**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**імені ТАРАСА ШЕВЧЕНКА**

****

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Кафедра прикладних інформаційних систем**

**Звіт до практичної роботи №8**

**з курсу**

**«Об’єктно-орієнтоване програмування»**

*студента 2 курсу*

*групи ПП-22*

*спеціальності 122 «Комп'ютерні науки»*

*ОП «Прикладне програмування»*

Шевлюк Вікторії Віталіївни

*Викладач:*

к.ф.-м.н., доц. Шолохов О.В.

**Київ – 2022**

**Тема:** Використання колекцій для роботи з масивами у С#.

**Мета роботи:** Набуття навичок використання узагальнених колекцій з масивами.

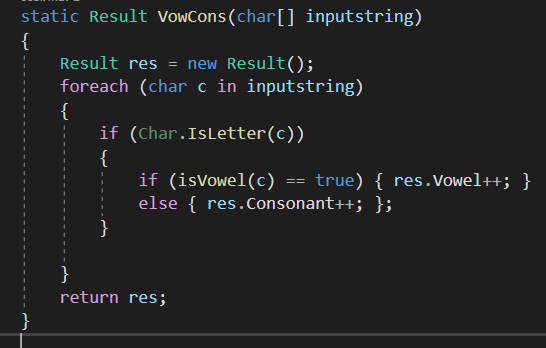
**Завдання**

Написати програму, яка обчислює кількість голосних та приголосних літер у файлі. Вміст текстового файлу заноситься до масиву символів. Кількість голосних та приголосних літер визначається проходом по масиву. Передбачити метод, вхідний параметр якого є масив символів. Метод обчислює кількість голосних та приголосних букв. Завдання також виконати за допомогою колекції List.

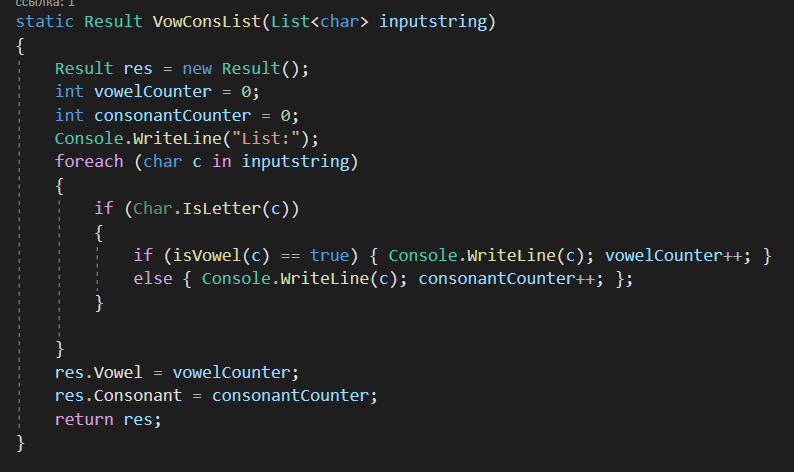
**Хід роботи:**

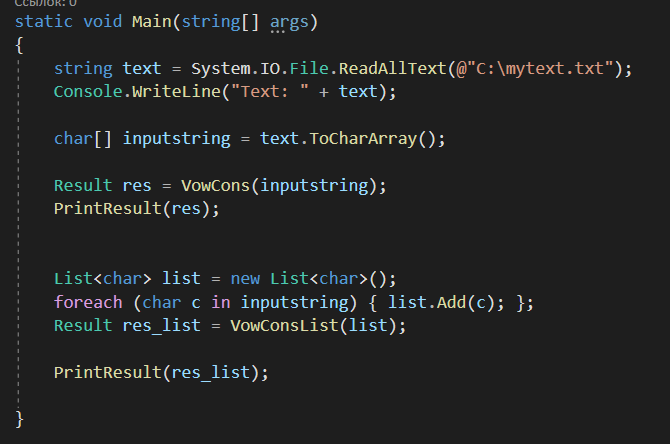
Реалізація даної програми полягає у наступному:

Програма зчитує текст з файлу, що розміщений на диску С у масив з символів, після чого ці дані передаватимуться у метод, що перевіряє, чи є зчитуваний елемент літерою, і якщо є – то перевіряє, чи голосна ця літера. В залежності від результату ведуться каунтери:

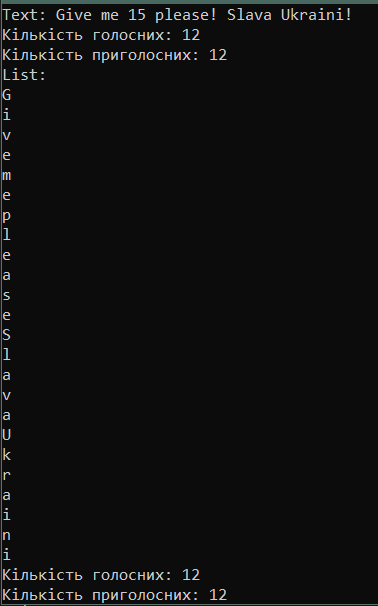


Далі ми створюємо список, де кожен символ (якщо він є літерою) – є елементом списку. У цьому списку також рахується кількість голосних та приголосних і виводиться на екран сам список і підрахунок літер.





Результат роботи програми:



**Висновок:** під час цієї лабораторної роботи я набула навичок використання узагальнених колекцій з масивами.

**Контрольні питання:**

***1. У чому відміна колекцій від масивів?***

Важлива особливість колекцій, що виділяє їх на тлі звичайних масивів - можливість додавати елементи до колекції та видаляти елементи з колекції. В роботі з колекціями процес «зміни» розміру масиву, що є основою колекції, автоматизований.

***2. Назвіть основні властивості та методи класу System.Array.***

Всі масиви C# побудовані на основі класу Array з простору імен System. Цей клас визначає ряд властивостей та методів, які ми можемо використовувати під час роботи з масивами. Основні властивості та методи:

► Length повертає довжину масиву

► Rank повертає розмірність масиву

► int BinarySearch (Array array, object? value) виконує бінарний пошук у відсортованому масиві та повертає індекс знайденого елемента

► Clear (Array array) очищає масив, встановлюючи для всіх його елементів значення за замовчуванням

► void Copy (Array sourceArray, int sourceIndex, Array destinationArray, int destinationIndex, int length) копіює з масиву sourceArray починаючи з індекс sourceIndex length елементів в масив destinationArray починаючи з індексу destinationIndex

► bool Exists<T> (T[] array, Predicate<T> match) перевіряє, чи містить масив array елементи, які відповідають умові делегата match

► void Fill<T> (T[] array, T value) заповнює масив array значенням value

► T? Find<T> (T[] array, Predicate<T> match) знаходить перший елемент, який відповідає певній умові з делегата match. Якщо елемент не знайдено, то повертається null

► T? FindLast<T> (T[] array, Predicate<T> match) знаходить останній елемент, який відповідає певній умові з делегата match. Якщо елемент не знайдено, то повертається null

► int FindIndex<T> (T[] array, Predicate<T> match) повертає індекс першого входження елемента, який задовольняє певну умову делегата match

► int FindLastIndex<T> (T[] array, Predicate<T> match) повертає індекс останнього входження елемента, який задовольняє певну умову

► T[] FindAll<T> (T[] array, Predicate<T> match) повертає всі елементи у вигляді масиву, які задовольняє певній умові з делегата match

► int IndexOf (Array array, object? value) повертає індекс першого входження елемента в масив

► int LastIndexOf (Array array, object? value) повертає індекс останнього входження елемента в масив

► void Resize<T> (ref T[]? array, int newSize) змінює розмір одновимірного масиву

► void Reverse (Array array) має елементи масиву у зворотному порядку

► Sort (Array array) сортує елементи одновимірного масиву

***3. Наведіть приклади опису масивів та колекцій.***

Масив: тип[] ім'я\_масиву = new тип[довжина\_масиву];

Список: List <тип> list = new List < тип >();

***4. Як передавати та повертати масиви та колекції з методів.***

*access return\_type* **MethodName**(*type*[] *parameterName*)

{

  // ...

}

Де *access* – модифікатор доступу (public, protected, private); *return\_type* – тип, що повертає метод; *MethodName* – назва методу, *type* – тип масиву, що передається в метод; *parameterName* – назва масиву, який є формальним параметром методу *MethodName*.

*access return\_type* **MethodName**(List <*type*> *parameterName*)

{

  // ...

}

Де *access* – модифікатор доступу (public, protected, private); *return\_type* – тип, що повертає метод; *MethodName* – назва методу; *type* – тип списку, що передається в метод; *parameterName* – назва списку, який є формальним параметром методу *MethodName*.

*5. Поясніть принцип роботи циклу foreach.*

Оператор циклу foreach призначений для перебору елементів колекції або масиву. Загальна форма оператора foreach наступна

foreach(type *identifier* in *container*)

{

  // оператори

  // ...

}

Де *type* – тип змінної з іменем *identifier*; *identifier* – ім’я змінної, яка використовується в якості ітератора. Змінна *identifier* набуває значення наступного елементу циклу на кожному кроці виконання циклу foreach. Тип змінної *identifier* повинен співпадати з типом масиву або колекції *container*. Зв’язок між ідентифікатор та контейнер реалізується з допомогою союзу in; *container* – ім’я колекії або масиву, який переглядається.

Оператор циклу foreach працює наступним чином. При входженні в цикл змінній *identifier* присвоюється перший елемент масиву (колекції) *container*. На кожному наступному кроці ітерації вибирається наступний елемент з *container*, який зберігається у змінній *identifier*. Цикл завершується, коли буде переглянуто усі елементи масиву (колекції) *container*.

***6. Розкажіть про плюси та мінуси використання двозв'язних списків.***

Перевага зв'язаного списку в тому, що операція вставки елемента всередину виконується дуже швидко. Це відбувається за рахунок того, що тільки посилання Next (наступний) попереднього елемента і Previous (попередній) наступного елемента повинні бути змінені так, щоб вказувати на елемент, що вставляється. У класі List при вставці нового елемента всі наступні мають бути зсунуті.

Звісно, зв'язані списки мають і свої недоліки. Так, наприклад, всі елементи таких списків доступні лише один за одним. Тому для знаходження елемента, що знаходиться в середині або наприкінці списку, потрібно досить багато часу. Зв'язаний список не може просто зберігати елементи у собі. Разом з кожним із них йому необхідно мати інформацію про наступний та попередній елементи.

***7. Наведіть практичні приклади ефективного використання розглянутих колекцій***

Масиви є зручним інструментом групування даних. Проте, масиви зберігають фіксовану кількість об'єктів, інколи ж заздалегідь невідомо, скільки потрібно об'єктів. І в цьому випадку набагато зручніше використовувати колекції.

Щодо продуктивності розглянутих тут колекцій зауважимо, що додавання нового об'єкта (Add) швидше робить List, повільніше − Dictionary. На пошук елемента йде приблизно однаковий час, проте пошук по ключу істотно швидше Dictionary. Видалення об'єкта повільніше робиться у класі List.