Министерство науки и образования РФ

Федеральное государственное бюджетное учреждение

высшего образования

**«Тверской государственный технический университет»**

(ТвГТУ)

Кафедра программного обеспечения

**Курсовая работа**

по дисциплине: «Теория алгоритмов»

Тема: «Машина Поста (работа с серией программ на входе)»

|  |
| --- |
| Выполнил:  студент группы  Б.ПИН.РИС-21.06  Сорокин Евгений |
| Проверила:  старший преподаватель кафедры ПО  Корнеева Е.И. |
|  |

Тверь 2023

**Содержание**

[**Введение** 3](#_Toc153355830)

[**Цели и задачи проекта:** 3](#_Toc153355831)

[**Теоретическая часть** 4](#_Toc153355832)

[**Теоретическая справка:** 4](#_Toc153355833)

[**Основные теоремы машины Поста включают:** 4](#_Toc153355834)

[**Сравнение с машиной Тьюринга**: 5](#_Toc153355835)

[**Алгоритм Машины Поста:** 5](#_Toc153355836)

[**Проектная часть** 6](#_Toc153355837)

[**Техническое задание:** 6](#_Toc153355838)

[**Цель проекта:** 6](#_Toc153355839)

[**Ключевые требования:** 6](#_Toc153355840)

[**Задачи проекта:** 6](#_Toc153355841)

[**Роли в проекте:** 6](#_Toc153355842)

[**Схема работы программы**: 7](#_Toc153355843)

[**GUI Lead** 8](#_Toc153355844)

[**Скриншот интерфейса программы:** 8](#_Toc153355845)

[**Функционал приложения:** 8](#_Toc153355846)

[**Заключение** 11](#_Toc153355847)

[**Выводы по работе:** 11](#_Toc153355848)

[**Выводы по своей роли:** 11](#_Toc153355849)

[**Список литературы** 12](#_Toc153355850)

# **Введение**

## **Цели и задачи проекта:**

Целью данного проекта является исследование и анализ машины Поста, а также определение областей ее применения в работе с серией программ на входе. Для достижения этой цели мы ставим следующие задачи:

* Изучение основных теорем машины Поста, включая теорему о существовании универсальной машины Поста.
* Исследование конкретных задач, которые можно решать с помощью машины Поста, такие как проверка на палиндром, обработка формальных языков и другие.
* Сравнение машины Поста с машиной Тьюринга с точки зрения вычислительной мощности, выразительности и применимости.

# **Теоретическая часть**

## **Теоретическая справка:**

Машина Поста состоит из каретки (или считывающей и записывающей головки) и разбитой на ячейки бесконечной в обе стороны ленты. Каждая ячейка ленты может находиться в 2 состояниях — быть либо пустой — 0, либо помеченной меткой 1. За такт работы машины каретка может сдвинуться на одну позицию влево или вправо, считать, изменить символ в своей текущей позиции.

Работа машины Поста определяется программой, состоящей из конечного числа строк. Для работы машины нужно задать программу и её начальное состояние (то есть состояние ленты и позицию каретки). Кареткой управляет программа, состоящая из пронумерованных не обязательно упорядоченных строк команд, если в каждой команде указана строка, на которую нужно перейти. Обычно принимается, что если в команде переход не указан, то переход происходит на следующую строку.

## **Основные теоремы машины Поста включают:**

* Теорема о существовании универсальной машины Поста: Существует машина Поста, которая может симулировать работу любой другой машины Поста, принимая ее описание и входные данные.
* Теорема о остановке: Невозможно написать программу на машине Поста, которая может точно определить, остановится ли данная машина Поста при выполнении данной программы.

Задачи, которые можно решать с помощью машины Поста, включают в себя проверку на палиндром, преобразование одного формального языка в другой, решение задачи о непустоте, и многие другие.

## **Сравнение с машиной Тьюринга**:

Машина Поста и машина Тьюринга обе являются моделями вычислений, но они имеют свои преимущества и недостатки:

***Преимущества машины Поста:***

* Она более интуитивно понятна и легче визуализируется, особенно при работе с последовательностями символов.
* Машина Поста более наглядно демонстрирует процессы, связанные с обработкой строковых данных.

***Недостатки машины Поста:***

* Она менее универсальна и ограничена в выразительности по сравнению с машиной Тьюринга.
* Невозможно определить остановку машины Поста для произвольной программы.

## **Алгоритм Машины Поста:**

1. Задать начальное состояние ленты и позицию каретки.

2. Задать программу, состоящую из конечного числа строк команд.

3. Установить текущую команду на первую строку программы.

4. Если текущая команда - ".", то остановить работу машины.

5. Если текущая команда - "V j", то поставить метку в текущей позиции и перейти к j-й строке программы.

6. Если текущая команда - "X j", то стереть метку в текущей позиции и перейти к j-й строке программы.

7. Если текущая команда - "← j", то сдвинуться влево на одну позицию и перейти к j-й строке программы.

8. Если текущая команда - "→ j", то сдвинуться вправо на одну позицию и перейти к j-й строке программы.

9. Если текущая команда - "? j1; j2", то если в текущей позиции нет метки, то перейти к j2-й строке программы, иначе перейти к j1-й строке программы.

10. Установить текущую команду на следующую строку программы.

11. Повторять шаги 4-10, пока не будет выполнена команда "!" или не будет достигнут максимальный лимит шагов.

# **Проектная часть**

## **Техническое задание:**

### **Цель проекта:**

Разработать программу для работы с серией программ на входе, используя Машину Поста.

### **Ключевые требования:**

- Разработка программы должна быть выполнена на языке Python;

- Программа должна иметь графический интерфейс пользователя (GUI);

- Программа должна содержать модульные тесты;

- Программа должна иметь

### **Задачи проекта:**

* Создать конечный автомат (машина Поста), который будет работать с введенной строкой и определять, является ли она палиндромом. Машина должна иметь состояния для чтения символов, перемещения влево и вправо по строке и принятия решения.
* Разработать интерфейс пользователя для ввода строки.
* Реализовать алгоритм работы машины Поста в программном коде.
* Провести тестирование программы на различных входных данных, включая палиндромы и не палиндромы, и убедиться в правильности определения.

### **Роли в проекте:**

1. Чесноков Дмитрий Сергеевич. Team Lead - Отвечает за общее руководство проектом, планирование и контроль выполнения работ, координацию команды.

2. Сорокин Евгений. Classes lead - Отвечает за разработку классов и методов для работы Машины Поста.

3. Леонов Алексей. GUI lead - Отвечает за разработку графического интерфейса пользователя.

4. Талакуцкая Анастасия. Test lead - Отвечает за разработку и проведение модульных тестов.

5. Лютый Максим. Docs lead - Отвечает за разработку документации.

## **Схема работы программы**:C:\Users\dcdim\AppData\Local\Microsoft\Windows\INetCache\Content.Word\diagram (2).png

Рис.1 Схема работы программы

## **Class Lead**

Данный код представляет собой два класса. Класс Tape представляет работу с лентой (tape) машины Поста, а класс PostMachine - саму машину Поста.

Методы класса **Tape:**

- init(self): инициализация ленты

- right(self): перемещение каретки вправо

- left(self): перемещение каретки влево

- num(self, a): запись числа в ячейку

- checknum(self): чтение числа из ячейки

Методы класса **PostMachine:**

- init(self): инициализация программы машины Поста

- add\_command(self, command): добавление команды в программу машины

- run(self): выполнение программы машины

- get\_tape(self): вывод всей ленты

Класс PostMachine описывает работу машины Поста, которая состоит из команд перемещения по ленте (вправо/влево), записи чисел в ячейку, проверки числа и завершения программы.

class PostMachine:

    def \_\_init\_\_(self):

        self.program = []  # Программа машины Поста

        self.current\_line = 0  # Текущая строка программы

        self.tape = Tape()

    def add\_command(self, command): #добавляем команду в машину

        self.program.append(command)

    def run(self):

        while self.current\_line < len(self.program): #пока не достигли конца программы

            command = self.program[self.current\_line]

            cmd, \*args = command.split()

            if cmd == ">":

                self.tape.right()

            elif cmd == "<":

                self.tape.left()

            elif cmd == "0":

                self.tape.num(0)

            elif cmd == "1":

                self.tape.num(1)

            elif cmd == "?":

                if self.tape.checknum != 0:

                    self.current\_line = int(args[1]) #- 1

                else:

                    self.current\_line = int(args[0]) #- 1

            elif cmd == ".":

                break

            if cmd != "?":

                if args is not None:

                    self.current\_line = int(args[0])

                else:

                    self.current\_line += 1

    def get\_tape(self): #вывод всей ленты

        t = list(reversed(self.tape.left\_tape))

        return "".join(map(str, t)) + "".join(map(str, self.tape.right\_tape))

class Tape:

    def \_\_init\_\_(self):

        self.left\_tape = [] # Левая часть ленты(она будет начинаться с -1)

        self.right\_tape = [0]  # Правая часть ленты(а она с 0)

        self.pos\_carriage = 0 #позиция каретки

    def right(self):

        self.pos\_carriage += 1

        if self.pos\_carriage > 0:

            if self.pos\_carriage + 1 > len(self.right\_tape): # если ленты не хватает добавляем ячейку

                self.right\_tape.append(0)

    def left(self):

        self.pos\_carriage -= 1

        if self.pos\_carriage == -1:

            if len(self.left\_tape) == 0: # добавляем элемент так как изначальна список пуст

                self.left\_tape.append(0)

        if self.pos\_carriage < -1:

            if abs(self.pos\_carriage) >= len(self.left\_tape): # если ленты не хватает добавляем ячейку

                self.left\_tape.append(0)

    def num (self, a):

        if self.pos\_carriage < 0:

            self.left\_tape[abs(self.pos\_carriage) - 1] = a

        else:

            self.right\_tape[self.pos\_carriage] = a

    def checknum(self):

        if self.pos\_carriage < 0:

            return self.left\_tape[abs(self.pos\_carriage) - 1]

        else:

            return self.right\_tape[self.pos\_carriage]# чтение числа из ячейки

Этот пример иллюстрирует использование машины Поста для выполнения простой задачи. В этом случае, задача заключается в выполнении различных действий в зависимости от значения, находящегося в текущей ячейке ленты.

   machine = PostMachine()

    machine.add\_command("1 1")#0

    #если в ячейке 0 то выполняем вторую команду если нет то первую команду

    machine.add\_command("? 2, 4")#1

    machine.add\_command("0 3")#2

    machine.add\_command("? 4, 5")#3

    machine.add\_command("1 5")#4

    machine.add\_command(". 6")#5

    machine.run()

    result = machine.get\_tape()

    print("Результат на ленте:", result)

**Пример по шагам:**

1.Создается экземпляр машины Поста machine.

2.Добавляются команды с использованием machine.add\_command():

* machine.add\_command("1 1") (строка 2): Эта команда означает, что если значение в текущей ячейке ленты равно 0, то выполнить команду с номером 1, иначе выполнить команду с номером 1.
* machine.add\_command("? 2, 4") (строка 4): Эта команда означает, что нужно выполнить проверку: если значение в текущей ячейке ленты равно 0, то выполнить команду с номером 2, иначе выполнить команду с номером 4.
* machine.add\_command("0 3") (строка 6): Если выполнена команда с номером 2 (которая выполняется, если значение в текущей ячейке равно 0), то записать 0 в текущую ячейку и выполнить команду с номером 3.
* machine.add\_command("? 4, 5") (строка 8): Если выполнена команда с номером 4 (которая выполняется, если значение в текущей ячейке не равно 0), то выполнить проверку: если значение в текущей ячейке равно 0, то выполнить команду с номером 4, иначе выполнить команду с номером 5.
* machine.add\_command("1 5") (строка 10): Если выполнена команда с номером 4 (которая выполняется, если значение в текущей ячейке не равно 0), то записать 1 в текущую ячейку и выполнить команду с номером 5.
* machine.add\_command(". 6") (строка 12): Команда "." завершает выполнение программы.

3.Затем машина Поста запускается с помощью machine.run().

4.После выполнения программы, результат находится на ленте, и он выводится с помощью machine.get\_tape().

В данном случае, результат будет зависеть от исходного значения в текущей ячейке ленты и последовательности выполнения команд. Результат будет выведен в переменной result, и он будет равен значению на ленте после выполнения программы.

# **Заключение**

В ходе выполнения данной работы была проведена исследовательская работа по машине Поста и ее применению для решения типовой задачи на проверку строки на палиндром. Работа включала в себя анализ основных теорем машины Поста, описание решения задачи, а также разработку программы, использующей машину Поста для данной задачи.

## **Выводы по работе:**

* Машина Поста - это мощный инструмент в теории вычислений, который может использоваться для решения разнообразных задач, связанных с обработкой строковых данных.
* Основные теоремы машины Поста, включая теорему о существовании универсальной машины Поста, имеют фундаментальное значение в теории вычислений.
* Программа, разработанная на машине Поста, может успешно определять, является ли введенная строка палиндромом. Это является примером эффективного применения машины Поста в практической задаче.

## **Выводы по своей роли:**

Моя роль (Class Lead) в выполнении этой работы заключалась в создании программы (два класса) для использования машины Поста, а также в создании описания решения типовой задачи на машине Поста.

# **Список литературы**

Машина Поста // Википедия URL::

<https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%B0_%D0%9F%D0%BE%D1%81%D1%82%D0%B0>  
(дата обращения: 27.09.2023).

Программирование на машине Поста (leventov) // Хабр URL: <https://habr.com/ru/articles/111272/>  
(дата обращения: 05.10.2023)

Машина Поста тренажер для изучения универсального исполнителя (Поляков К.) // kpolyakov.spb.ru URL: <https://kpolyakov.spb.ru/prog/post.htm>

(дата обращения: 17.10.2023).