МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

**Домашняя работа №2**

Вариант 10

Выполнил:

Круглов Егор Ильич, P3324

Преподаватель:

Лаздин Артур Вячеславович

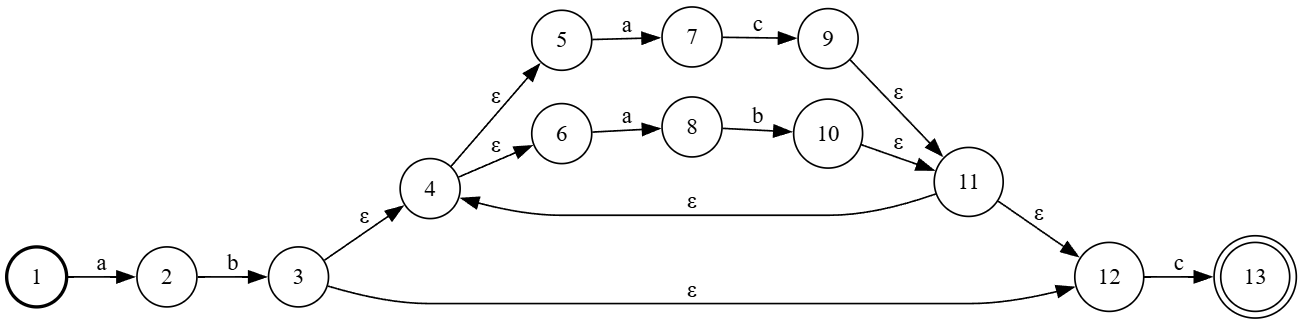
Санкт-Петербург

2025

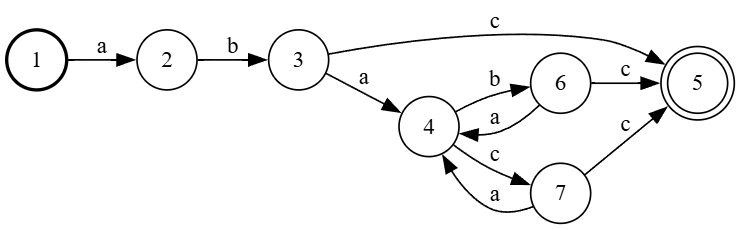
**Выполнение**

Регулярное выражение - *ab(ac|ab)\*c.*

**НКА**

****

**ДКА (не минимальный)**

****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | State | a | b | c |
| 1 | 1 | 2 | - | - |
| 2 | 2 | - | 3, 4, 5, 6, 12 | - |
| 3 | 3, 4, 5, 6, 12 | 7, 8 | - | 13 |
| 4 | 7, 8 | - | 4, 5, 6, 10, 11, 12 | 4, 5, 6, 9, 11, 12 |
| 5 | 13 | - | - | - |
| 6 | 4, 5, 6, 9, 11, 12 | 7, 8 | - | 13 |
| 7 | 4, 5, 6, 10, 11, 12 | 7, 8 | - | 13 |

**ДКА (минимальный)**

Разделим состояния на группы:

{1, 2, 3, 4, 6, 7} – неконечные

{5} – конечные

1. P = { {1, 2, 3, 4, 6, 7}, {5}}

W = {5}

В {5} можно попасть по c из {3, 6, 7}.

Разбиваем {1, 2, 3, 4, 6, 7} на {1, 2, 4} и {3, 6, 7}.

1. P = {{1, 2, 4}, {3, 6, 7}, {5}}

W = {3, 6, 7}

В {3, 6, 7} можно попасть по b из {2, 4} и с из {4}

Разбиваем {1, 2, 4} на {1}, {2, 4} -> {1}, {2}, {4}

1. P = {{1}, {2}, {4}, {3, 6, 7}, {5}}

W = {1}

В {1} нельзя никак попасть

1. P = {{1}, {2}, {4}, {3, 6, 7}, {5}}

W = {2}

В {2} можно попасть по a из 1 (уже выделили в отдельную группу)

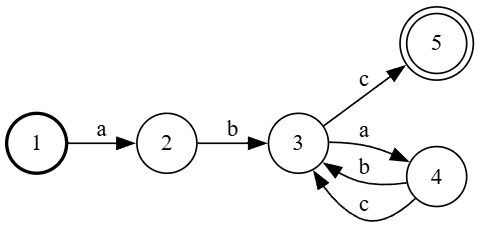
1. P = {{1}, {2}, {4}, {3, 6, 7}, {5}}

W = {4}

В {4} можно попасть по a из {3, 6, 7} (уже выделили в отдельную группу)

Таким образом получаем новое минимальное разбиение: {{1}, {2}, {4}, {3, 6, 7}, {5}}.

Объединяем {3, 6, 7} в новое {3}.



|  |  |  |  |
| --- | --- | --- | --- |
| State | a | b | c |
| 1 | 2 | - | - |
| 2 | - | 3 | - |
| 3 | 4 | - | 5 |
| 4 | - | 3 | 3 |
| 5 | - | - | - |

**Код:**

Ссылка на github: <https://github.com/KruglovEgor/Compilers/blob/main/lab2/main.py>

1. **class** DFA:
2. **def** \_\_init\_\_(self):
3. # Определяем таблицу переходов: {текущее состояние: {символ: следующее состояние}}
4. self.transitions **=** {
5. 1: {'a': 2},
6. 2: {'b': 3},
7. 3: {'a': 4, 'c': 5},
8. 4: {'b': 3, 'c': 3},
9. 5: {}
10. }
11. self.start\_state **=** 1  # Начальное состояние
12. self.accept\_states **=** {5}  # Множество терминальных состояний
14. **def** recognize(self, input\_string):
15. state **=** self.start\_state
16. **for** char **in** input\_string:
17. **if** char **in** self.transitions[state]:
18. state **=** self.transitions[state][char]
19. **else**:
20. **return** False  # Недопустимый символ или нет перехода
21. **return** state **in** self.accept\_states  # Проверяем, является ли состояние конечным

24. # Создаем экземпляр ДКА
25. dfa **=** DFA()
27. # Тестовые строки, которые должны принадлежать языку
28. test\_strings **=** {
29. "valid": [
30. "abc",
31. "ababacc",
32. "abacc",
33. "abababc"
34. ],
35. "invalid": [
36. "",
37. "abab",
38. "cabac",
39. "abacac"
40. ]
41. }
43. print("Ожидаем, что принадлежит:")
44. **for** string **in** test\_strings["valid"]:
45. result **=** "принадлежит" **if** dfa.recognize(string) **else** "не принадлежит"
46. print(f"Строка '{string}' {result} языку.")
48. print("\nОжидаем, что не принадлежит:")
49. **for** string **in** test\_strings["invalid"]:
50. result **=** "принадлежит" **if** dfa.recognize(string) **else** "не принадлежит"
51. print(f"Строка '{string}' {result} языку.")