МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

Національний технічний університет України

"Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Програмування. Частина 2. Основи програмування

Практична робота №9

Тема: String

Варіант №11

Виконав: Коваль Богдан

студент групи ІС-31

Перевірив:

Майер І.С.

Київ-2024

Хід роботи

1. Ознайомитись з API класу String. Особливу увагу звернути на методи:

▪ Split

▪ ToCharArray

▪ ToLower

▪ ToUpper

2. Виконати завдання з таблиці 2 відповідно до свого варіанту у таблиці 1.

◦ Кожне завдання має бути реалізовано як окремий клас.

◦ Кожен клас має складатись щонайменше з двох методів:

▪ public static void Main() - точка входу. Містить код, що кілька разів знаходить результат

завдання при різних значеннях аргументів та параметрів. Для перевірки мають бути

присутні як дозволені так і заборонені комбінації аргументів та параметрів.

▪ Метод, що реалізує задане завдання. Метод має перевіряти аргументи та у разі їх

помилковості аварійно закінчувати свою роботу шляхом викидання стандартного

виключення ArgumentOutOfRangeException або ArgumentNullException(дивись л/р

No 3). В жодному разі цей метод не повинен напряму взаємодіяти з користувачем

через консоль або інший UI (ніколи не змішуйте бізнес-логіку та користувацький

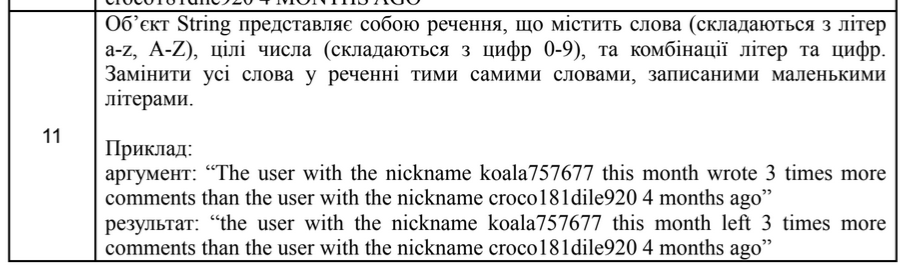
інтерфейс).

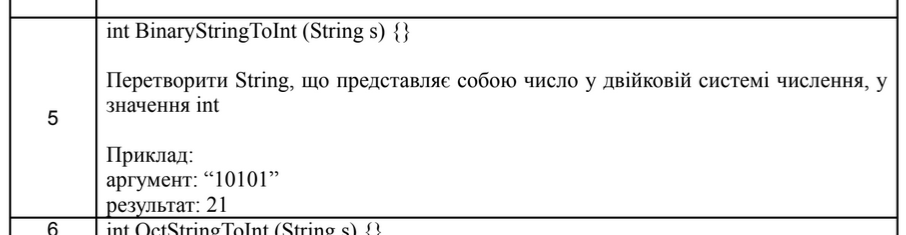
▪ Не можна використовувати для конвертування значень стандартні чи сторонні

бібліотеки.

◦ Клас може містити інші допоміжні методи.

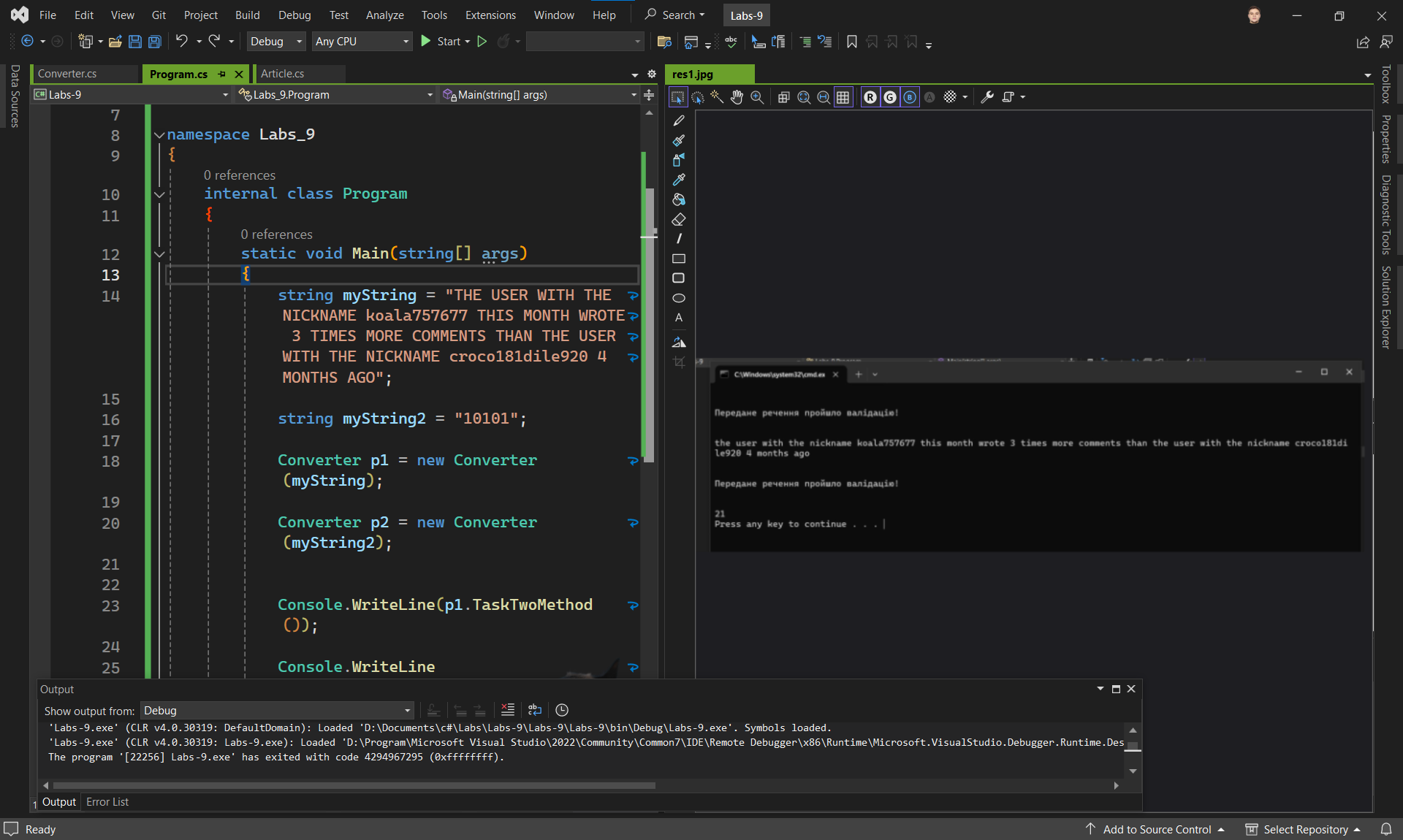
3. Відповісти на контрольні питання

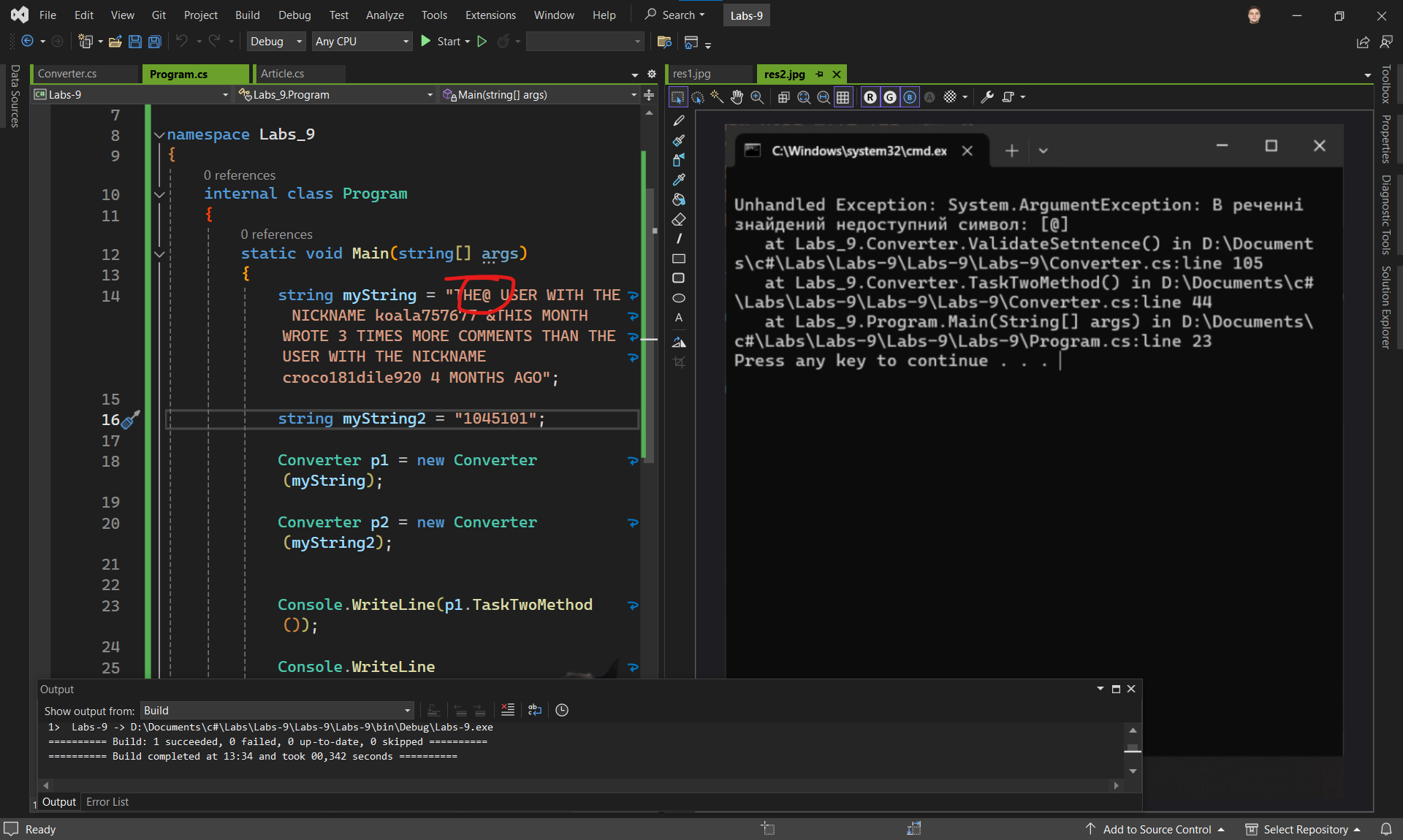


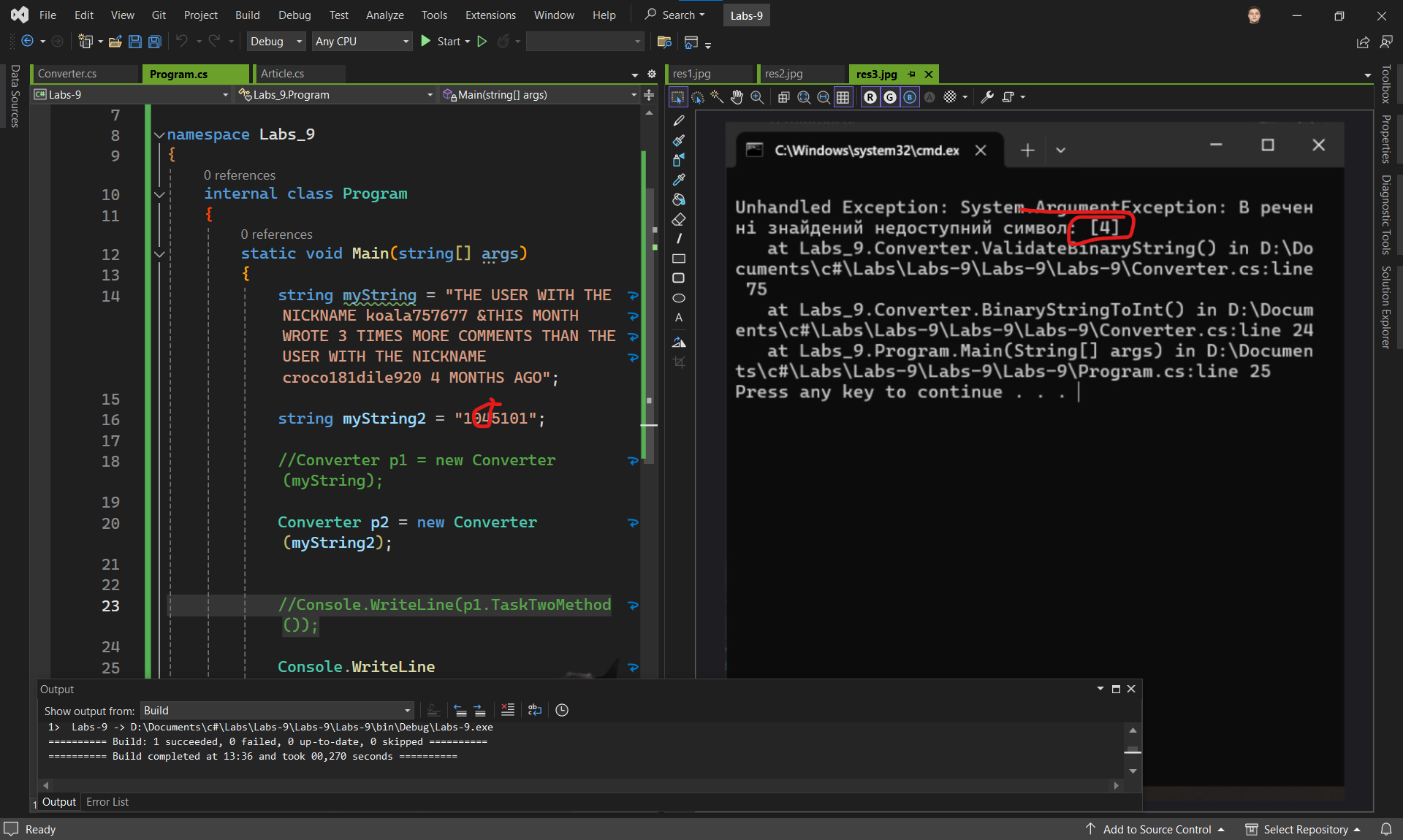


Code:

| using System; using System.Text;  namespace Labs\_9 {  class Converter  {  public string \_string { get; private set; }  public Converter(string stringDate)  {  if (string.IsNullOrEmpty(stringDate))  {  throw new ArgumentNullException("Переданий об'єкт String не може бути пустим!");  }  \_string = stringDate;  }   */// <summary>*  */// Завдання №5*  */// </summary>*  */// <returns></returns>*  public int BinaryStringToInt()  {  ValidateBinaryString();  double result = 0;   for (int i = \_string.Length - 1; i >= 0; i--)  {  if (\_string[i] == '1')  {  result += Math.Pow(2, i);  };    }  return (int)result;  }   */// <summary>*  */// Завдання №21*  */// </summary>*  */// <returns></returns>*  public string TaskTwoMethod()  {  ValidateSetntence();  string result = "";   foreach (var elem in \_string.Split(' '))  {   if (IsWord(elem))  {  foreach (var let in elem)  {  result += char.ToLower(let);  }  result += ' ';  }  else result += $" {elem} ";  }  *// Удалить зайвий пробіл :D*  result = result.Substring(0, result.Length - 1);  return result;  }    */// <summary>*  */// Система валiдацiя для завдання №1 перевiряє строку на наявнiсть недопустимих символiв*  */// У випадку знаходження викликає помилку*  *///*  */// Логiка валiдацiї - White List (Бiлий список) система пропускає лише обранi символи,*  */// знаходження вiдбувається за iндексом в системi ASCII*  *///*  */// Значення в системi ASCII крайнiх допустимих символiв*  */// White List:*  */// 1) Цифри: 0-9 [0 - 48, 1 - 49]*   */// </summary>*  */// <param name="String"></param>*  */// <exception cref="ArgumentNullException"></exception>*  public void ValidateBinaryString()  {  foreach (var el in \_string)  {  int ascii = (int)el;  if (!(ascii == 48 || ascii == 49))  {  throw new ArgumentException($"В реченнi знайдений недоступний символ: [{el}]");  }  }  Console.WriteLine("\n\nПередане речення пройшло валiдацiю!\n\n");  }    */// <summary>*  */// Система валiдацiя для завдання №2 перевiряє строку на наявнiсть недопустимих символiв*  */// У випадку знаходження викликає помилку*  *///*  */// Логiка валiдацiї - White List (Бiлий список) система пропускає лише обранi символи,*  */// знаходження вiдбувається за iндексом в системi ASCII*  *///*  */// Значення в системi ASCII крайнiх допустимих символiв*  */// White List:*  */// 1) Лiтери: a-z [a - 97, z - 122]; A-Z [A - 65, Z - 90]*  */// 2) Цифри: 0-9 [0 - 48, 9 - 57]*  */// 3) Пробіл: [' ' - 32]*  *///*  */// </summary>*  */// <param name="String"></param>*  */// <exception cref="ArgumentNullException"></exception>*  public void ValidateSetntence()  {  foreach (var el in \_string)  {  int ascii = (int)el;  if (!(ascii >= 97 && ascii <= 122 || ascii >= 65 && ascii <= 90  || ascii >= 48 && ascii <= 57 || ascii == 32))  {  throw new ArgumentException($"В реченнi знайдений недоступний символ: [{el}]");  }  }  Console.WriteLine("\n\nПередане речення пройшло валiдацiю!\n\n");  }   public bool IsWord(string word)  {   foreach(var letter in word)  {  int letter\_ascii = (int)letter;  if (!(letter\_ascii >= 97 && letter\_ascii <= 122 || letter\_ascii >= 65 && letter\_ascii <= 90))  {  return false;  }  }   return true;  }  } } |
| --- |







Відповіді на контрольні питання в контексті C#

*Питання 1: Що таке незмінний (Immutable) об’єкт? Чому об'єкти класу String в C# зробили незмінними (Immutable)?*

Незмінний (Immutable) об'єкт — це об'єкт, який не можна змінити після його створення. Якщо необхідно внести зміни, потрібно створити новий об'єкт. Клас String в C# є незмінним з тих же причин, що й в інших мовах програмування: для забезпечення безпеки, надійності та ефективності. Незмінність запобігає неочікуваним змінам рядків під час передачі, забезпечує можливість кешування і спрощує управління пам'яттю.

*Питання 2: Що таке регулярні вирази в C#? Наведіть приклади регулярних виразів.*

Регулярні вирази (regex) в C# — це набір правил або патернів для пошуку або валідації тексту. Вони використовуються з класом System.Text.RegularExpressions.Regex. Приклади регулярних виразів у C#:

- \d+ — один або більше чисел.

- [a-zA-Z]+ — один або більше латинських букв.

- \s\* — будь-яка кількість пробілів.

- ^Hello — рядок, що починається зі слова "Hello".

- \bword\b — слово "word" як окреме слово.

*Питання 3: В чому різниця між оператором == та методом Equals в C#?*

Оператор == в C# порівнює об'єкти за їх референсами. Він вказує, чи два об'єкти є одним і тим самим об'єктом у пам'яті. Метод Equals використовується для перевірки логічної рівності, тобто порівнює об'єкти за їх змістом. Цей метод може бути перевизначеним у користувацьких класах, щоб забезпечити логічне порівняння.

*Питання 4: Для чого потрібен клас StringBuilder в C#?*

StringBuilder в C# використовується для побудови та редагування рядків без створення нових об'єктів. Це особливо корисно для операцій, де рядки часто змінюються або доповнюються, що забезпечує вищу продуктивність у порівнянні з класом String.

*Питання 5: Як найпростіше видалити пробіли на початку та кінці об'єкту String в C#?*

У C# найпростіший спосіб видалити пробіли з початку та кінця рядка — використовувати метод Trim(). Він видаляє всі пробіли, табуляції та інші невидимі символи з обох кінців рядка. Приклад:

| string str = " Hello, World! "; string trimmedStr = str.Trim(); *// "Hello, World!"* |
| --- |