Лабораторная работа №1

"Базовые типы данных языка C#. Работа с консолью"

Цель: научиться программировать операции консольного ввода-вывода с помощью фреймворка .NET, а также базовые математические операции с данными целочисленного и вещественного типа.

Ход работы.

1. Алгоритм - это определенная последовательность действий, которые необходимо выполнить, чтобы получить результат.

Любой алгоритм должен обладать свойствами:

- повторяемость (неизменность) получаемого результата при многократных расчетах с одними и теми же исходными данными;
- результативность - обязательное получение некоторого результата или сообщения о том, что данный алгоритм неприменим для решения поставленной задачи;
- массовость - возможность получения результата при различных исходных данных для некоторого класса сходных задач.
- дискретность - возможность разбиения алгоритма на отдельные элементарные действия.

На практике наиболее распространены следующие формы представления алгоритмов:
- словесная (на естественном языке)
- в виде блок-схемы (графический способ)
- в виде программы (на языках программирования)

2. Компиляция - это преобразование исходного кода в машинный.

Линковка - это соединение кода в один большой выполнимый файл.

Суть заложена в самих определениях: линковка "собирает" все части вместе, а компиляция переводя код на машинный, который обрабатывает компьютер.

3. Выделяют такие парадигмы программирования как:
- директивное (структурное)
- объектно-ориентированное
- декларативное (функционально-логическое)

1) в структурном программировании от входных данных полностью зависит последовательность выполнения команд.
Директивная программа предписывает, как достичь результата, пошагово описывая действия.

2) объектно-ориентированное строится на таких понятиях как наследование, полиморфизм и инкапсуляция, особое внимание уделяется данным, которые представляются в виде объектов, объекты взаимодействуют между собой с помощью механизма передачи сообщений.

3) Функциональное программирование основано на математическом понятии функции, которая не изменяет свое окружение. Программа состоит из совокупности определений функций, которые в свою очередь представляют собой вызовы других функций и предложений, управляющих последовательностью вызовов.

4. - После запуска Visual Studio на начальном экране выбрать „Создать проект".

- Для быстрого выбора напишем в поле поиска окна „Создание проекта" „Веб и консоль", затем выбрать C# в списке.

- После применения фильтров, выбрать шаблон „Консольное приложение (.NET Core)" и нажать „Далее"

- Готово! Остается только дать имя проекту и нажать „Создать".

Ctrl + F5 запускает код без отладки;

Shift + F6 - фокус на пред. часть.

5. Отладка в Visual Studio 2019 запускается нажатием клавиши F5. Но без поставленных точек останова программа то выполнит весь написанный код.

Точки останова - один из самых простых и важных компонентов надежной отладки. Точка останова указывает, где Visual Studio следует приостановить выполнение кода, чтобы разработчик мог проверить значения переменных или поведение памяти, либо выполнение ветви кода.

Точки останова ставятся с помощью клика мыши на левое поле строки кода.

6. .NET Framework состоит из двух главных компонентов: библиотеки базовых классов и CLR.

1) Библиотека базовых классов представляет собой коллекцию типов, которые тесно интегрируются со средой CLR. Она предоставляет типы, от которых управляемый код пользователя может наследовать функции.

2) CLR - Common Language Runtime - общая для языков среда исполнения NET-приложений.

Среда CLR управляет памятью, выполнением потоков, выполнением кода, проверкой безопасности, компиляцией и другими системными службами. Эти средства являются внутренними для управляемого кода, выполняемого в среде CLR.

7. Тип данных — Размер в байтах

bool	1
byte	1
char	2
decimal	16
double	8
float	4
int	4
long	8
string	(зависит от длины)

8. Переменная - именованный зарезервированный участок памяти, точнее ссылка на него, который можно использовать для хранения значений.

• Константа в C# объявляется следующим образом:

const [Тип] имя_константы = значение;

const - ключевое слово

[Тип] - тип данных, которые сохраняются в константе.

• Переменные объявляются след. образом:

Тип имя_переменной;

где Тип - тип данных, которые сохраняются в переменной.

• Инициализация позволяет задать начальное значение переменной при ее объявлении. Например: Тип имя_переменной = Знач.

9. Ключевое слово var ссылается на тип неявным способом. Это неявными любого типа. Реальный тип определит компилятор C#.

10. Преобразование значения переменной одного типа в значение другого называется приведением типа и бывает явным и неявным.

Неявное преобразование типов происходит автоматически, когда данные одного типа присваиваются переменной другого типа, при следующих условиях:

- оба типа совместимы;
- диапазон представления чисел целевого типа шире, чем у исходного.

Несмотря на всю полезность неявных преобразований, они не способны удовлетворить все потребности, поскольку лишь допускают расширяющие преобразования совместимых типов, а в остальных случаях приходится обращаться к приведению типов.

Для этого потребуется явное преобразование

(целевой_тип) выражение

целевой_тип обозначает тот тип, в который желаемо преобразовать указанное выражение.

11.

11. С префиксными операторами инкремента/декремента значение переменной сначала увеличивается/уменьшается, а затем вычисляется. Например:

```
int x = 5;
int y = ++x; // x = 6 и 6 присв. y.
```

А вот с постфиксными операторами инкремента/декремента компилятор создаёт временную копию переменной x, увеличивает или уменьшает оригинальный x, а затем возвращает копию только после возврата копия удаляется. Например:

```
int x = 5;
int y = x++; // x = 6, но y присв. 5.
```

12. Одним из способов форматирования вывода является использование метода String.format().

- Format принимает строку с "плейсхолдерами" типа {0}, {1} и т.д., а также набор аргументов, которые вставляются на место данных плейсхолдеров. В итоге генерируется новая строка.
- Метод ToString() - не только получает строковое описание объекта, но и может осуществлять форматирование.
- интерполяция строк - новая функцио-

полностью C# 6.0. Она призвана заменить форматирование строк.

Знак доллара перед строкой указывает, что будет осуществляться интерполяция. Внутри строки опять используются имена каждого {...}, только внутри фигурных скобок уже можно напрямую писать те выражения, которые мы хотим вывести.

Вывод: на данной лабораторной работе мы закрепили на практике навыки программирования консольного приложения на C#; принципами ООП. операции и методы для решения общих и индивидуальных задач; закрепили знания о базовых принципах ООП.