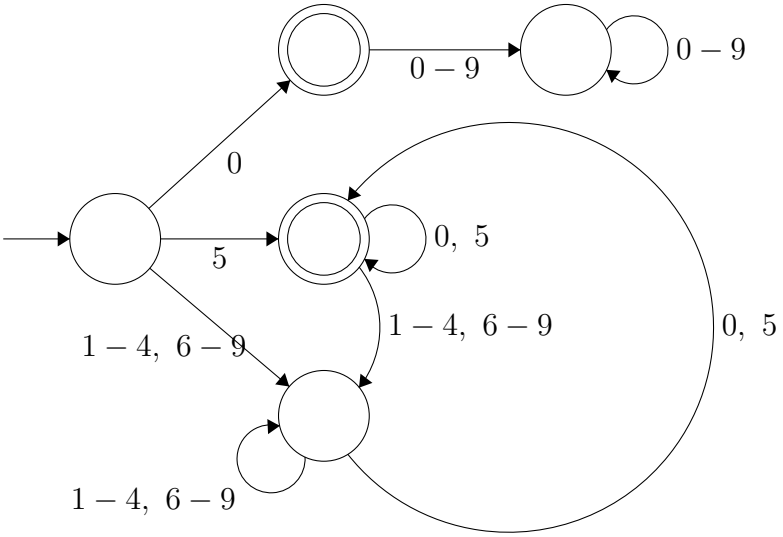
Домашнее задание 1 по формальным языкам

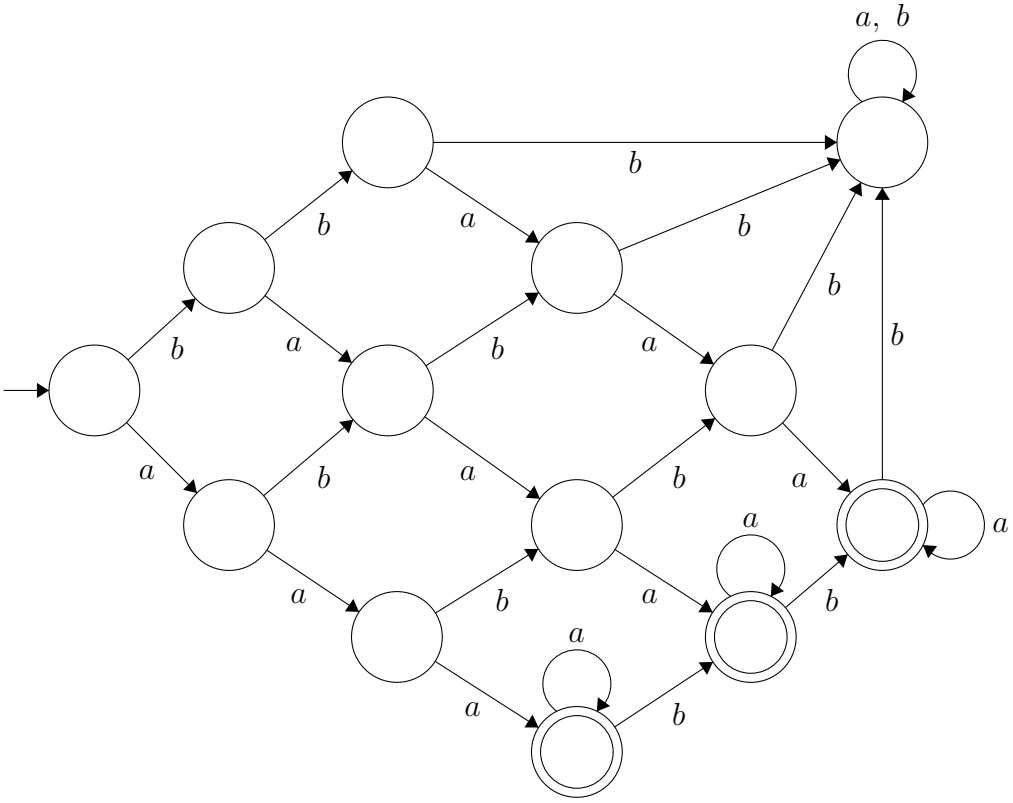
Денисов Никита

13.09.2021

1. Считаем что на вход могут дать пустое слово, что не является числом и соответственно не делится на 5. Пользуемся признаком делимости на 5: если последняя цифра 0 или 5, то и все число делится на 5. Также надо добавить проверку на лидирующие нули: если он один и больше ничего нет, то ок, а если сначала ноль, а потом что-то, то попадаем в сток. Запись на ребре 1-4 обозначает 4 ребра с надписями 1, 2, 3 и 4 соответственно. Аналогично 6-9.



2. Автомат на рисунке. Сделали хотя бы 3 шага вниз $\Rightarrow$ встретили хотя бы 3 буквы a. Если же сделали хотя бы 3 шага вверх, то встретили больше двух b, а значит слово уже не подошло — попали в сток.



3. Нашел документацию языка Python. Из особенностей лексического синтаксиса узнал про то, как используются ключевые слова async и await, а именно: перед объявлением функции идет слово assync, а в теле используется конструкция await function(), которая ждет пока вызванная функция завершится. Их особенность состоит в том, что они взяты за основу для множественных асинхронных фреймворков. Вот пример:

```
import asyncio
import time

async def say_after(delay, what):
    await asyncio.sleep(delay)
    print(what)

async def main():
    print(f"started at {time.strftime('%X')}")

    await say_after(1, 'hello')
    await say_after(2, 'world')

    print(f"finished at {time.strftime('%X')}")

asyncio.run(main())
```

4. Конечный автомат это орграф, поэтому удобно задать его списком ориентированных рёбер. А именно, по данному автомату обойдем его dfs и когда будем проходить очередное ребро будем записывать строчку: "v *to* x *by* c" (без кавычек), где v — состояние из которого исходит ребро, x — состояние куда приходит ребро, c — символ алфавита, соответствующий текущему ребру. Начальное состояние будет всегда нумероваться индексом 0. Терминальные состояния отмечаются в конце следующим образом: "*terminal* n" (без кавычек), после чего следует n строк с индексами вершин, которые являются терминальными.

Итого, язык состоит из: цифр (для описания чисел, которые задают номер состояния данного автомата); ключевых слов *to*, *by* и *terminal*; символов алфавита; пробельных символов.

Прикладываю 3 файла, описывающих разные конечные автоматы.

first.txt описывает автомат, который распознает неотрицательные двоичные числа без лидирующих нулей, делящиеся на 2.

second.txt описывает автомат, который распознает последовательность символов из алфавита $\{a, b\}^*$ четной длины.

third.txt описывает автомат, который распознает есть ли среди последовательности символов из алфавита $\{x, y, z\}^*$ хотя бы один x.

5. Взял готовый скелет экстешена в папке vscode_ext и поменял там регулярку, чтобы она матчила мои ключевые слова для их подсвечивания: $to, by, terminal$. Теперь при открытии в среде разработки vscode файлов и нажатии f5 должна быть подсветка.