РО 5.1.2 Формализация и алгоритмизация поставленных задач

Лекция 2

Введение в C#.
Язык C# и платформа .NET

- На сегодняшний момент язык программирования C# один из самых мощных, быстро развивающихся и востребованных языков в ИТ-отрасли. В настоящий момент на нем пишутся самые различные приложения: от небольших десктопных программок до крупных веб-порталов и веб-сервисов, обслуживающих ежедневно миллионы пользователей.
- C# уже не молодой язык и как и вся платформа .NET уже прошел большой путь. Первая версия языка вышла вместе с релизом Microsoft Visual Studio .NET в феврале 2002 года. Текущей версией языка является версия C# 10.0, которая вышла 8 ноября 2021 года вместе с релизом .NET 6.

- C# является языком с Си-подобным синтаксисом и близок в этом отношении к C++ и Java. Поэтому, если вы знакомы с одним из этих языков, то овладеть C# будет легче.
- C# является объектно-ориентированным и в этом плане много перенял у Java и C++. Например, C# поддерживает полиморфизм, наследование, перегрузку операторов, статическую типизацию. Объектно-ориентированный подход позволяет решить задачи по построению крупных, но в тоже время гибких, масштабируемых и расширяемых приложений. И C# продолжает активно развиваться, и с каждой новой версией появляется все больше интересных функциональностей.

- *.NET* является программной платформой, разработанной компанией *Microsoft*. Выделим некоторые из ее особенностей:
- Мультиязыковая ориентированность. Основным языком программирования для *.NET* является *C#*, помимо него платформа поддерживает еще ряд языков: *VB.NET*, *F#*, *C++* и др. Эта возможно благодаря тому, что в основе платформы лежит общеязыковая среда исполнения *Common Language Runtime* (CLR). Код на любом из поддерживаемых языков компилируется в сборку на языке *CIL* (C*ommon Intermediate Language*) – аналог ассемблера для *.NET*, что позволяет разрабатывать отдельные части приложения на разных языках.
- Кроссплатформенность. Еще одной особенностью платформы является поддержка большинства современных ОС (*Windows, MacOS, Linux*). Это позволяет заниматься разработкой приложений на языке *C#* на той ОС которая вам ближе, а запускать их можно на разных ОС и архитектурах.

- Обширная библиотека классов. Платформа включает в себя большое количество библиотек классов. Все они доступны для любого поддерживаемого языка. Внутри библиотек можно найти готовые инструменты для решения широкого круга задач.
- Широкий спектр решений для создания приложений. В рамках платформы *.NET* разработчик получает доступ к большому количеству инструментов для решения различных задач: *ADO.NET* и *Entity Framework* для работы с базами данных, *WPF* и *UWP* для разработки десктопных графических приложений, *ASP.NET* для разработки веб-приложений, *Blazor* для *frontend* разработки и т.д.

- Установка Visual Studio
- Задания1. docx

Структура программы

- В предыдущем разделе, мы создали простейшее приложение, которое выводит строку **Hello World!** и запустили на различных операционных системах. Рассмотрим подробнее структуру программы в файле ***Program.cs*.**

```
using System;

namespace FirstApp
{
    /*Моя первая программа.*/
    class Program
    {
        static void Main(string[] args) // Метод Main - точка входа в приложение
        {
            Console.WriteLine("Hello World!"); // WriteLine - выводит строку
        }
    }
}
```

Разберем построчно исходный код:

- using System – подключает пространство имен ***System*** с библиотекой классов.
- namespace FirstApp является контейнером для классов в рамках приложения, объединяя их в одно пространство имён.

- Фигурные скобки{} обозначают начало и конец блока кода.

- class Program – класс с данными и методами, который привносит функциональность в программу.
- Каждая строка исполняемого кода на C# должна находиться внутри класса. В нашем случае класс имеет имя ***Program***.

- Код внутри метода ***Main*** будет выполняться первым, этот метод является точкой входа в приложение.

- Console – класс пространства имен ***System***, имеющий метод WriteLine(), который используется для вывода текста. У нас будет выведена строка **«Hello World!»**.

- Если опустить строку using System, то придется писать System.Console.WriteLine() для вывода текста.

Примечание

Каждый оператор C # заканчивается точкой с запятой ; , к тому же язык чувствителен к регистру: «MyClass» и «myclass» имеют разное значение.

- Ещё одним немаловажным блоком кода являются комментарии: они бывают многострочными и однострочными, а задаются символами /**/ и // соответственно.

C# Типы данных

Переменные и типы данных

- Переменные служат для хранения данных и операций над ними. Объявлять переменные можно используя классический синтаксис:
  - тип имя = значение;

```
Пример: int number = 10;
```

- Поддерживается интерференция типов: компилятор C# способен автоматически определять тип переменной, используя ключевое слово var.

```
Пример: var number = 9;
```

- Для объявления констант добавляется ключевое слово const и их значения остаются фиксированными во время выполнения программы.

```
Пример: const int sizeArr = 10;
```

- C# имеет следующие базовые типы: ***int, double, string, char, bool, float, long***. Остановимся на базовых типах для простоты понимания фундамента языка.

- Типы.docx

Дом. Задания 05.09

- Установить программу
- Приводить 1 пример
- Типы языка C# изучить