

Отчет по лабораторной работе № 7 по курсу «Фундаментальная информатика»

Студент группы М8О-105Б-21 Минеева Светлана Алексеевна, № по списку 14

Контакты e-mail: svetlana.mineewa2003@yandex.ru

Работа выполнена: «20» октября 2021 г.

Преподаватель: Вячеслав Константинович Титов каф. 805

Отчет сдан «20» октября 2021 г., итоговая оценка _____

Подпись преподавателя _____

1. **Тема:** Программирование в алгоритмической модели Маркова
2. **Цель работы:** составить программу в алгоритмической модели Маркова
3. **Задание (вариант №2.32):** $A=\{a,b,c\}$. Удалить из слова Р третье вхождение символа а, если такое есть.
4. **Оборудование (лабораторное):**
ЭВМ _____, процессор _____, имя узла сети _____ с ОП _____ Мб,
НМД _____ Мб. Терминал _____ адрес _____. Принтер _____
Другие устройства _____

Оборудование ПЭВМ студента, если использовалось:

Процессор 2,9 GHz 2-ядерный процессор Intel Core i5 с ОП 8 Гб, НМД 500 Гб. Монитор 13,3-дюймовый (2560 x 1600).

Другие устройства _____

5. **Программное обеспечение (лабораторное):**
Операционная система семейства _____, наименование _____ версия _____
интерпретатор команд _____ версия _____
Система программирования _____ версия _____
Редактор текстов _____ версия _____
Утилиты операционной системы _____
Прикладные системы и программы _____
Местонахождение и имена файлов программ и данных _____

Программное обеспечение ЭВМ студента, если использовалось:

Операционная система семейства UNIX, наименование Terminal версия 2.10

интерпретатор команд bash версия 3.2.

Система программирования _____ версия _____

Редактор текстов emacs версия 27.2-2

Утилиты операционной системы _____

Прикладные системы и программы _____

Местонахождение и имена файлов программ и данных на домашнем компьютере _____

6. Идея, метод, алгоритм решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальные спецификации с пред- и постусловиями)

Алгоритм:

- 1) Ставим в начало слова *;
- 2) Двигаем * до первого вхождения а и при достижении этого вхождения меняем знак * на # и ставим его справа от а. Однако, если в процессе этого движения мы достигаем конца слова, то меняем * на пустоту и останавливаем программу;
- 3) Двигаем # до второго вхождения а и при достижении этого вхождения меняем знак # на \$ и ставим его справа от а. Однако, если в процессе этого движения мы достигаем конца слова, то меняем # на пустоту и останавливаем программу;
- 4) Двигаем \$ до третьего вхождения а и при достижении этого вхождения меняем знак \$а на пустоту и останавливаем программу. Однако, если в процессе этого движения мы достигаем конца слова, то меняем \$ на пустоту и останавливаем программу.

Я буду использовать интерпретатор алгоритмов Маркова eMain. Ссылка на данный интерпретатор: <https://freeshell.de/~jcm/projects/live/emain/em-interpreter.html>

7. Сценарий выполнения работы [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

```
$a->. // если встречаем $a (третье вхождение а), то заменяем $a на пустоту и останавливаем программу
*a->a# // если встречаем *a (первое вхождение а), то заменяем *a на a#
#a->a$ // если встречаем #a (второе вхождение а), то заменяем #a на a$
*b->b* // если встречаем *b, то заменяем *b на b* - движение к первому вхождению а
*c->c* // если встречаем *c, то заменяем *c на c* - движение к первому вхождению а
#b->b# // если встречаем #b, то заменяем #b на b# - движение ко второму вхождению а
#c->c# // если встречаем #c, то заменяем #c на c# - движение ко второму вхождению а
$b->b$ // если встречаем $b, то заменяем $b на b$ - движение к третьему вхождению а
$c->c$ // если встречаем $c, то заменяем $c на c$ - движение к третьему вхождению а
*->. // если встречаем *, то есть мы дошли до конца строки и не встретили ни одного вхождения а, то заменяем * на пустоту и останавливаем программу
#->. // если встречаем #, то есть мы дошли до конца строки и не встретили второго вхождения а, то заменяем # на пустоту и останавливаем программу
$->. // если встречаем $, то есть мы дошли до конца строки и не встретили третьего вхождения а, то заменяем $ на пустоту и останавливаем программу
->*. // ставим в начало слова *
```

Тесты:

- 1) abcbbacbaa

Вывод: abcbbacba

- 2) bcabcabb

Вывод: bcabcabb

- 3) bacsbaba

Вывод: bacsbab

- 4) aaa

Вывод: aa

Пункты 1-7 отчета составляются строго до начала лабораторной работы.

8. Распечатка протокола (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем).

Last login: Wed Oct 20 09:56:18 on console

The default interactive shell is now zsh.

To update your account to use zsh, please run `chsh -s /bin/zsh`.

For more details, please visit <https://support.apple.com/kb/HT208050>.

MacBook-Pro-MacBook:~ macbookpro\$ cat zag.txt

```
*****
* Минеева Светлана Алексеевна. *
*      М8О-105Б-21              *
*  Лабораторная работа №7      *
*****
```

rules definition or information

```
$a->.
*a->a#
#a->a$
#b->b*
*b->b$
#b->b#
#b->b$
$b->b$
$b->b$
$b->b$
#->.
#->.
$->.
->*
```

input definition

```
abcbbaabaa
```

substitution output

```
[01|13] *abcbbaabaa
[02|02] a#bcbbaabaa
[03|06] ab#cbbaabaa
[04|07] abc#bbaabaa
[05|06] abcb#baabaa
[06|06] abcbba#abaa
[07|03] abcbba$baa
[08|09] abcbba$bba
[09|08] abcbba$bba
[10|01] abcbbaabaa
```

main control

-
- auto pause substitution in case of
 - ☐ a number of substitutions was made
 - ☒ 1
 - ☐ 1
 - ☐ length of resulting word became
 - larger than
 - ☐ resulting word became equal to

statistics

- number of substitutions
- substitution status

debug switches

- add specified info to substitution output
 - ☒ substitutions since start
 - ☐ substitutions since continue
 - ☐ length of resulting word
 - ☒ ordinal number of used rule
- using leading zeros

rules definition or information

```
$a->.  
*a->a#  
#a->a$  
*b->b$  
*b->b*  
*c->c*  
#b->b#  
#c->c#  
$b->b$  
$c->c$  
*->.  
#->.  
$->.  
->*
```

input definition

bcabcabb

substitution output

```
[01|13] *bcabcabb  
[02|04] b*cabcabb  
[03|05] bc*abcabb  
[04|02] bca#bcabb  
[05|06] bcab#cabb  
[06|07] bcabc#abb  
[07|03] bcabc$a$bb  
[08|08] bcabcab$b  
[09|08] bcabcabb$  
[10|12] bcabcabb
```

main control

-
- auto pause substitution in case of
 - ☐ a number of substitutions was made
 - ☒ 1
 - ☐ 1
 - ☐ length of resulting word became
 - larger than
 - ☐ resulting word became equal to

statistics

- number of substitutions
- substitution status

debug switches

- add specified info to substitution output
 - ☒ substitutions since start
 - ☐ substitutions since continue
 - ☐ length of resulting word
 - ☒ ordinal number of used rule
- using leading zeros

rules definition or information

```
$a->.  
*a->a#  
#a->a$  
*b->b$  
*b->b*  
*c->c*  
#b->b#  
#c->c#  
$b->b$  
$c->c$  
*->.  
#->.  
$->.  
->*
```

input definition

baccbaba

substitution output

```
[01|13] *baccbaba  
[02|04] b*accbaba  
[03|02] ba#ccbaba  
[04|07] bac#cbaba  
[05|07] bacc#baba  
[06|06] baccb#aba  
[07|03] baccba$ba  
[08|08] baccbab$a  
[09|01] baccbab
```

main control

-
- auto pause substitution in case of
 - ☐ a number of substitutions was made
 - ☒ 1
 - ☐ 1
 - ☐ length of resulting word became
 - larger than
 - ☐ resulting word became equal to

statistics

- number of substitutions
- substitution status

debug switches

- add specified info to substitution output
 - ☒ substitutions since start
 - ☐ substitutions since continue
 - ☐ length of resulting word
 - ☒ ordinal number of used rule
- using leading zeros

rules definition or information

```
$a->.  
*a->a#  
#a->a$  
*b->b*  
*c->c*  
#b->b#  
#c->c#  
$b->b$  
$c->c$  
*->.  
#->.  
$->.  
->*
```

input definition

```
aaa
```

substitution output

```
[01|13] *aaa  
[02|02] a#aa  
[03|03] aa$a  
[04|01] aa
```

main control

-
- auto pause substitution in case of
 - ☐ a number of substitutions was made
 - ☒ 1
 - ☐ 1
 - ☐ length of resulting word became
 - larger than
 - ☐ resulting word became equal to

statistics

- number of substitutions
- substitution status

debug switches

- add specified info to substitution output
 - ☒ substitutions since start
 - ☐ substitutions since continue
 - ☐ length of resulting word
 - ☒ ordinal number of used rule
- using leading zeros

9. Дневник отладки должен содержать дату и время сеансов отладки и основные события (ошибки в сценарии и программе, нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

№	Лаб. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание
1	Дом.	19.10.21	19:00	Программа зацикливается, если не встречает а, не встречает второе вхождение а, не встречает третье вхождение а	Проблема устранена, программа работает корректно	Включены новые строки кода для устранения проблемы: *->. #->. \$->.

10. Замечания автора по существу работы

Данная работа очень полезна, она отлично развивает мышление.

11. Выводы

Я составила программу в алгоритмической модели Маркова.

Недочёты при выполнении задания могут быть устранены следующим образом: больше практиковаться в написании программ в алгоритмической модели Маркова.

Подпись студента Минеева С.А