**Контрольні питання**

1. **Поясніть особливості створення та використання рядків.** Рядки в C# є незмінними (immutable) посилальними типами, що означає, що будь-яка операція, яка здається модифікацією, насправді створює новий об'єкт рядка в пам'яті. Це забезпечує безпеку, але для частих змін рекомендується використовувати StringBuilder; вони також підтримують Unicode і оптимізовані через пул інтернованих рядків для однакових літералів.

2. **Наведіть приклади класів і методів, що перетворюють рядок на значення іншого типу даних.** Для перетворення рядків на інші типи даних у C# використовуються методи Parse() (наприклад, int.Parse("123")), які генерують виняток у разі помилки, або безпечніші методи TryParse() (наприклад, int.TryParse("abc", out num)), що повертають булеве значення успіху. Також можна використовувати статичні методи класу System.Convert (наприклад, Convert.ToInt32("456")) для універсального перетворення між базовими типами.

3. **За допомогою якого методу у С# значення інших типів даних перетворюються на рядок?** Значення інших типів даних у C# найчастіше перетворюються на рядок за допомогою нестатичного методу ToString(), доступного для всіх типів, оскільки він успадковується від System.Object (наприклад, 123.ToString()). Також можна використовувати інтерполяцію рядків ($"") або статичний метод string.Join() для об'єднання колекцій.

4. **Для чого потрібні регулярні вирази? Які класи реалізують можливість створювати та використовувати регулярні вирази?** Регулярні вирази є потужним інструментом для пошуку, валідації, заміни та виділення тексту за складними шаблонами. У C# для роботи з ними використовується простір імен System.Text.RegularExpressions, головним класом якого є Regex, що надає методи для всіх основних операцій. Класи Match та MatchCollection представляють результати пошуку.