Лабораторна робота № 8.

Тема: ***Символьні рядки. Регулярні вирази.***

Мета роботи: Ознайомлення з засобами обробки текстової інформації в C#.

Теоретичні відомості

Мова C# надає розв'язку широкий набір засобів обробки тестової інформації: окремі символи, масиви символів, змінювані й незмінні рядки та регулярні вирази.

Клас char

Символьний клас char, заснований на класі `System.Char`, що використовує двохбайтове кодування `Unicode` представлення символів. У цьому класі визначені статичні методи, що дозволяють задати вид і категорію символу, а також перетворити символ у верхній або нижній регістр і в число. Для цього типу в мові визначені символьні сталі - символьні літерали. Сталі можна задавати:

- символом, укладеним в одинарні лапки;
- escape-послідовністю, що задає код символу;
- Unicode-послідовністю, що задає Unicode-код символу.

```
char ch1='A', ch2 ='\x5A', ch3='\u0058';
char ch = new char();
```

Три символьні змінні ініціалізовані сталими, значення яких задано трьома різними способами. Змінна `ch` оголошується в об'єктному стилі, використовуючи `new` і виклик конструктора класу. Тип `char`, як і всі типи `C#`, є класом. Цей клас успадковує властивості та методи класу `object`.

Масив символів, як і масив будь-якого іншого типу, побудований на основі базового класу `Array`. Застосування цих методів дозволяє ефективно вирішувати деякі задачі. У мові `C#` визначений клас `char[]` і його можна використовувати для вистави рядків постійної довжини. Оскільки масиви в `C#` динамічні, то розширюється клас задач, у яких можна використовувати масиви символів.

Клас string

Основним типом при роботі з рядками є тип `string`, що задає рядка змінної довжини. Клас `string` у мові `C#` відноситься до типів посилання типів та призначений для робот з рядок символів у кодуванні `Unicode`. Йому відповідає базовий тип класу `System.String` бібліотеки `.Net`. Над рядками - об'єктами цього класу - визначений широкий набір операцій.

Над рядками визначені наступні операції:

- присвоєння (=);
- дві операції перевірки еквівалентності (= =) і (!=);
- конкатенація або зчеплення рядків (+);
- індексації ([]).

Кожний об'єкт `string` - це незмінна послідовність символів `Unicode`, тобто методи, призначені для зміни рядків, повертають змінені копії, вихідні ж рядки

залишаються незмінними. Невикористані "старі" копії автоматично віддаляються збирачем сміття.

Рядка типу StringBuilder

Можливості, надавані класом **`string`**, широкі, однак вимога незмінності його об'єктів може виявитися незручним. У цьому випадку для роботи з рядками застосовується клас **`StringBuilder`**, певний у просторі імен **`System.Text`**, що й дозволяє змінювати значення своїх екземплярів.

Регулярні вирази

Стандартний клас string дозволяє виконувати над рядками різні операції, у тому числі пошук, заміну, вставку і видалення підрядків. Тим не менш, є класи задач з обробки символьної інформації, де стандартних можливостей не вистачає. Щоб полегшити вирішення подібних завдань, у мові **C#** вбудований більш потужний апарат роботи з рядками, заснований на регулярних виразах.

Регулярні вирази призначені для обробки текстової інформації і забезпечують:

1. Ефективний пошук в тексті за заданим шаблоном;
2. Редагування тексту;
3. Формування підсумкових звітів за результатами роботи з текстом.

Регулярний вираз - це шаблон, по якому виконується пошук відповідного фрагмента тексту. Мова опису регулярних виразів складається з символів двох видів: звичайних символів і метасимволів. Звичайний символ представляє у виразі сам себе, а метасимвол - деякий клас символів.

Розглянемо найбільш вживані метасимволи:

Клас символів	Опис
`.`	Будь-який символ, крім \n.
`[]`	Будь-який одинарний символ з послідовності, записаної всередині дужок. Допускається використання діапазонів символів.
`[^]`	Будь-який одинарний символ, який не входить в послідовність, записану в середині дужок. Допускається використання діапазонів символів.
`\w`	Будь-який алфавітно-цифровий символ.
`\W`	Будь-який не алфавітно-цифровий символ.
`\s`	Будь-який пробільний символ.
`\S`	Будь-який не пробільний символ.
`\d`	Будь-яка десяткова цифра
`\D`	Будь-який символ, який не є десяткового цифрою

Крім метасимволів, що позначають класи символів, можуть застосовуватися уточнюючі метасимволи:

Уточнюючі символи	Опис
`^`	Фрагмент,що співпадає з регулярними виразами, слід шукати лише на початку рядка
`$`	Фрагмент, що співпадає з регулярними виразами, слід шукати тільки

	в кінці рядка
\A	Фрагмент, що співпадає з регулярними виразами, слід шукати лише на початку багаторядкового рядка
\Z	Фрагмент,що співпадає з регулярними виразами, слід шукати тільки в кінці багаторядкового рядка
\b	Фрагмент, що співпадає з регулярними виразами, починається або закінчується на межі слова, тобто між символами, які відповідають мета символам **\w** і **\W**
\B	Фрагмент, що співпадає з регулярними виразами, не повинен зустрічатися намежі слів

У регулярних виразах часто використовуються повторювачі - метасимволи, які розташовуються безпосередньо після звичайного символу або групи символів і задають кількість його повторень у виразі.

Повторювач	Опис
*****	Нуль або більше повторень попереднього елемента
+	Одне або більше повторень попереднього елемента
?	Не більше одного повторення попереднього елемента
{n}	Рівно n повторень попереднього елемента
{n,}	Принаймні n повторень попереднього елемента
{n, m}	Від n до m повторень попереднього елемента

Регулярний вираз записується у вигляді рядкового літерала, причому перед рядком необхідно ставити символ **@**, який говорить про те, що рядок потрібно буде розглядати і в тому випадку, якщо він буде займати кілька рядків на екрані. Однак символ **@** можна не ставити, якщо як шаблон використовується шаблон без метасимволів.

Зауваження. Якщо потрібно знайти якийсь символ, який є метасимволом, наприклад, точка, можна це зробити поставивши перед ним зворотній слеш (**\.**).

Пошук в тексті за шаблоном

Простір імен бібліотеки базових класів **System.Text.RegularExpressions** містить всі об'єкти платформи **.NET Framework**, що мають відношення до регулярних виразів. Найважливішим класом, що підтримує регулярні вирази, є клас **Regex**. Для опису регулярного виразів у класі визначено кілька перевантажених конструкторів:

1. **Regex ()** - створює порожній вираз;
2. **Regex (String)** - створює заданий вираз;
3. **Regex (String, RegexOptions)** - створює заданий вираз і задає параметри для його обробки за допомогою елементів **RegexOptions** (наприклад, розрізняти чи ні великі та малі літери).

Пошук фрагментів рядків, які відповідають заданому виразу, виконується за допомогою методів **IsMatch**, **Match**, **Matches** класу **Regex**.

Метод **IsMatch** повертає **true**, якщо відповідний фрагмент в заданій рядку знайдено, і **false** у противному випадку.

Наприклад, необхідно визначити, чи зустрічається в заданому тексті слово собака:

```
static void Main ()
{
  Regex r = new Regex ("собака", RegexOptions.IgnoreCase);
  string text1 = "Кіт у будинку, собака в будці.";
  string text2 = "Котик в будинку, собачка в будці.";
  Console.WriteLine (r.IsMatch (text1));
  Console.WriteLine (r.IsMatch (text2));
}
```

Зауваження. **RegexOptions.IgnoreCase** - означає, що регулярний вираз застосовується без врахування регістра символів.

Визначення, чи є в заданих рядках номера телефону у форматі **xx-xx-xx** або **xxx-xx-xx** задається:

```
static void Main ()
{
  Regex r = new Regex (@ "\ d {2,3} (- \ d \ d) {2}");
  string text1 = "tel :123-45-67";
  string text2 = "tel: no";
  string text3 = "tel :12-34-56";
  Console.WriteLine (r.IsMatch (text1));
  Console.WriteLine (r.IsMatch (text2));
  Console.WriteLine (r.IsMatch (text3));
}
```

Метод **Match** класу **Regex** не просто визначає, чи міститься текст, відповідний шаблону, а повертає об'єкт класу **Match** - послідовність фрагментів тексту, які співпали з шаблоном. Наступний приклад дозволяє знайти всі номери телефонів в зазначеному фрагменті тексту:

```
static void Main ()
{
  Regex r = new Regex (@ "\ d {2,3} (- \ d \ d) {2}");
  string text = @ "Контакти в Києві tel :044-45-67-85, 044-45-34-56;
fax :123-56-45
                  Контакти в Чернівцях  tel :037-12-34-56; fax :
037-12-56-45 ";
  Match tel = r.Match (text);
  while (tel.Success)
  {
    Console.WriteLine (tel);
    tel = tel.NextMatch ();
  }
}
```

Наступний приклад дозволяє підрахувати суму цілих чисел, що зустрічаються в тексті:

```
static void Main ()
{
```

```
  Regex r = new Regex (@"[-+]? \ d + ");
  string text = @ "5 * 10 = 50 -80/40 =- 2";
  Match teg = r.Match (text);
  int sum = 0;
  while (teg.Success)
  {
    Console.WriteLine (teg);
    sum + = int.Parse (teg.ToString ());
    teg = teg.NextMatch ();
  }
  Console.WriteLine ("sum =" + sum);
}
```

Метод **Matches** класу **Regex** повертає об'єкт класу **MatchCollection** - колекцію всіх фрагментів заданого рядки, що збіглися з шаблоном. При цьому метод **Matches** багаторазово запускає метод **Match**, щоразу починаючи пошук із того місця, на якому закінчився попередній пошук.

```
  static void Main (string [] args)
  {
    string text = @ "5 * 10 = 50 -80/40 =- 2";
    Regex theReg = new Regex (@"[-+]? \ d + ");
    MatchCollection theMatches = theReg.Matches (text);
    foreach (Match theMatch in theMatches)
    {
      Console.Write ("{0}", theMatch.ToString ());
    }
    Console.WriteLine ();
  }
}
```

Редагування тексту

Регулярні вирази можуть ефективно використовуватися для редагування тексту. Наприклад, метод **Replace** класу **Regex** дозволяє виконувати заміну одного фрагмента тексту іншим або видалення фрагментів тексту.

Наприклад, зміна номерів телефонів:

```
static void Main (string [] args)
 {
 string text = @ "Контакти в Києві tel : 044-23-45-67, 044-23-34-56;
fax : 044-23-56-45.
Контакти в Чернівцях tel :037-12-34-56; fax : 037-11-56-45 ";
 Console.WriteLine ("Старі дані \ n" + text);
 string newText = Regex.Replace (text, "044 -", "037 -");
 Console.WriteLine ("Нові дані \ n" + newText);
 }
```

Видалення всіх номерів телефонів з тексту:

```
  static void Main (string [] args)
 {
 string text = @ "Контакти в Києві tel : 044-23-45-67, 044-23-34-56;
fax : 044-23-56-45.
Контакти в Чернівцях tel :037-12-34-56; fax : 037-11-56-45 ";
```

```
    Console.WriteLine ("Старі дані \ n" + text);
    string newText = Regex.Replace (text, @ "\ d {2,3} (- \ d \ d)
{2}", "");
    Console.WriteLine ("Нові дані \ n" + newText);
  }
}
```

Розбиття вихідного тексту на фрагменти:

```
static void Main (string [] args)
 {
 string text = @ "Контакти в Києві tel : 044-23-45-67, 044-23-34-56;
fax : 044-23-56-45.
Контакти в Чернівцях tel :037-12-34-56; fax : 037-11-56-45 ";
  string [] newText = Regex.Split (text, "[,.:;]+");
  foreach (string a in newText)
  Console.WriteLine (a);
}
```

Завдання до лабораторної роботи:

Порядок виконання роботи:

1) Написати C# програми згідно з варіантом завдання. При вирішенні завдання передбачити:
   - пошук ;
   - підрахунок кількості входжень;
   - пошук із заміною на нове значення.
2) Підготувати звіт у твердій копії та в електронному виді.

Дано рядок, в якому міститься осмислене текстове повідомлення. Слова повідомлення розділяються пробілами та розділовими знаками.

Завдання 1. Варіанти задач. Задано текст. Вивести всі підтести заданого формату, підрахувати їх кількість, вилучити та замінити деякі з них, за вказаними параметрами користувача.

1.1. У тексті може міститися дата в форматі **дд.мм.рррр**. Де **дд** - ціле число з діапазону від **1** до **31**, **мм** - ціле число з діапазону від **1** до **12**, а **рррр** - ціле число з діапазону від **1900** до **2099** (якщо якась частина формату порушена, то даний підрядок в якості дати не розглядається).

1.2. У тексті можуть міститися IP-адреси комп'ютерів в форматі **d.d.d.d**, де **d** - ціле число з діапазону від **0** до **255**.

1.3. У тексті можуть міститися IP-адреси комп'ютерів в форматі **d.d.d.d**, де **d** - ціле шіснацяткове число з діапазону від 0 до FF.

1.4. У тексті можуть міститися IP-адреси комп'ютерів в форматі **d.d.d.d**, де **d** - ціле двійкове число з діапазону від 0 до 11111111.

1.5. У тексті можуть міститися адреси web-сайтів домена com.

1.6. У тексті можуть міститися адреси web-сайтів домена edu.ua.
1.7. У тексті можуть міститися адреси web-сайтів домена cv.ua.
1.8. У тексті можуть міститися електронні адреси.
1.9. У тексті можуть міститися електронні адреси gmail.
1.10. У тексті можуть міститися електронні адреси ukrnet.
1.11. У тексті може міститися час у форматі **гг: хх: сс**. У заданому форматі **гг** - ціле число з діапазону від **00** до **24**, **хх** і **сс** - цілі числа з діапазону від **00** до **60** (якщо якась частина формату порушена, то дана підрядок в якості дати не розглядається).
1.12. У тексті може міститися дата- час в форматі **рррр.мм.дд:гг:хх**. Де **дд** - ціле число з діапазону від **1** до **31**, **мм** - ціле число з діапазону від **1** до **12**, а **рррр** - ціле число з діапазону від **1900** до **2099**, **гг** - ціле число з діапазону від **00** до **24**, а **хх** - ціле число з діапазону від **00** до **60** (якщо якась частина формату порушена, то даний підрядок в якості дати не розглядається).
1.13. У тексті може міститися географічні координати у форматі
1.14. У тексті може міститися координати вектора у форматі
1.15. У тексті може міститися координати вектора у форматі

Завдання 2. Варіанти задач. Задано текст.

2.1. Визначте, чи міститься у тексті задане слово.
2.2. Виведіть всі слова заданої довжини.
2.3. Видаліть всі однобуквені слова та слова які починаються буквами: 'a', 'b', 'c', 'd' і 'e' .
2.4. Видаліть всі знаки пунктуації та цифри.
2.5. Видаліть із повідомлення тільки ті українські слова, які починаються на голосну літеру.
2.6. Видаліть із повідомлення тільки ті українські слова, які починаються на голосну літеру
2.7. Замінити всі англійські слова на три крапки.
2.8. Знайти максимальне ціле число, що зустрічається в тексті.
2.9. Замінити всі шістнадцяткові цифри ('0', '1', ..., '9', 'a'-'f') на знак '+';
2.10. Замінити послідовність букв в алфавітному порядку на скорочений запис (наприклад: abcdf -> a-f);
2.11. Замінити скорочений запис на послідовність букв (наприклад: c-e -> cdfe, k-t -> klmnoprst);
2.12. Видаліть слова, які мають префікс "re", "not" та "be" та замінє слова, які мають префікс "не" на "not".
2.13. Видаліть слова, які мають закінчення "re", "nd" та "less" та замінє слова, які мають префікс "to" на "at".
2.14. Видаліть всі ідентифікатори, де ідентифікатор – будемо називати послідовність букв латиського алфавіту, символу

підкреслення та цифр, яка починається з букв латиського алфавіту або з символу підкреслення.

2.15. Видаліть всі не ідентифікатори де ідентифікатор – будемо називати послідовність букв латиського алфавіту, символу підкреслення та цифр, яка починається з букв латиського алфавіту або з символу підкреслення.

Завдання 3. Варіанти задач. Задано текст.

3.1. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає з цього тексту всі слова з подвоєнням літер і записує їх в окремий рядок, розділяючи пробілами. Друкує окремо вилучені слова і текст, що залишився після вилучення слів.

3.2. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає з цього тексту всі слова найбільшої довжини. (Слів найбільшої довжини може бути декілька). Друкує текст, що залишився після вилучення слів.

3.3. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка знаходить і друкує всі симетричні слова (наприклад, слово абввба є симетричним).

3.4. Задано два тексти, слова в яких розділені пробілами і розділовими знаками. Розробити програму, яка вилучає із першого тексту всі слова, що містяться у другому тексті.

3.5. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає в кожному слові цього тексту всі наступні входження першої літери.

3.6. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає в кожному слові цього тексту всі попередні входження останньої літери.

3.7. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає з цього тексту всі повторні входження слів.

3.8. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка знаходить і вилучає всі слова, що входять в цей текст по одному разу.

3.9. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка в словах непарної довжини цього тексту вилучає середню літеру.

3.10.Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка знаходить і друкує всі слова, що входять у заданий текст по одному разу.

3.11. Задано текст, слова в якому розділені пробілами і розділовими знаками, та два окремих слова. Розробити програму, яка замінює всі входження в заданий текст першого слова другим словом.

3.12. Задано два тексти, слова в яких розділені пробілами і розділовими знаками. Розробити програму, яка вилучає із другого тексту всі входження слів першого тексту.

3.13.Задано два тексти, слова в яких розділені пробілами і розділовими знаками. Розробити програму, яка створює третій текст із слів першого тексту, які не входять у другий текст, розділяючи їх пробілами.

3.14.Задано два тексти, слова в яких розділені пробілами і розділовими знаками, та окреме слово. Розробити програму, яка після кожного входження заданого слова в перший текст вставляє в нього другий текст.

3.15.Задано символ і текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка знаходить і друкує всі слова, що містять заданий символ найбільшу кількість разів.

3.16.Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка знаходить і друкує найдовший ланцюжок із слів однакової довжини.

3.17.Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає із заданого тексту всі слова непарної довжини.

3.18.Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка знаходить і друкує слово з найбільшою кількістю однакових символів (якщо таких слів декілька, то взяти перше з них).

3.19.Задано текст із малих латинських літер, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка знаходить і друкує всі слова з літерами, розміщеними в лексикографічному порядку.

3.20.Задано два тексти, слова в яких розділені пробілами і розділовими знаками. Розробити програму, яка створює третій текст із слів першого тексту, які входять у другий текст, і розділяє їх пробілами.

3.21. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає із заданого тексту всі попередні входження останнього слова.
3.22. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає із кожного слова заданого тексту всі повторні входження кожної літери.
3.23. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає з цього тексту всі слова з повторенням літер.
3.24. Задано текст із малих латинських букв, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка знаходить і друкує всі слова, букви в яких розміщені в алфавітному порядку.
3.25. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка знаходить і вилучає всі слова, літери в яких розміщені в лексикографічному порядку.
3.26. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає із кожного слова заданого тексту всі попередні входження останньої літери.
3.27. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка знаходить і друкує всі слова, буква ’A‘ або a’ в яких зустрічається найбільшу кількість разів.
3.28. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка знаходить і друкує всі слова, букви в яких не повторюються.
3.29. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає у кожному слові цього тексту всі повторні входження кожної букви.
3.30. Задано текст, слова в якому розділені пробілами і розділовими знаками. Розробити програму, яка вилучає із цього тексту всі слова, які починаються з голосної букви.

Контрольні питання