**Коллекции и классы-прототипы**

**Задание №1.** Решить задачи с использованием класса Stack. Пусть символ # определен в текстовом редакторе как стирающий символ Backspace, т.е. строка abc#d##c в действительности является строкой ac. Дантекст, в котором встречается символ #. Преобразовать его с учетом действия этого символа.

Листинг программы:

static void Main(string[] args)

{

Write("Введите строку, элемент '#' = backspace: ");

var myString = ReadLine();

StackManager stackManager = new StackManager(myString);

var myStack = stackManager.Delete();

Write($"Исходная строка: {myString}\n");

Write("Выходная строка:"); stackManager.WriteStack(myStack);

ReadKey();

}

Анализ результатов:

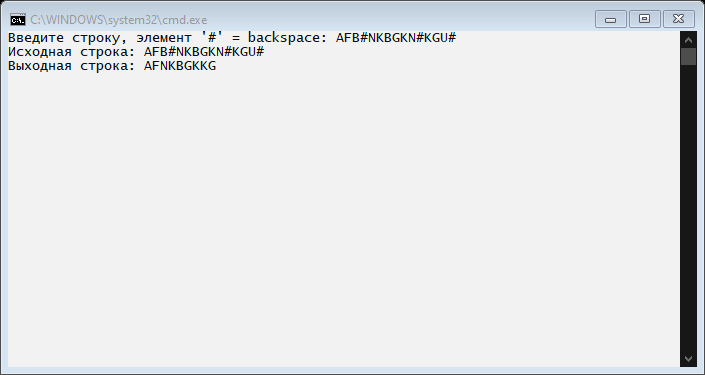


Рисунок 1.1 – Результат работы программы

**Задание № 2.** Дан текстовый файл. За один просмотр файла напечатать элементы файла в следующем порядке: сначала все символы, отличные от цифр, а затем все цифры, сохраняя исходный порядок в каждой группе символов.

Листинг программы:

static void Main(string[] args)

{

string textPath = @"E:\ПРАКТИКА ПО ПРОГРАММИРОВАНИЮ\Практика 18\Практика 18\Задание 2\bin\Text.txt";

StreamReader streamReader = new StreamReader(textPath);

QueueManager queueManager = new();

var readed = default(string);

while ((readed = streamReader.ReadLine()) != null)

{

foreach (var item in readed)

{

if (char.IsDigit(item))

{

queueManager.AddToDigitQueue(item);

}

else

{

queueManager.AddToNoDigitQueue(item);

}

}

}

queueManager.GetQueueNoDigits()?.ToList().ForEach(i => Write($"{i}"));

WriteLine();

queueManager.GetQueueDigits()?.ToList().ForEach(i => Write($"{i} "));

ReadLine();

}

Анализ результатов:

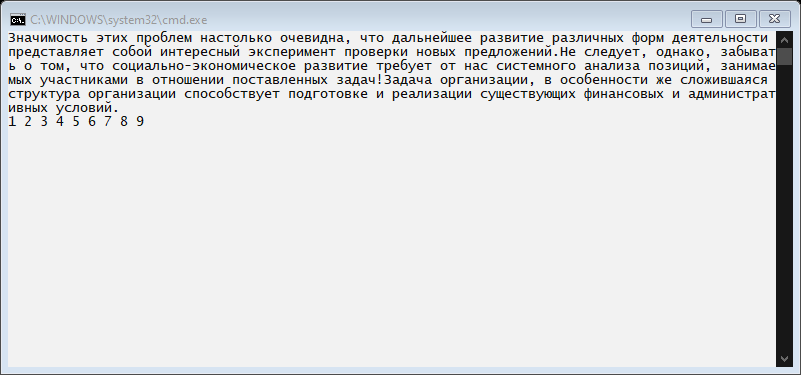


Рисунок 1.2 – Результат работы программы

**Задание № 3.** Дан текстовый файл. За один просмотр файла напечатать элементы файла в следующем порядке: сначала все символы, отличные от цифр, а затем все цифры, сохраняя исходный порядок в каждой группе символов. Тема: Животные.

Листинг программы:

static void Main(string[] args)

{

List<Animal> animals = new();

Random random = new Random();

var countOfAnimals = random.Next(3, 10);

string type;

int age;

string? name;

for (int i = 0; i < countOfAnimals; i++)

{

var whatCreate = random.Next(0, 2);

var ifName = random.Next(0, 2);

age = random.Next(1, 20);

if (whatCreate == 0)

{

type = Faker.EnumFaker.SelectFrom<EnumForModels.TypeOfBird>().ToString();

double wingspan = random.NextDouble();

int beakLength = random.Next(3, 10);

if (ifName == 0)

{

name = Faker.Name.First();

}

else

{

name = null;

}

animals.Add(new Bird(wingspan, beakLength, type, name, age));

}

else

{

type = Faker.EnumFaker.SelectFrom<EnumForModels.HorseType>().ToString(); // breed

var weight = random.Next(100, 300);

var color = Faker.EnumFaker.SelectFrom<EnumForModels.Color>().ToString();

if (ifName == 0)

{

name = Faker.Name.First();

}

else

{

name = null;

}

animals.Add(new Horse(weight,color, type, name, age));

}

}

foreach (var animal in animals)

{

WriteLine(animal.Information());

}

ReadKey();

}

Анализ результатов:

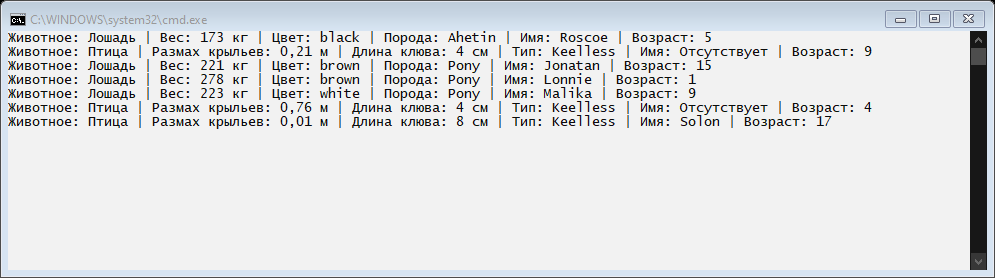


Рисунок 1.3 – Результат работы программы