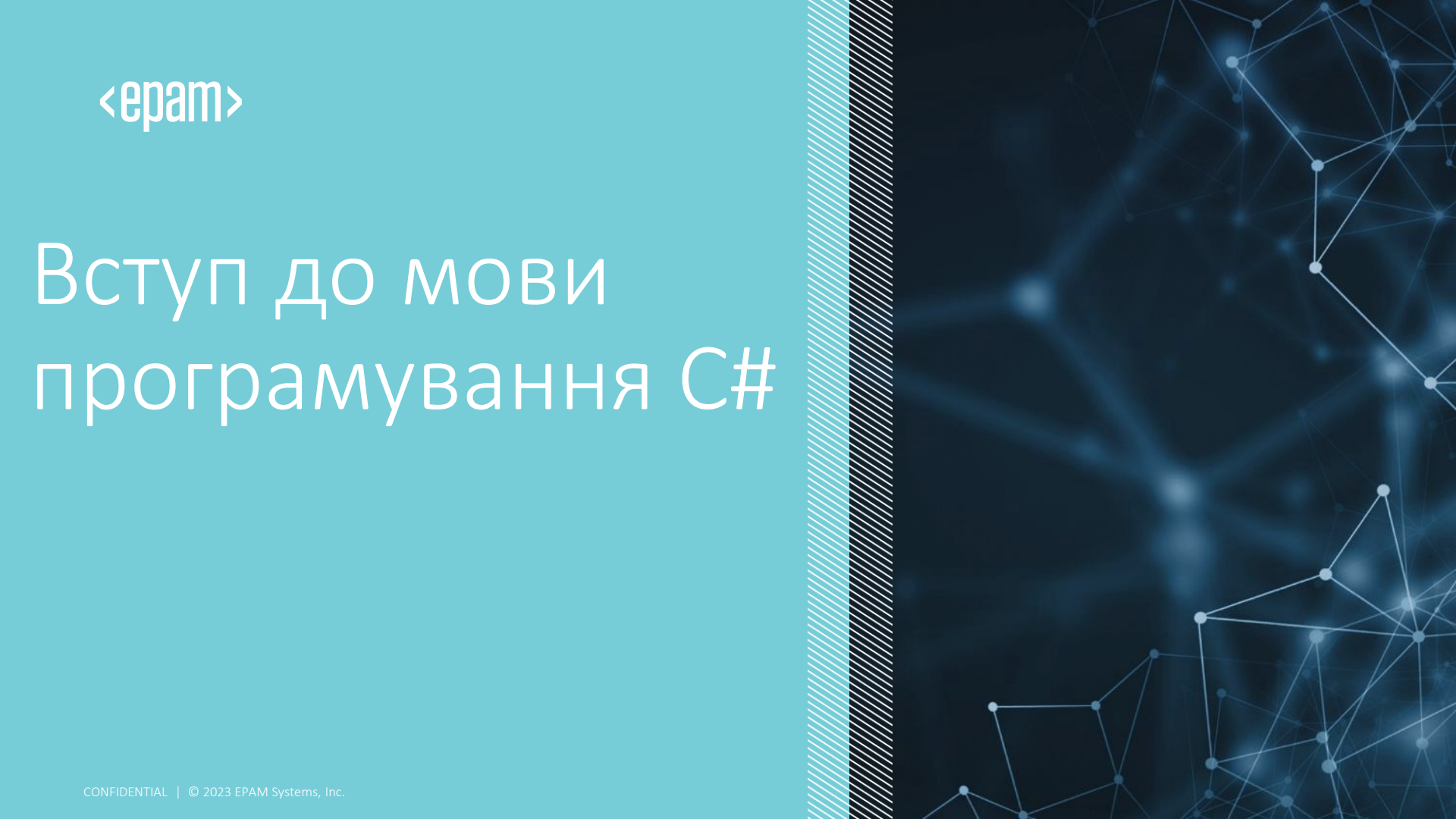


Вступ до мови програмування C#

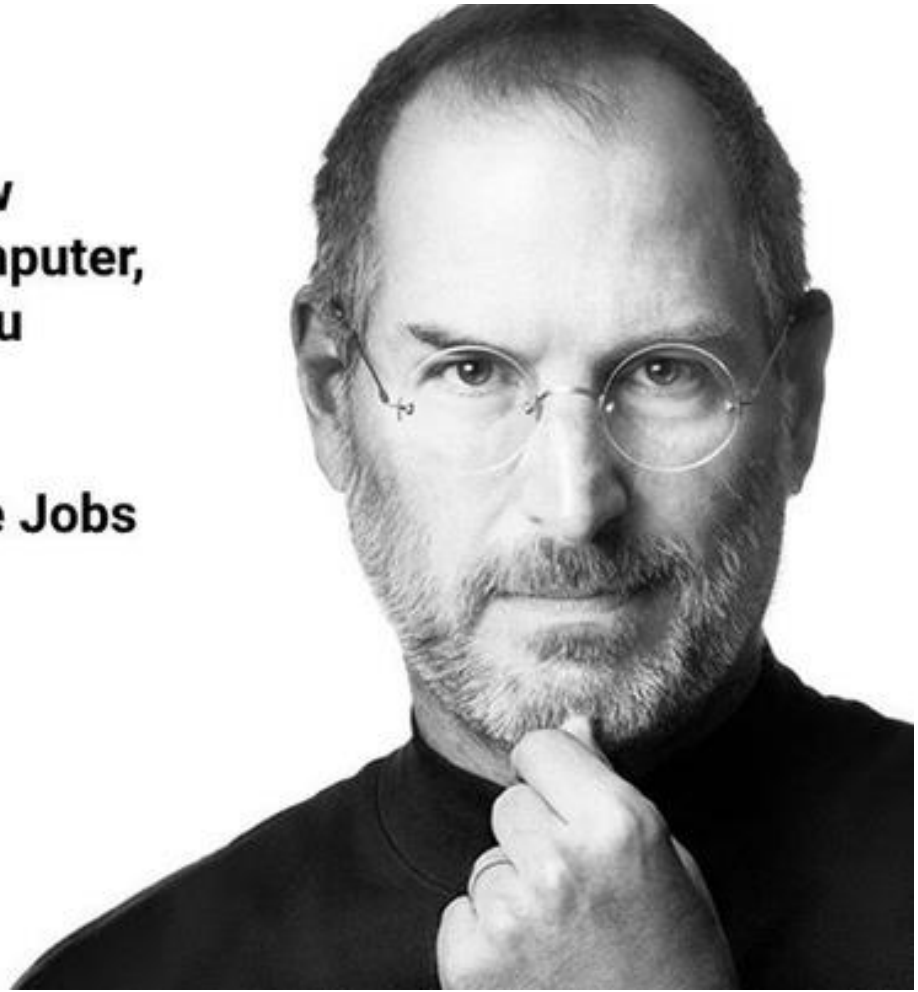
Вступ до мови програмування C#

Тема 1. Алгоритмізація. Програмування лінійних обчислювальних процесів

Навіщо потрібно вивчати програмування?

"Everyone should know
how to program a computer,
because it teaches you
how to think."

Steve Jobs

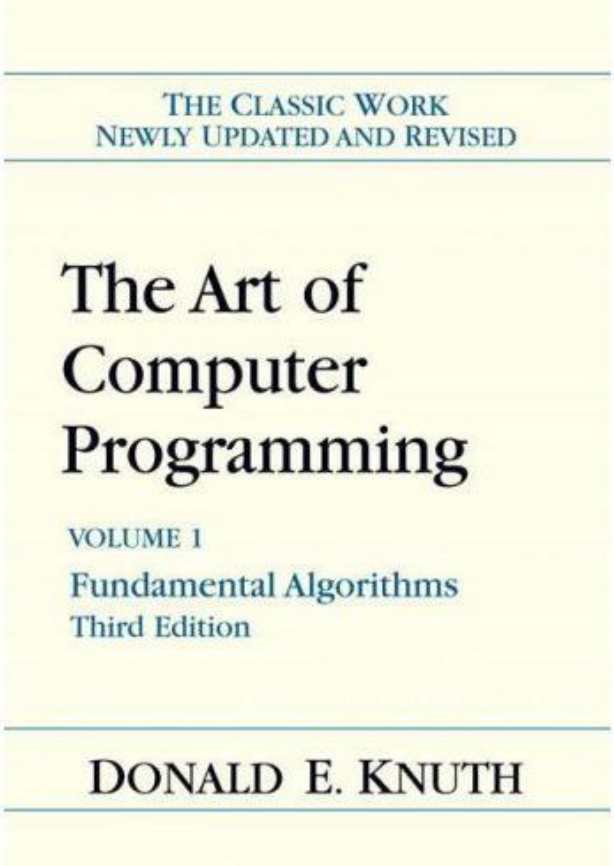


Кожен має освоїти "computational thinking" – так вважає Жанетт Вінг, корпоративний віце-президент Microsoft Research і в минулому – професор комп'ютерних наук в Університеті Карнегі-Меллон. "Computational thinking" допомагає людям мислити абстрактно і розділяти завдання на невеликі частини. Програмування – це один із способів навчитися цій навичці, вважає Вінг.

"Програмування є, можливо, найважливішою новою дисципліною постіндустріальної ери". Ніклаус Вірт, видатний швейцарський учений, фахівець у галузі інформатики.

«Уміння програмувати стало четвертою складовою грамотності. Кожен має знати, як наш цифровий світ працює, не лише інженери». Марк Серман, виконавчий директор Mozilla Foundation.

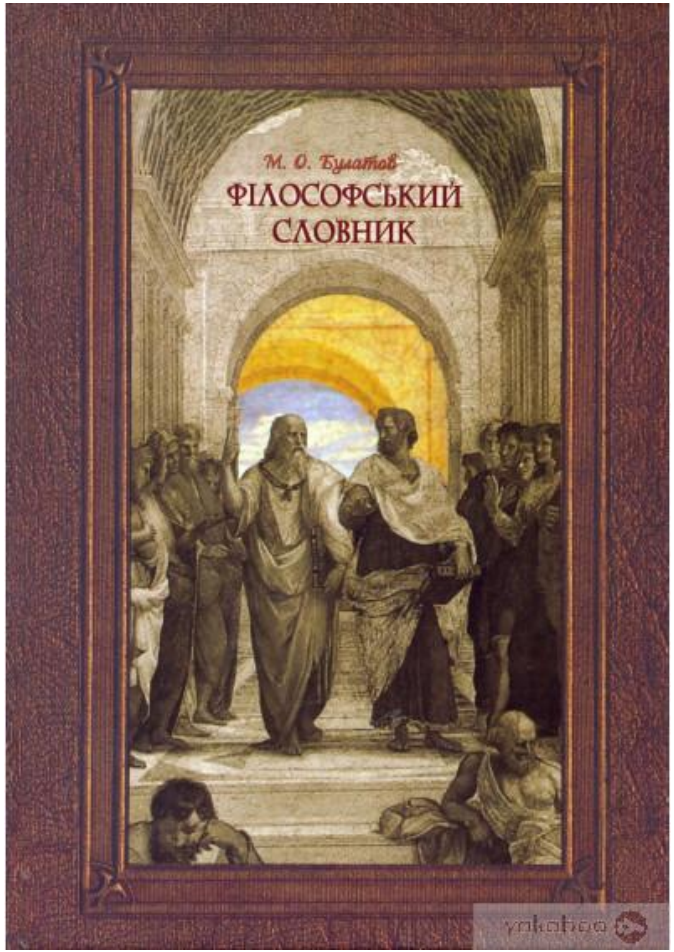
"Уміння програмувати буде корисним завжди". Білл Гейтс, засновник компанії Microsoft.



Процес підготовки програм для комп'ютера – дуже захоплююче заняття. І справа не тільки в тому, що воно виправдовує себе з економічної та наукової точок зору; воно може викликати такі ж естетичні переживання, подібні до тих, які відчувають творчі особистості при написанні музики або віршів

Donald Ervin Knuth

professor at Stanford University



Програмування - це особлива форма організації проблемного мислення і діяльності, що передбачає складання програми.

Філософський словник

Загальна характеристика мови C#

- C# (читається як “сі шарп”) – проста, потужна, сучасна, об’єктно орієнтована мова програмування від компанії Microsoft.
- Перша версія мови програмування C# була випущена в 2002 році. З того часу ця мова програмування постійно розвивається та вдосконалюється. Восени 2022 року було випущено версію C# 11.0
- Мова C# є універсальною мовою програмування, яка широко застосовується для створення застосунків будь-якого типу, починаючи зі звичайних десктопних програм і закінчуючи складними Web-сайтами, програмами для хмарних обчислень або застосунками для мобільних пристроїв.
- Мова C# також використовується для розробки комп’ютерних ігор на платформі Unity.
- Мова C# має простий та зрозумілий синтаксис, який дуже схожий на синтаксис таких мов, як C++ або Java. Тому, вивчивши мову C#, можна досить легко перейти на інші мови програмування.

Структура консольної програми

- Мова програмування C# є об’єктно-орієнтованою, тому навіть найпростіша програма повинна мати як мінімум один клас, в якому має містити метод Main. Метод Main є “точкою входу в програму”, оскільки саме з нього починається виконанная будь-якої програми.
- Саме тому під час при створення нової консольної програми середовище програмування Visual Studio або онлайн компілятори автоматично генерують новий клас з назвою Program, в якому оголошується єдиний поки що метод Main. Межі класу Program і межі методу Main також визначаються відповідними відкривальною та закривальною фігурними дужками.

```
1 using System;
2
3 public class Program
4 {
5     public static void Main()
6     {
7         Console.WriteLine("Hello World");
8     }
9 }
```

- У подальшому програміст може наповнювати метод Main програмним кодом, використовуючи конструкції мови C#, оголошувати нові методи в класі Program, нові класи тощо.

Введення-виведення даних у консольній програмі

- Консольна програма працює в консольному вікні, яке дозволяє здійснювати введення – виведення тексту за допомогою клавіатури. Простір імен System містить клас Console, в якому є методи, що забезпечують виведення текстової інформації у вікно консолі та зчитування тексту, який користувач програми вводить за допомогою клавіатури.
- Для виведення тексту в консольне вікно використовують методи Write та WriteLine класу Console. Обидва методи виводять у консольне вікно рядок тексту, який передається як параметр методу та записується в круглих дужках після назви методу. У мові програмування C# текстом чи рядком є послідовність будь-яких символів, обмежених подвійними лапками. Навіть один символ, обмежений подвійними лапками, також є рядком тексту.
- Приклади рядків:
- “Це рядок тексту”
- “Рядок може містити цифри 123 та інші символи !№;% ”
- “a”

Виведення тексту на екран

- Щоб програма вивела в консольне вікно будь-який текст (наприклад, “C# – це мова програмування”), потрібно в методі Main написати такий програмний код:

```
using System;

public class Program
{
    public static void Main()
    {
        Console.WriteLine("C# - це мова програмування");
    }
}
```

```
C# - це мова програмування
```

Методи Write та WriteLine

- Різниця між методами Write та WriteLine полягає у тому, що метод Write виводить рядок у вікно та залишає курсор у поточній позиції. Тому наступний текст буде виведено у цьому ж рядку відразу після попередньо виведеного тексту. А метод WriteLine виводить текст у вікно й автоматично переводить курсор на наступний новий рядок.
- Приклади використання цих методів і результати їх роботи:

```
using System;

public class Program
{
    public static void Main()
    {
        Console.WriteLine("Текстовий рядок 1 ");
        Console.WriteLine("Текстовий рядок 2 ");
        Console.WriteLine("Текстовий рядок 3 ");
    }
}
```

```
Текстовий рядок 1
Текстовий рядок 2
Текстовий рядок 3
```

```
using System;

public class Program
{
    public static void Main()
    {
        Console.Write("Текстовий рядок 1 ");
        Console.Write("Текстовий рядок 2 ");
        Console.Write("Текстовий рядок 3 ");
    }
}
```

```
Текстовий рядок 1 Текстовий рядок 2 Текстовий рядок 3
```

Типи даних

- Комп'ютерні програми розробляються для того, щоб виконувати обробку певних даних. Дані можуть бути числами (цілими або дійсними), рядками, символами або логічними значеннями.
- Одним із основних понять програмування є тип даних. Тип даних визначає, діапазон значень, множину допустимих операцій із цими значеннями і спосіб збереження значень та виконання операцій. Таким чином, будь-яка інформація, якою оперує програма, відноситься до певного типу даних. Основні типи даних в мові C# наведені у таблиці:

Тип C#	Опис	Діапазон
sbyte	8-бітове знакове ціле	-128…127
byte	8-бітове беззнакове ціле	0…255
short	16-бітове знакове ціле	-32768…32767
ushort	16-бітове беззнакове ціле	0…65535
int	32-бітове знакове ціле	-2147483648… 2147483647
uint	32-бітове беззнакове ціле	0…4294967295
long	64-бітове знакове ціле	-9223372036854775808… 9223372036854775807
double	Число з плаваючою комою подвоєної точності	-1.797×10^{-308}…1.797×10^{308}
decimal	Десятковий тип	$\pm7.923\times10^{28}$
bool	Логічний тип	false, true
char	Символ Unicode	
string	Послідовність символів Unicode	

Змінні

- Змінна – це ділянка оперативної пам’яті певного розміру (від одного до декількох байт), де під час виконання програми зберігаються дані. Кожна змінна має тип і ім’я (назву). Ім’я – це послідовність літер та цифр, яка починається з літери.
- Усі змінні в програмі повинні бути оголошені до того, як будуть використовуватися. Для того, щоб оголосити змінну, потрібно вказати її тип та ім’я. Декілька змінних одного типу можна оголошувати в одному рядку, вказуючи їх імена через кому.
- Приклади оголошення змінних в програмі на мові C#:

```
int number1;

int x1, x2, x3;

double suma;

char c;

bool f;

string str;
```

Присвоєння значень змінним

- Для того, щоб змінна почала зберігати певне значення, це значення їй потрібно присвоїти. В мові програмування C# для того, щоб вказати, що змінній присвоюється значення, треба написати ім’я змінної, потім поставити символ ‘=‘ (знак «дорювнює»), а потів вказати значення, яке буде присвоюватися змінній.
- Приклади:

```
int x;
double y;
x = 10;
y = 3.15;
```

- Можна поєднувати оголошення змінної і присвоєння їй значення, записуючи в один рядок:

```
int x = 10;
double y = 4.95;
 int number1 = 5, number2 = 10, suma = 0;
```

Присвоєння значень змінним

- Присвоювати можна не лише числа, а і вирази, які складаються з чисел, інших змінних, які вже мають значення та знаків математичних операцій. Значення виразів обчислюється за правилами математики відповідно пріорітету кожної операції. Спочатку виконуються операції множення та ділення, а потім додавання та віднімання. Для зміни порядку обчислення виразу використовують круглі дужки, які мають найвищий пріорітет.
- Приклад.

```
int x = 10;

int y = 20;

int s =  x + y;

int  sum = x + 35 * s + 20 * ( x - y );
```

Введення тексту (даних рядкового типу)

- Введення або зчитування рядка в консольному вікні здійснюється за допомогою метода ReadLine класу Console. Результатом роботи цього методу буде рядковий тип даних, тобто string. Для того щоб введений рядок можна було використовувати в програму, потрібно оголосити змінну типу string і присвоїти їй рядок, який буде уведено методом ReadLine. Наприклад:

```
1  using System;
2
3  public class Program
4  {
5      public static void Main()
6      {
7          Console.WriteLine("Як твоє ім'я ?");
8          string name = Console.ReadLine();
9          Console.WriteLine("Привіт, " + name + "!");
10     }
11 }
```

- Якщо у відповідь на запитання про ім’я ввести, наприклад, рядок “Петро”, то у вікні з’явиться текст “Привіт, Петро!”.

Введення чисел

- Треба зазаначити, що метод ReadLine завжди повертає результат типу “рядок”, навіть коли користувач програми вводить число або один символ. У цьому разі потрібно виконати приведення рядка до відповідного типу. Для цього в мові програмування C# є дві можливості. Перша – використання відповідних методів класу Convert. Друга – використання методу Parse, який має усі стандартні типи.

- Наприклад, ціле число можна ввести так:

  ```
  int x = Convert.ToInt32(Console.ReadLine());
  ```

  або так:

  ```
  int y = int.Parse(Console.ReadLine());
  ```

- Таким же чином можна ввести дані інших стандартних типів.

Форматування виводу

- Для виводу на консоль можливо організувати форматування виводу, тобто отримати значення на екрані у зручному для сприйняття вигляді. Для цього у текстову константу – аргумент методу Console.WriteLine – треба помістити так звані плейсхолдери (число у фігурних дужках) з номером потрібного елемента списку виводу, причому нумерація починається з нуля. На місці кожного з таких плейсхолдерів на екрані з’явиться відповідне значення. Або використати інтерполяцію рядків, додавши перед рядком символ $.
- Приклади:

```
using System;

public class Program
{
    public static void Main()
    {
        int x = 10;
        int y = 20;
        Console.WriteLine("{0} + {1} = {2}", x, y, x+y);
    }
}
```

```
10 + 20 = 30
```

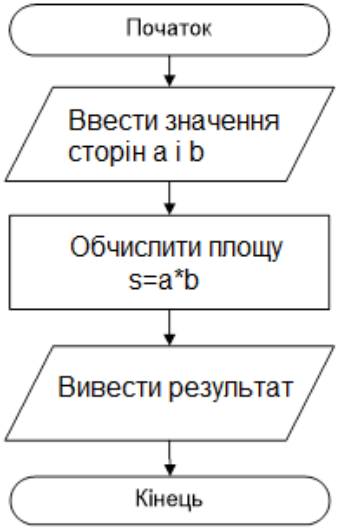
```
using System;

public class Program
{
    public static void Main()
    {
        int x = 10;
        int y = 20;
        Console.WriteLine($"{x} + {y} = {x+y}");
    }
}
```

```
10 + 20 = 30
```

Лінійний обчислювальний процес

- Алгоритми, в яких використовується тільки структура слідування, називаються лінійними.
- Лінійний алгоритм описує обчислювальний процес, у якому етапи виконуються послідовно, тобто лінійно (один за одним незалежно від жодних умов). Наприклад, для визначення площі прямокутника треба спочатку увести значення однієї сторони, потім – другої, потім перемножити ці значення і вивести результат обчислень. Приклад алгоритму:



Приклад лінійної програми

- Для визначення площі прямокутника треба спочатку увести значення однієї сторони, потім – другої, потім перемножити ці значення і вивести результат обчислень. Приклад програми:

```
using System;

public class Program
{
    public static void Main()
    {
        Console.WriteLine("Введіть значення сторони a");
        double a = double.Parse(Console.ReadLine());
        Console.WriteLine("Введіть значення сторони b");
        double b = double.Parse(Console.ReadLine());
        double s = a*b;
        Console.WriteLine("Площа прямокутника зі сторонами {0} і {1} дорівнює {2}", a, b, s);
    }
}
```

```
Введіть значення сторони a
4
Введіть значення сторони b
5
Площа прямокутника зі сторонами 4 і 5 дорівнює 20
```

Домашнє завдання

- Один зошит коштує x гривень, а одна ручка – y гривень. Петрик вирішив купити n зошитів та m ручок. Напишіть програму, яка дозволить визначити загальну суму покупки, якщо ввести усі значення, тобто кількість зошитів та ручок, а також їх ціну.

За бажанням можна надіслати виконане домашнє завдання на перевірку одним з двох можливих варіантів:

- В особистий чат в Teams: @Oleksandr Shcherbakov
- На електронну пошту: oleksandr_shcherbakov@epam.com

Надсилати потрібно лише текст програми, скопіювавши його в повідомлення в чаті або приєднати до електронного листа файл з назвою Program.cs. Можна також надіслати посилання на власний репозиторій в Git.