

Содержание

[Введение 3](#_Toc166889382)

[Разработка кода 5](#_Toc166889383)

[Результаты 11](#_Toc166889384)

[Заключение 12](#_Toc166889385)

[Список использованной литературы 13](#_Toc166889386)

# Введение

В ходе выполнения данной работы была разработана программа, реализующая интерактивное меню на языке C#. Меню позволяет пользователю выбрать между информацией об авторе, реализацией польской записи и скобок. Кроме того, программа включает в себя пример работы со стеком, демонстрируя способность языка C# работать с данными структурами.

Программа написана на языке программирования C#, она представляет собой пример навыков работы с консольными приложениями, циклами и методами в C#, а также предоставляет пример реализации интерактивного меню. Стек в программе используется для хранения и обработки данных, что позволяет эффективно управлять информацией и выполнять различные операции.

Стек (Stack) – абстрактная структура данных, которая позволяет хранить и извлекать элементы в порядке последнего вхождения (LIFO – Last In, First Out). Стек может быть реализован с помощью массива или связанного списка. Основные операции со стеком: push (добавление элемента), pop (удаление элемента), peek (просмотр верхнего элемента).

Постфиксная (или обратная польская) запись арифметического выражения – это способ записи выражений, в котором знак операции записывается не между операндами, а после операндов.

Числа добавляются в стек, а результат операций применяется к двум последним числам в стеке, которые из стека удаляются. Затем результат кладется в стек.

То есть выражение «2 3 + 4 \*» в постфиксной записи может быть вычислено так:

* Число 2 кладется в стек.
* Число 3 кладется в стек.
* Из стека извлекаются числа 3 и 2, к ним применяется операция сложения, результат (число 5) кладется в стек.
* Число 4 кладется в стек.
* Из стека удаляются числа 5 и 4, результат их умножения 20 кладется в стек.

В итоге в стеке оказывается одно число – 20, которое и есть результат вычисления выражения.

Польская запись широко используется в компиляторах, интерпретаторах и калькуляторах, так как она позволяет эффективно обрабатывать математические выражения и логические операции.

# Разработка кода

Этот код представляет собой интерактивное меню, которое позволяет пользователю выбрать один из четырех пунктов:

1. Информация об авторе
2. Польская запись
3. Скобки
4. Выход из меню

При выборе первого пункта меню, пользователь видит информацию об авторе программы.

При выборе второго пункта меню, пользователь может ввести выражение в обратной польской нотации, которое будет вычислено с помощью стека. В этом случае используется класс Stack<T> из пространства имен System.Collections.Generic. Сначала пользователь вводит выражение, которое разбивается на отдельные элементы с помощью метода Split. Затем каждый элемент проверяется на то, является ли он числом или операцией. Если элемент является числом, то он добавляется в стек. Если элемент является операцией, то из стека извлекаются два операнда, выполняется операция и результат добавляется в стек. В конце процесса стек должен содержать только один элемент – результат выражения.

При выборе третьего пункта меню, пользователь может ввести строку со скобками, которая будет проверена на правильность расстановки скобок. В этом случае также используется класс Stack<T>. Сначала пользователь вводит строку со скобками, которая разбивается на отдельные символы. Затем каждый символ проверяется на то, является ли он открывающей или закрывающей скобкой. Если символ является открывающей скобкой, то он добавляется в стек. Если символ является закрывающей скобкой, то из стека извлекается открывающая скобка и проверяется, соответствует ли она закрывающей скобке. Если соответствие найдено, то открывающая скобка удаляется из стека. Если соответствие не найдено, то закрывающая скобка добавляется в стек. В конце процесса стек должен быть пустым, если все скобки расставлены правильно.

При выборе четвертого пункта меню, программа завершает свою работу.

В целом, этот код демонстрирует использование стека в C# для решения различных задач, таких как вычисление выражений в обратной польской нотации и проверка правильности расстановки скобок.

using System;

class InteractiveMenu

{

static void Main(string[] args)

{

while (true)

{

Console.Clear();

Console.WriteLine("Меню:");

Console.WriteLine("1. Информация об авторе");

Console.WriteLine("2. Польская запись");

Console.WriteLine("3. Скобки");

Console.WriteLine("4. Выход из меню");

Console.Write("Выберите пункт меню: ");

int choice = Convert.ToInt32(Console.ReadLine());

switch (choice)

{

case 1:

ShowAuthorInfo();

break;

case 2:

PolishNotation();

break;

case 3:

Brackets();

break;

case 4:

Environment.Exit(0);

break;

default:

Console.WriteLine("Неверный выбор. Пожалуйста, выберите снова.");

Console.ReadKey();

break;

}

}

}

static void ShowAuthorInfo()

{

Console.Clear();

Console.WriteLine("Информация об авторе:");

Console.WriteLine("Автор: Колодницкий Илья");

Console.WriteLine("Группа: ФИТ-232");

Console.WriteLine("Нажмите любую клавишу, чтобы вернуться назад...");

Console.ReadKey();

Main(null);

}

static void PolishNotation()

{

Console.Clear();

Console.WriteLine("Польская запись:");

Console.WriteLine("Введите выражение в обратной польской нотации:");

string str = Console.ReadLine();

var notation = str.Split();

string[] operations = new string[] { "+", "-", "\*", "/" };

Stack<double> stack = new Stack<double>();

bool valid = true;

foreach (string elem in notation)

{

bool isDouble = double.TryParse(elem, out double number);

if (isDouble)

{

stack.Push(number);

}

else if (Array.IndexOf(operations, elem) != -1)

{

if (stack.Count < 2)

{

valid = false;

break;

}

double n1 = stack.Pop();

double n2 = stack.Pop();

if (elem == "+")

{

stack.Push(n2 + n1);

}

else if (elem == "-")

{

stack.Push(n2 - n1);

}

else if (elem == "\*")

{

stack.Push(n2 \* n1);

}

else if (elem == "/")

{

if (n1 == 0)

{

throw new DivideByZeroException("Нельзя делить на ноль!");

}

else

{

stack.Push(n2 / n1);

}

}

}

}

if (stack.Count != 1)

{

valid = false;

}

if (valid)

{

Console.WriteLine("Запись верна!");

Console.WriteLine($"Результат: {stack.Peek()}");

}

else

{

Console.WriteLine("Запись НЕВЕРНА!");

}

Console.WriteLine("Нажмите любую клавишу, чтобы вернуться назад...");

Console.ReadKey();

Main(null);

}

static void Brackets()

{

Console.Clear();

Console.WriteLine("Скобки:");

Console.WriteLine("Введите строку со скобками:");

string str = Console.ReadLine();

char[] parens = new char[] { '(', ')', '[', ']', '{', '}' };

Stack<char> stack = new Stack<char>();

foreach (char c in str)

{

int index = Array.IndexOf(parens, c);

if (c == '(' || c == '[' || c == '{')

{

stack.Push(c);

}

else if (index != -1)

{

if (stack.Count > 0 && stack.Peek() == parens[index - 1])

{

stack.Pop();

}

else

{

stack.Push(c);

}

}

}

if (stack.Count > 0)

{

Console.WriteLine("Скобки расставлены НЕВЕРНО!");

}

else

{

Console.WriteLine("Все верно!");

}

Console.WriteLine("Нажмите любую клавишу, чтобы вернуться назад...");

Console.ReadKey();

Main(null);

}

}

# Результаты

Результат работы программы, в котором показано меню, представлен на рисунке 1.

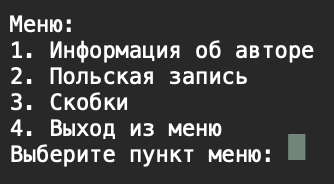


Рисунок 1 – Результат программы

На рисунке 2 представлена информация об авторе.

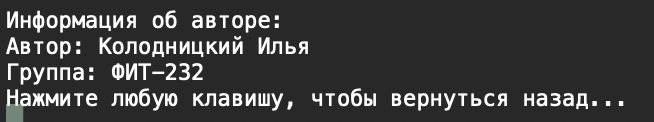


Рисунок 2 – Результат программы

На рисунке 3 представлен результат работы программы польской записи.

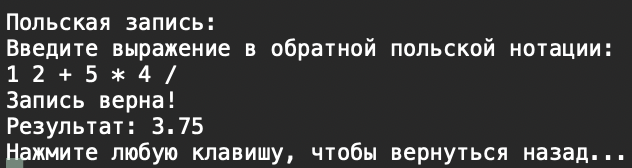


Рисунок 3 – Результат программы

На рисунке 4 представлен результат работы программы скобки.

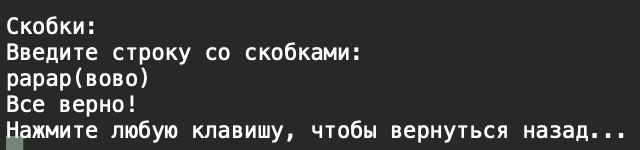
****

Рисунок 4 – Результат программы

# Заключение

В ходе выполнения данной расчетно-графической работы была разработана программа, реализующая интерактивное меню для выполнения различных задач. Программа содержит четыре пункта меню:

1. Информация об авторе
2. Польская запись
3. Скобки
4. Выход из меню

При выборе пункта 1, выводится информация об авторе программы.

При выборе пункта 2, пользователь может ввести выражение в обратной польской нотации, и программа вычислит его значение.

При выборе пункта 3, пользователь может ввести строку со скобками, и программа проверит, правильно ли они расставлены.

При выборе пункта 4, программа завершает свою работу.

Вышеприведенный код на языке C# реализует данную программу. Он содержит класс InteractiveMenu, в котором определены методы Main, ShowAuthorInfo, PolishNotation и Brackets. Метод Main выводит меню и обрабатывает выбор пользователя, вызывая соответствующий метод. Метод ShowAuthorInfo выводит информацию об авторе программы. Метод PolishNotation вычисляет значение выражения в обратной польской нотации. Метод Brackets проверяет правильность расстановки скобок в строке.

# Список использованной литературы

# 1) Полное руководство по языку программирования С# 12 и платформе .NET 8 [электронный ресурс]: Режим доступа: <https://metanit.com/sharp/tutorial/>. ­­– (дата обращения 17.05.24)

# 2) Справочник по языку C# [электронный ресурс]: Режим доступа:  <https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/>. – (дата обращения 17.05.24)

# 3) Работа со структурами в C# [электронный ресурс]: Режим доступа: <https://habr.com/ru/articles/114953/>. – (дата обращения 17.05.24)