**ПРАКТИЧЕСКОЕ ЗАДАНИЕ 18**

**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. Дан текст, в котором встречается символ #. Преобразовать его с учетом действия этого символа.

Листинг программы

try

{

Console.WriteLine("Введите строку: ");

string str = Console.ReadLine();

var textStack = new Stack<char>();

for (int i = 0; i < str.Length; i++)

textStack.Push(str[i]);

Console.WriteLine(EraseCharacters(textStack));

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

string EraseCharacters(Stack<char> text)

{

int counter = 0;

string str = string.Empty;

while (text.Count > 0)

{

char c = text.Pop();

if (c == '#')

counter++;

else if (counter > 0)

counter--;

else str += c;

}

char[] chars = new char[str.Length];

chars = str.Select(x => x).ToArray();

Array.Reverse(chars);

str = new string(chars);

return str;

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| abc#d##c | ac |

Анализ результатов:

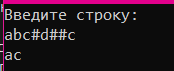


Рисунок 1.1 – Результат работы программы

Задание 2. Решить следующие задачи с использованием класса Queue. 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)

{

Queue<char> symbols = new Queue<char>();

Queue<char> digits = new Queue<char>();

using (StreamReader sr = new StreamReader(@"C:\Users\Полина\Desktop\Кпияп\Практика\practice18\job2\job2.txt"))

{

while (!sr.EndOfStream)

{

char s = (char)sr.Read();

if (char.IsDigit(s)) digits.Enqueue(s);

else symbols.Enqueue(s);

}

}

Console.WriteLine(string.Join("", symbols));

Console.WriteLine(string.Join("", digits));

Console.ReadKey();

}

Таблица 1.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 1 2 3 Полина Никита Инна Настя 45 7 9 Стандарт 4 | Полина Никита Инна Настя Стандарт  12345794 |

Анализ результатов:



Рисунок 1.2 – Результат работы программы

Задание 3. Создать класс-прототип, с обобщенным методом, с методами добавления, удаления, клонирования экземпляров класса.

В основной программе продемонстрировать:

функциональность созданного класса и всех его методов;

использование обнуляемых типов,

использование абстрактных структур (List&lt;P&gt; Dictionary&lt;Key,Value&gt;). Урожай

Листинг программы:

namespace RefactoringGuru.DesignPatterns.Prototype.Conceptual

{

public class Harvest

{

public int? Age = null;

public string? Title = null;

public IdInfo? IdInfo = null;

public Harvest ShallowCopy()

{

return (Harvest)this.MemberwiseClone();

}

public Harvest DeepCopy()

{

Harvest clone = (Harvest)this.MemberwiseClone();

clone.IdInfo = new IdInfo(IdInfo.IdNumber);

clone.Title = Title;

return clone;

}

}

public class IdInfo

{

public int IdNumber;

public IdInfo(int idNumber)

{

this.IdNumber = idNumber;

}

}

class Program

{

static void Main(string[] args)

{

Harvest p1 = new Harvest();

p1.Age = 1;

p1.Title = "Wheat";

p1.IdInfo = new IdInfo(1);

Harvest p2 = p1.ShallowCopy();

Harvest p3 = p1.DeepCopy();

Console.WriteLine("Original values of p1, p2, p3:");

Console.WriteLine(" p1 instance values: ");

DisplayValues(p1);

Console.WriteLine(" p2 instance values:");

DisplayValues(p2);

Console.WriteLine(" p3 instance values:");

DisplayValues(p3);

p1.Age = 2;

p1.Title = "Corn";

p1.IdInfo.IdNumber = 2;

Console.WriteLine("\nValues of p1, p2 and p3 after changes to p1:");

Console.WriteLine(" p1 instance values: ");

DisplayValues(p1);

Console.WriteLine(" p2 instance values (reference values have changed):");

DisplayValues(p2);

Console.WriteLine(" p3 instance values (everything was kept the same):");

DisplayValues(p3);

var animals = new Dictionary<int, Harvest>();

animals[p1.IdInfo.IdNumber] = p1;

animals[p2.IdInfo.IdNumber] = p2;

animals[p3.IdInfo.IdNumber] = p3;

}

public static void DisplayValues(Harvest p)

{

Console.WriteLine(" Title: {0:s}, Age: {1:d}",

p.Title, p.Age);

Console.WriteLine(" ID#: {0:d}", p.IdInfo.IdNumber);

}

}

}

Таблица 1.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | Original values of p1, p2, p3:  p1 instance values:  Title: Wheat, Age: 1  ID#: 1  p2 instance values:  Title: Wheat, Age: 1  ID#: 1  p3 instance values:  Title: Wheat, Age: 1  ID#: 1  Values of p1, p2 and p3 after changes to p1:  p1 instance values:  Title: Corn, Age: 2  ID#: 2  p2 instance values (reference values have changed):  Title: Wheat, Age: 1  ID#: 2  p3 instance values (everything was kept the same):  Title: Wheat, Age: 1  ID#: 1 |

Анализ результатов:

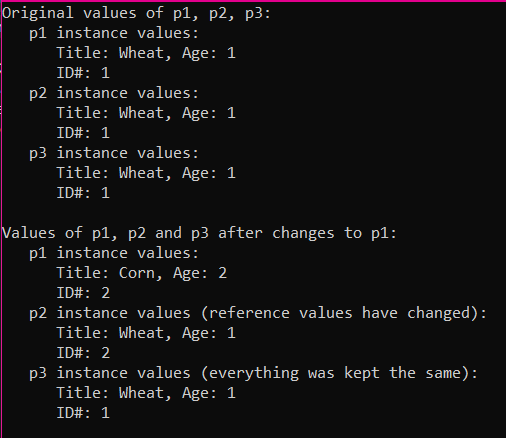


Рисунок 1.3 – Результат работы программы

Задание 4. Решить задачу с использованием класса HashTable. Реализовать простейший каталог музыкальных компакт-дисков, который позволяет: добавлять и удалять диски, добавлять и удалять песни, просматривать содержимое целого каталога и каждого диска в отдельности.

Листинг программы:

Dictionary<string, Dictionary<string, string>> catalog = new Dictionary<string, Dictionary<string, string>>();

int number = 9;

while (number != 0)

{

try

{

Console.Write("1)Создание диска\n2)Удаление диска\n3)Добавление записи\n4)Удаление записи\n5)Просмотр каталога\n6)Просмотр определенного диска\nВведите '0' для завершения программы\n");

number = int.Parse(Console.ReadLine());

if (number == 1)

{

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

Dictionary<string, string> dict = new Dictionary<string, string>();

Console.Write("Введите ключ диска: ");

string keyOfDisk = Console.ReadLine();

catalog.Add(keyOfDisk, dict);

Console.WriteLine("Диск создан");

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

}

else if (number == 2)

{

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

Console.Write("Введите ключ диска, который Вы хотите удалить: ");

string keyOfDiskDDelete = Console.ReadLine();

catalog.Remove(keyOfDiskDDelete);

Console.WriteLine("Диск удален");

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

}

else if (number == 3)

{

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

Console.Write("Введите ключ диска, в который Вы хотите добавить запись: ");

string keyOfDisk = Console.ReadLine();

var dict1 = AddMusic(keyOfDisk);

}

else if (number == 4)

{

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

Console.Write("Введите ключ диска, с которого хотите удалить запись запись: ");

string keyOfDisk = Console.ReadLine();

var dict1 = RemoveMusic(keyOfDisk);

}

else if (number == 5)

{

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

foreach (var cat in catalog)

{

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

Console.WriteLine(cat.Key);

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

foreach (var catI in catalog[cat.Key])

{

Console.Write($"ключ: {catI.Key} ");

Console.WriteLine($"название: {catI.Value}" + "\n");

}

}

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

}

else if (number == 6)

{

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

Console.Write("Введите ключ диска, который Вы хотите просмотреть: ");

string keyOfDisk = Console.ReadLine();

CheckDisk(keyOfDisk);

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

Dictionary<string, string> AddMusic(string keyOfDisk)

{

var dict1 = catalog[keyOfDisk];

Console.Write("Ключ песни: ");

string keyOfMusic = Console.ReadLine();

Console.Write("Название песни: ");

string valOfMusic = Console.ReadLine();

dict1.Add(keyOfMusic, valOfMusic);

Console.WriteLine("Песня добавлена");

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

return dict1;

}

Dictionary<string, string> RemoveMusic(string keyOfDisk)

{

var dict1 = catalog[keyOfDisk];

Console.Write("Ключ песни: ");

string keyOfMusic = Console.ReadLine();

dict1.Remove(keyOfMusic);

Console.WriteLine("Песня удалена");

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

return dict1;

}

void CheckDisk(string keyOfDisk)

{

var dict1 = catalog[keyOfDisk];

foreach (var mus in dict1)

{

Console.WriteLine($"ключ: {mus.Key}");

Console.WriteLine($"название песни: {mus.Value} \n");

}

Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

}

Анализ результатов:

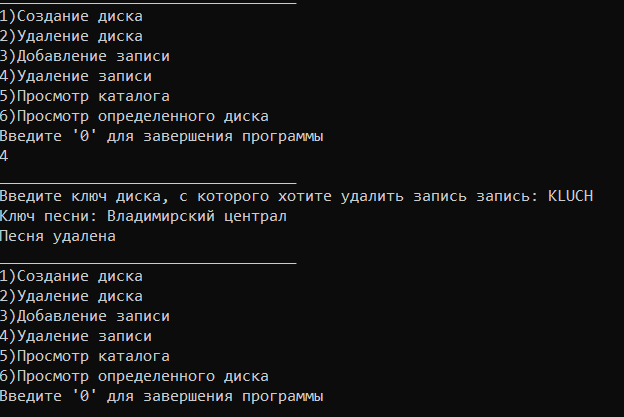


Рисунок 1.4 – Результат работы программы