МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

# «БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

# Лабораторная работа №3

по дисциплине: Теория автоматов и формальных языков тема: «Регулярные языки и конечные распознаватели»

Выполнил: ст. группы ПВ-222 Гоголев Владимир Владимирович

Проверил:

Рязанов Юрий Дмитриевич

Белгород 2024 г.

**Цель работы:** изучить основные способы задания регулярных языков, способы построения, алгоритмы преобразования, анализа и реализации конечных распознавателей.

**Задание 1.**

Язык L1 в алфавите {a,b}, представляющий собой множество цепочек, в

которых символ а встречается не менее одного раза, а символ b — не более

одного раза, задан грамматикой:

S → aAbA

S → baA

A → Aa

A → ε

Построить детерминированный конечный распознаватель языка L1.

Устраним левую рекурсию:

S → aAbA

S → baA

A → aB1

B1 → aB1

B1 → ε

A → ε

Устраним пустое правило:

S → aAbA

S → aAb

S → abA

S → ab

S → baA

S → ba

A → aB1

A → a

B1 → aB1

B1 → a

Правила начинаются с терминала, поэтому приведём грамматику к правосторонней грамматике:

Выпишем замены:

N1 → Ab : (N1 → aB1b, N1 → ab)

N2 → B1b : (N2 → aN2, N2 → ab)

N3 → N1A : (N3 → aN2A, N3 → abA)

N4 → N2A : (N4 → aN4, N4 → abA)

Получим грамматику:

S → aN3

S → aN1

S → abA

S → ab

S → baA

S → ba

A → aB1

A → a

B1 → aB1

B1 → a

N1 → aN2

N1 → ab

N2 → aN2

N2 → ab

N3 → aN4

N3 → abA

N4 → aN4

N4 → abA

Приводим к автоматной правосторонней грамматике:

S → aN3

S → aN1

S → aS1

S1 → bA

S → aS5

S5 → bE

S → bS2

S2 → aA

S → bS6

S6 → aE

A → aB1

A → aE

B1 → aB1

B1 → aE

N1 → aN2

N1 → aS7

S7 → bE

N2 → aN2

N2 → aS8

S8 → bE

N3 → aN4

N3 → aS3

S3 → bA

N4 → aN4

N4 → aS4

S4 → bA

E → ε

Получаем НКР:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | S | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 | A | B1 | N1 | N2 | N3 | N4 | E |
| a | N3  N1  S1  S5 |  | A |  |  |  | E |  |  | B1  E | B1  E | N2  S7 | N2  S8 | N4  S3 | N4  S4 |  |
| b | S2  S6 | A |  | A | A | E |  | E | E |  |  |  |  |  |  |  |

Приведём НКР к ДКР:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | {S} | {N3,  N1,  S1,  S5} | {N4  S3,  N2,  S7} | {N4,  S4,  N2,  S8} | {S2,  S6} | {A,  E} | {B1,  E} |
| a | {N3,  N1,  S1,  S5} | {N4  S3,  N2,  S7} | {N4,  S4,  N2,  S8} | {N4,  S4,  N2,  S8} | {A,  E} | {B1,  E} | {B1,  E} |
| b | {S2,  S6} | {A,  E} | {A,  E} | {A,  E} |  |  |  |

Получим новый детерминированный распознаватель:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **↓** |  |  |  |  | 1 | 1 |
|  | S | S1 | S2 | S3 | S4 | A | B1 |
| a | S1 | S2 | S3 | S3 | A | B1 | B1 |
| b | S4 | A | A | A |  |  |  |

Получим минимизированный детерминированный распознаватель (потребуется в 3-ем задании):

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | К1 | | | | | | К2 | |
|  | S | S1 | S2 | S3 | S4 |  | A | B1 |
| a | К1 S1 | К1 S2 | К1 S3 | К1 S3 | К2 A | К1 | К2 B1 | К2 B1 |
| b | К1 S4 | К2 A | К2 A | К2 A | К1 | К1 | К1 | К1 |

Класс 1-эквивалентности:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | К1 | | | К2 | | | К3 | |
|  | S | S4 |  | S1 | S2 | S3 | A | B1 |
| a | К2 S1 | К3 A | К1 | К2 S2 | К2 S3 | К2 S3 | К3 B1 | К3 B1 |
| b | К1 S4 | К1 | К1 | К3 A | К3 A | К3 A | К1 | К1 |

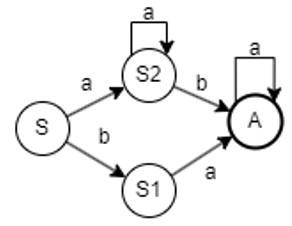
Класс 2-эквивалентности:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | К5 | К4 | К1 | К2 | | | К3 | |
|  | S | S4 |  | S1 | S2 | S3 | A | B1 |
| a | К2 S1 | К3 A | К1 | К2 S2 | К2 S3 | К2 S3 | К3 B1 | К3 B1 |
| b | К4 S4 | К1 | К1 | К3 A | К3 A | К3 A | К1 | К1 |

Распознаватель:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **↓** |  |  | 1 |
|  | S | S1 | S2 | A |
| a | S2 |  | S2 | A |
| b | S1 | A | A |  |

Схема распознавателя:



**Задание 2.** Язык L2 в алфавите {a,b}, представляющий собой множество цепочек, в

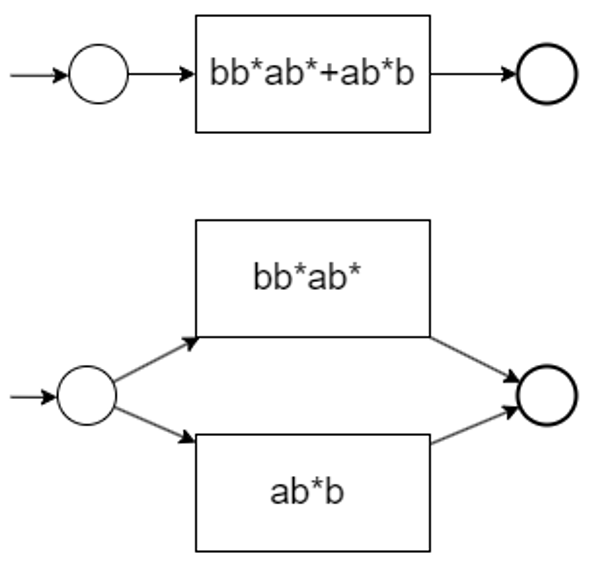
которых символ b встречается не менее одного раза, а символ а — не более

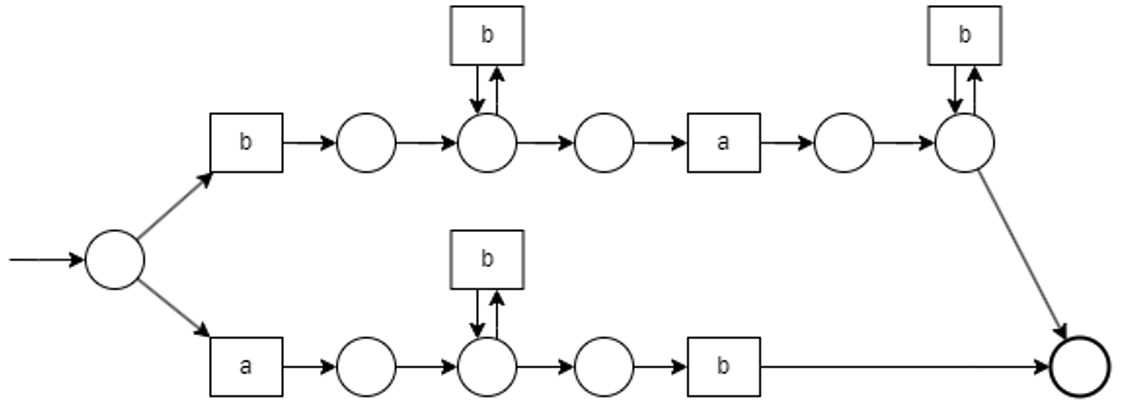
одного раза, задан регулярным выражением:

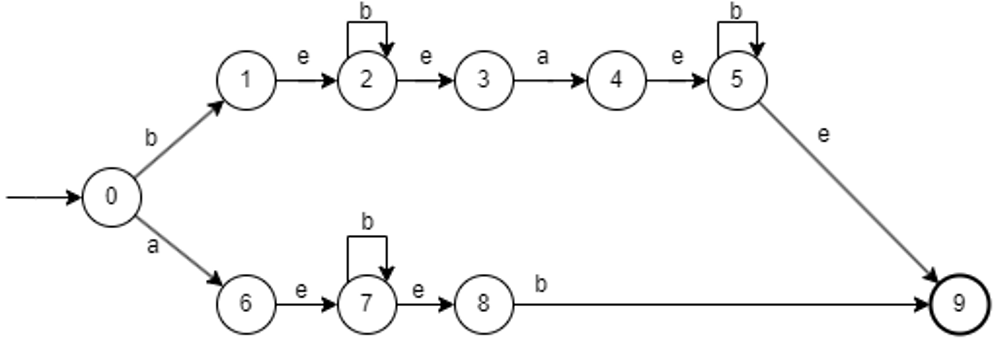
bb\*ab\*+ab\*b

Построить детерминированный конечный распознаватель языка L2.

Вывод недетерминированного распознавателя по шагам:







Устраним e-переходы:

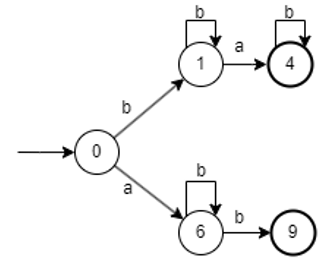
Недетерминированный конечный распознаватель:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | S0 | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 | S9 |
| a | S6 |  |  | S4 |  |  |  |  |  |  |
| b | S1 |  | S2 |  |  | S5 |  | S7 | S9 |  |
| e |  | S2 | S3 |  | S5 | S9 | S7 | S8 |  |  |

Устраним e-переходы:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | e(S0)  {S0} | e(S1)  {S1,S2,S3} | e(S2)  {S2,S3} | e(S3)  {S3} | e(S4)  {S4,S5,S9} | e(S5)  {S5,S9} | e(S6)  {S6,S7,S8} | e(S7)  {S7,S8} | e(S8) | e(S9) |
| a | e(S6) | e(S4) | e(S4) | e(S4) |  |  |  |  |  |  |
| b | e(S1) | e(S2) | e(S2) |  | e(S5),  e(S9) | e(S5),  e(S9) | e(S7),  e(S9) | e(S7),  e(S9) | e(S9) |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **↓** |  | 1 |  | 1 |
|  | S0 | S1 | S4 | S6 | S9 |
| a | S6 | S4 |  |  |  |
| b | S1 | S1 | S4 | S6,S9 |  |



Приведём к детерминированному конечному распознавателю:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **↓** |  | 1 |  | 1 |
|  | {S0} | {S1} | {S4} | {S6} | {S6,S9} |
| a | {S6} | {S4} |  |  |  |
| b | {S1} | {S1} | {S4} | {S6,S9} | {S6,S9} |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **↓** |  | 1 |  | 1 |
|  | S0 | S1 | S4 | S6 | S6\_1 |
| a | S6 | S4 |  |  |  |
| b | S1 | S1 | S4 | S6\_1 | S6\_1 |

Получим минимизированный детерминированный конечный распознаватель (потребуется в 3-ем задании):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | К1 | | | | К2 | |
|  | S0 | S1 | S6 |  | S6\_1 | S4 |
| a | К1 S6 | К2 S4 | К1 | К1 | К1 | К1 |
| b | К1 S1 | К1 S1 | К2 S6\_1 | К1 | К2 S6\_1 | К2 S4 |

Класс 1-эквивалентности:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | К1 | | К4 | К3 | К2 | |
|  | S0 |  | S1 | S6 | S6\_1 | S4 |
| a | К1 S6 | К1 | К2 S4 | К1 | К1 | К1 |
| b | К1 S1 | К1 | К1 S1 | К2 S6\_1 | К2 S6\_1 | К2 S4 |

Класс 2-эквивалентности:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | К5 | К4 | К3 | К1 | К2 | |
|  | S0 | S1 | S6 |  | S6\_1 | S4 |
| a | К3 S6 | К2 S4 | К1 | К1 | К1 | К1 |
| b | К4 S1 | К4 S1 | К2 S6\_1 | К1 | К2 S6\_1 | К2 S4 |

Получим минимизированный детерминированный конечный распознаватель:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **↓** |  |  | 1 |
|  | S0 | S1 | S6 | S4 |
| a | S6 | S4 |  |  |
| b | S1 | S1 | S4 | S4 |

**Задание 3.** Построить минимальный детерминированный конечный распознаватель

языка L3 в алфавите {a,b}, представляющий собой множество цепочек, в

которых символ а встречается не менее одного раза, а символ b — не более

одного раза, или символ b встречается не менее одного раза, а символ а — не

более одного раза.

Заметим, что язык L1 и L2 схожи за исключением того, что символы инвертированы. В языке L3 нужно учесть верные строки ‘ab’ и ‘ba’ и также неоднозначность после ‘ab’ и ‘ba’, где могут продолжиться строчка из a или строчка из b.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **↓** |  |  | 1 |
|  | S | S1 | S2 | A |
| a | S2 |  | S2 | A |
| b | S1 | A | A |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **↓** |  |  | 1 |
|  | S0 | S1 | S6 | S4 |
| a | S6 | S4 |  |  |
| b | S1 | S1 | S4 | S4 |

Выпишем языки L1 и L2 в виде КС-грамматик:

|  |  |
| --- | --- |
| L1 | L2 |
| S → aS2  S → bS1  S1 → bA  S2 → aS2  S2 → bA  A → aA  A → e | T → bT2  T → aT1  T1 → aB  T2 → bT2  T2 → aB  B → bB  B → e |

Объединим их и добавим правила S` → S и S` → T, где S` – стартовый нетерминал:

S` → S

S` → T

S → aS2

S → bS1

S1 → bA

S2 → aS2

S2 → bA

A → aA

A → e

T → bT2

T → aT1

T1 → aB

T2 → bT2

T2 → aB

B → bB

B → e

Применим правила:

S` → aS2

S` → bS1

S1 → bA

S2 → aS2

S2 → bA

A → aA

A → e

S` → bT2

S` → aT1

T1 → aB

T2 → bT2

T2 → aB

B → bB

B → e

НКР имеет вид:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **↓** |  |  | 1 |  |  | 1 |
|  | S` | S1 | S2 | A | T1 | T2 | B |
| a | S2  T1 |  | S2 | A | B | B |  |
| b | S1  T2 | A | A |  |  | T2 | B |

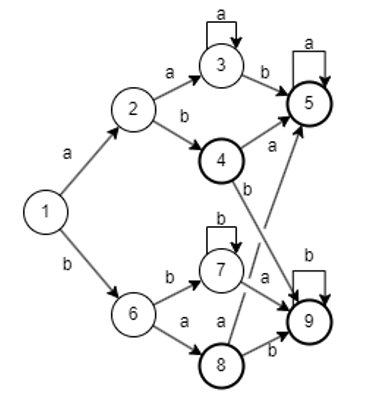
Приведём к ДКР:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | {S`} | {S1} | {S2} | {A} | {T1} | {T2} | {B} |
| a | {S2,T1} |  | {S2} | {A} | {B} | {B} |  |
| b | {S1,T2} | {A} | {A} |  |  | {T2} | {B} |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | {S`} | {S2,T1} | {S1,T2} | {S2,B} | {T2,A} | {A,B} | {S1} | {S2} | {A} | {T1} | {T2} | {B} |
| a | {S2,T1} | {S2,B} | {B} | {S2} | {A,B} | {A} |  | {S2} | {A} | {B} | {B} |  |
| b | {S1,T2} | {A} | {T2,A} | {A,B} | {T2} | {B} | {A} | {A} |  |  | {T2} | {B} |

ДКР имеет вид:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **↓** |  |  | 1 | 1 |  |  | 1 | 1 |
|  | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 | S9 |
| a | S2 | S3 | S3 | S5 | S5 | S8 | S9 | S5 |  |
| b | S6 | S4 | S5 | S9 |  | S7 | S7 | S9 | S9 |



Минимизируем, используя эквивалентные преобразования:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | К1 | | | | | | К2 | | | |
|  | S1 | S2 | S3 | S6 | S7 |  | S4 | S8 | S5 | S9 |
| a | K1 S2 | K1 S3 | K1 S3 | K2 S8 | K2 S9 | К1 | K2 S5 | K2 S5 | К2 S5 | К1 |
| b | K1 S6 | K2 S4 | K2 S5 | K1 S7 | K1 S7 | К1 | K2 S9 | K2 S9 | К1 | К2 S9 |

Классы 1-эквивалентности:

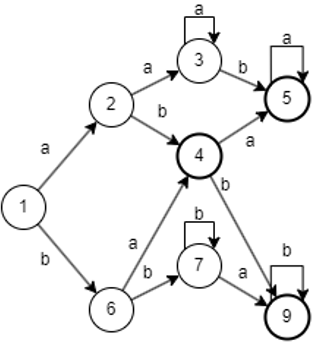
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | К1 | | | | | | К3 | | К2 | |
|  | S1 | S2 | S3 | S6 | S7 |  | S4 | S8 | S5 | S9 |
| a | K1 S2 | K1 S3 | K1 S3 | K3 S8 | K2 S9 | К1 | K2 S5 | K2 S5 | К2 S5 | K1 |
| b | K1 S6 | K3 S4 | K2 S5 | K1 S7 | K1 S7 | К1 | K2 S9 | K2 S9 | K1 | К2 S9 |

Классы 2-эквивалентности:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | К8 | К7 | К6 | К5 | К4 | К1 | К3 | | К2 | |
|  | S1 | S2 | S3 | S6 | S7 |  | S4 | S8 | S5 | S9 |
| a | K7 S2 | K6 S3 | K6 S3 | K3 S8 | K2 S9 | К1 | K2 S5 | K2 S5 | К2 S5 | K1 |
| b | K6 S6 | K3 S4 | K2 S5 | K4 S7 | K4 S7 | К1 | K2 S9 | K2 S9 | K1 | К2 S9 |

Получим минимизированный детерминированный конечный распознаватель:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **↓** |  |  | 1 | 1 |  |  | 1 |
|  | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S9 |
| a | S2 | S3 | S3 | S5 | S5 | S4 | S9 |  |
| b | S6 | S4 | S5 | S9 |  | S7 | S7 | S9 |



**Задание 4.** Написать программу компиляционного типа для реализации

минимального детерминированного конечного распознавателя языка L3.

Код программы:

MESSAGES = {  
 -1: "Отвергнуть. Последовательность пуста",  
 -2: "Отвергнуть. Входной символ невалидный",  
 -3: "Отвергнуть. Символ b должен быть введён не более 1 раза",  
 -4: "Отвергнуть. Символ а должен быть введён не более 1 раза",  
 0: "Допустить."  
}  
  
  
def L3validator\_comp(input: str):  
 S = 1  
 while (True):  
 if S == 1:  
 if len(input) == 0:  
 S = -1  
 elif input[0] == 'a':  
 S = 2  
 elif input[0] == 'b':  
 S = 6  
 else:  
 S = -2  
 elif S == 2:  
 if len(input) == 0:  
 S = -5  
 elif input[0] == 'a':  
 S = 3  
 elif input[0] == 'b':  
 S = 4  
 else:  
 S = -2  
 elif S == 3:  
 if len(input) == 0:  
 S = -5  
 elif input[0] == 'a':  
 S = 2  
 elif input[0] == 'b':  
 S = 5  
 else:  
 S = -2  
 elif S == 4:  
 if len(input) == 0:  
 S = 0  
 elif input[0] == 'a':  
 S = 5  
 elif input[0] == 'b':  
 S = 9  
 else:  
 S = -2  
 elif S == 5:  
 if len(input) == 0:  
 S = 0  
 elif input[0] == 'a':  
 S = 5  
 elif input[0] == 'b':  
 S = -3  
 else:  
 S = -2  
 elif S == 6:  
 if len(input) == 0:  
 S = -6  
 elif input[0] == 'a':  
 S = 4  
 elif input[0] == 'b':  
 S = 7  
 else:  
 S = -2  
 elif S == 7:  
 if len(input) == 0:  
 S = -6  
 elif input[0] == 'a':  
 S = 9  
 elif input[0] == 'b':  
 S = 7  
 else:  
 S = -2  
 elif S == 9:  
 if len(input) == 0:  
 S = 0  
 elif input[0] == 'a':  
 S = -4  
 elif input[0] == 'b':  
 S = 9  
 else:  
 S = -2  
 else:  
 raise Exception("Undefined state.")  
  
 if S <= 0:  
 return S  
  
 input = input[1:]

**Задание 5.** Написать программу интерпретационного типа для реализации

минимального детерминированного конечного распознавателя языка L3.

Код программы:

CONVERT = {  
 2: -5,  
 3: -5,  
 6: -6,  
 7: -6,  
 -1: -1,  
 -2: -2,  
 -3: -3,  
 -4: -4,  
 -5: -5,  
 -6: -6,  
}  
  
MESSAGES = {  
 -1: "Отвергнуть. Последовательность пуста",  
 -2: "Отвергнуть. Входной символ невалидный",  
 -3: "Отвергнуть. Символ b должен быть введён не более 1 раза",  
 -4: "Отвергнуть. Символ а должен быть введён не более 1 раза",  
 -5: "Отвергнуть. Символ b должен быть введён хотя бы 1 раз",  
 -6: "Отвергнуть. Символ a должен быть введён хотя бы 1 раз",  
 0: "Допустить",  
}  
  
PERMITTING = [4, 5, 8]  
  
MATRIX = {  
 'a': [2, 3, 3, 5, 5, 4, 8, -4],  
 'b': [6, 4, 5, 8, -3, 7, 7, 8],  
}  
  
  
def L3validator\_interp(input):  
 if len(input) == 0:  
 return -1  
  
 S = 1  
 while len(input) > 0 and S > 0:  
 if not input[0] in MATRIX.keys():  
 return -2  
 S = MATRIX[input[0]][S - 1]  
 input = input[1:]  
  
 if S in PERMITTING:  
 return 0  
  
 return S

**Задание** **6.** Подобрать наборы тестовых данных так, чтобы в процессе тестирования

сработал каждый переход конечного распознавателя.

Тестовые данные:

"aaab" – Допустить

"aab" – Допустить

"aaba" – Допустить

"ab" – Допустить

"aba" – Допустить

"abaa" – Допустить

"ba" – Допустить

"bab" – Допустить

"babb" – Допустить

"bbba" – Допустить

"bba" – Допустить

"bbbab" – Допустить

"bbab" – Допустить

**Задание** **7.** Подобрать наборы тестовых данных так, чтобы в процессе тестирования

распознаватель закончил обработку цепочек в каждом состоянии конечного

распознавателя.

Тестовые данные:

"" – Отвергнуть

"c" – Отвергнуть

"a" – Отвергнуть

"aaa" – Отвергнуть

"ab" – Допустить

"ba" – Допустить

"aab" – Допустить

"aba" – Допустить

"baa" – Допустить

"aabb" – Отвергнуть

"b" – Отвергнуть

"bb" – Отвергнуть

"bba" – Допустить

"bab" – Допустить

"abb" – Допустить

"bbaa" – Отвергнуть

**Задание** **8.** Выполнить тестирование программ для реализации минимального

детерминированного конечного распознавателя языка L3.

check\_set = [  
 ("aaab", 1),  
 ("aab", 1),  
 ("aaba", 1),  
 ("ab", 1),  
 ("aba", 1),  
 ("abaa", 1),  
 ("ba", 1),  
 ("bab", 1),  
 ("babb", 1),  
 ("bbba", 1),  
 ("bba", 1),  
 ("bbbab", 1),  
 ("bbab", 1),  
 ("", 0), # 1  
 ("c", 0), # 1  
 ("a", 0), # 2  
 ("aaa", 0), # 3  
 ("ab", 1), # 4  
 ("ba", 1), # 4  
 ("aab", 1), # 5  
 ("aba", 1), # 5  
 ("baa", 1), # 5  
 ("aabb", 0), # 5  
 ("b", 0), # 6  
 ("bb", 0), # 7  
 ("bba", 1), # 9  
 ("bab", 1), # 9  
 ("abb", 1), # 9  
 ("bbaa", 0) # 9  
]  
  
for check in check\_set:  
 res = L3validator(check[0])  
 if check[1]:  
 print(check[0], res >= 0, MESSAGES[res], sep='\t')  
 else:  
 print(check[0], res < 0, MESSAGES[res], sep='\t')

Вывод программы:

aaab True Допустить.

aab True Допустить.

aaba True Допустить.

ab True Допустить.

aba True Допустить.

abaa True Допустить.

ba True Допустить.

bab True Допустить.

babb True Допустить.

bbba True Допустить.

bba True Допустить.

bbbab True Допустить.

bbab True Допустить.

True Отвергнуть. Последовательность пуста

c True Отвергнуть. Входной символ невалидный

a True Отвергнуть. Символ b должен быть введён хотя бы 1 раз

aaa True Отвергнуть. Символ b должен быть введён хотя бы 1 раз

ab True Допустить.

ba True Допустить.

aab True Допустить.

aba True Допустить.

baa True Допустить.

aabb True Отвергнуть. Символ b должен быть введён не более 1 раза

b True Отвергнуть. Символ a должен быть введён хотя бы 1 раз

bb True Отвергнуть. Символ a должен быть введён хотя бы 1 раз

bba True Допустить.

bab True Допустить.

abb True Допустить.

bbaa True Отвергнуть. Символ а должен быть введён не более 1 раза

**Вывод:** в ходе работы изучены основные способы задания регулярных языков, способы построения, алгоритмы преобразования, анализа и реализации конечных распознавателей.