования «Тираспольский техникум информатики и права»**

**ОТЧЁТ**

**Практическое задание №6**

по дисциплине «Разработка программных модулей»

**тема: «Типы значений и ссылочные типы»**

Работу выполнил

студент группы 215

Кучеренко Данил

Руководитель,

преподаватель

Подсекина Т.С.

Тирасполь, 2025

**Вариант 2**

**Задание:** Клонирование объектов (поверхностное и глубокое)

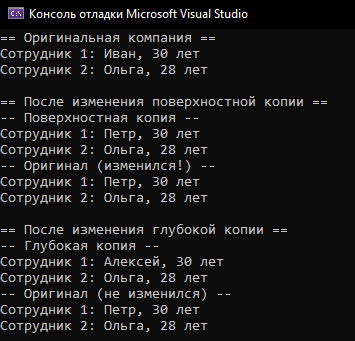
Создайте класс Company, содержащий массив Employee[].  
Задача: Реализуйте поверхностное и глубокое копирование компании.

* При поверхностном копировании измените сотрудника в копии.
* Докажите, что изменения повлияли на оригинал.
* Реализуйте глубокое копирование и убедитесь, что изменения теперь не затрагивают оригинал.

|  |
| --- |
| class Program  {  static void Main(string[] args)  {  // Оригинальная компания  var originalCompany = new Company(new Employee[]  {  new Employee("Иван", 30),  new Employee("Ольга", 28)  });  Console.WriteLine("== Оригинальная компания ==");  originalCompany.PrintEmployees();  // === Поверхностное копирование ===  var shallowCopy = originalCompany.ShallowCopy();  shallowCopy.Employees[0].Name = "Петр"; // Меняем имя первого сотрудника  Console.WriteLine("\n== После изменения поверхностной копии ==");  Console.WriteLine("-- Поверхностная копия --");  shallowCopy.PrintEmployees();  Console.WriteLine("-- Оригинал (изменился!) --");  originalCompany.PrintEmployees();  // === Глубокое копирование ===  var deepCopy = originalCompany.DeepCopy();  deepCopy.Employees[0].Name = "Алексей"; // Меняем имя в глубокой копии  Console.WriteLine("\n== После изменения глубокой копии ==");  Console.WriteLine("-- Глубокая копия --");  deepCopy.PrintEmployees();  Console.WriteLine("-- Оригинал (не изменился) --");  originalCompany.PrintEmployees();  }  }  class Employee  {  public string Name { get; set; }  public int Age { get; set; }  public Employee(string name, int age)  {  Name = name;  Age = age;  }  // Глубокое копирование сотрудника  public Employee DeepCopy()  {  return new Employee(Name, Age);  }  public override string ToString()  {  return $"{Name}, {Age} лет";  }  }  class Company  {  public Employee[] Employees { get; set; }  public Company(Employee[] employees)  {  Employees = employees;  }  // Поверхностное копирование  public Company ShallowCopy()  {  return new Company(this.Employees);  }  // Глубокое копирование  public Company DeepCopy()  {  Employee[] newEmployees = new Employee[Employees.Length];  for (int i = 0; i < Employees.Length; i++)  {  newEmployees[i] = Employees[i].DeepCopy();  }  return new Company(newEmployees);  }  public void PrintEmployees()  {  for (int i = 0; i < Employees.Length; i++)  {  Console.WriteLine($"Сотрудник {i + 1}: {Employees[i]}");  }  }  } |

Листинг 1 – программа определения max и min числа

Тест программы

****