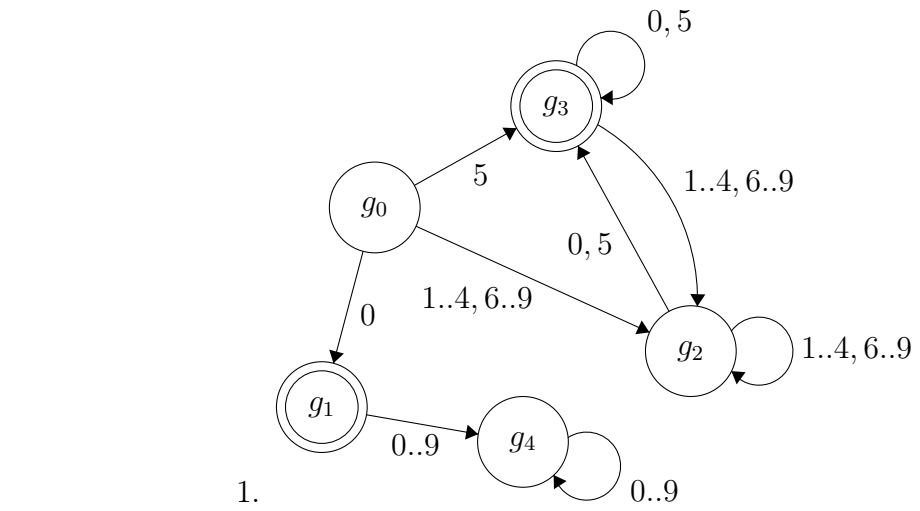
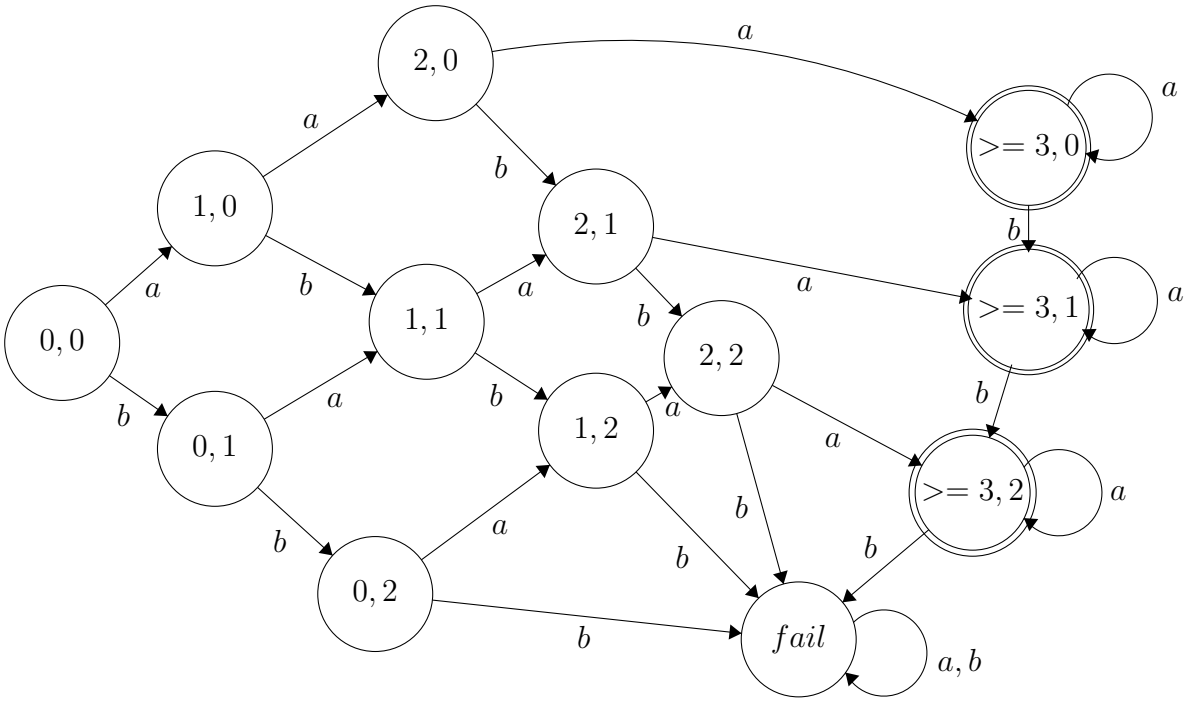
Формальные языки 1

Зеленцова Ирина

13.09.2021



1.



2.

3. *HaskellDocumentation – link* 27 страница - лексические структуры
   Поскольку по остальным языкам ищется плохо, значит Haskell станет любимым(за одно узнаю синтаксис)

   (a) Комментарии: однострочные: $--$, расчитанные на несколько строк: $\{-coment-\}$

(b) операция присваивания $let\ var\_name = value$

(c) стандартный вывод строк $putStrLn"string"$

(d) вывод переменных: $print(variable)$

(e) [1..10] - генерация

(f) $[[Int]]$ массив, $[Int]$ - список символов

(g) упорядоченные множества (2.4, "cat") (Float, [Char])

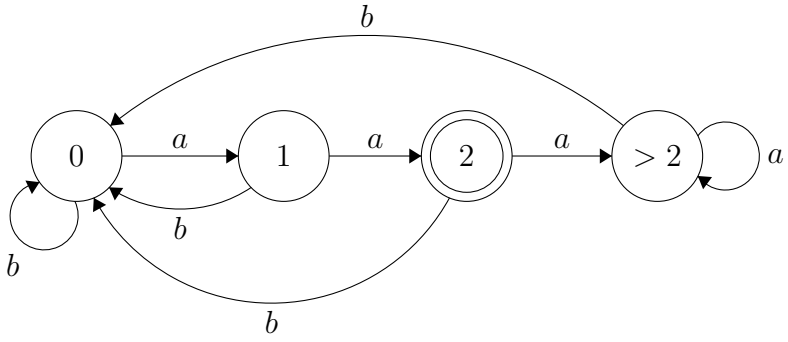
(h) функции
```
square :: Integer -> Integer
square x = x*x
```

(i) конструкция ветвления
```
abs x = if x>=0 then x else -x
```

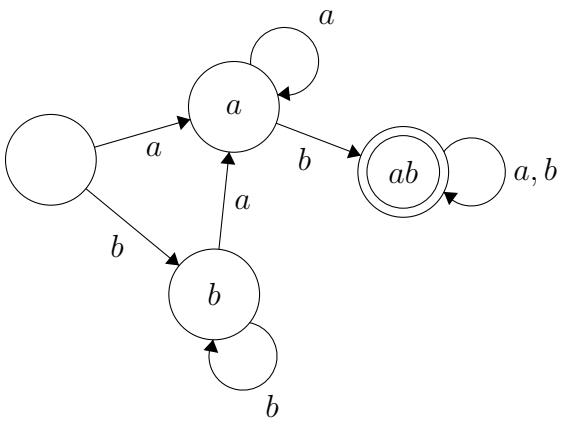
(j) сопоставлене с образцом
```
fact :: Integer -> Integer
fact 0 = 1
fact n = n * fact (n-1)
```

4. Поскольку конечный автомат задается ориентированным графом, то можем для каждого автомата задать его матрицей, и каждая матрица задает ориентированный граф. Тогда запишем матрицу как одну строку, где вид будет следующий:
$\{g_0|0\{a_1, q_{i1}\}, \{a_2, q_{j1}\}, ...\}\{g_1|0\{a_1, q_{i2}\}, ...\}$, где для каждого состояния идет описание пар вида {исходное состояние, является ли оно терминальным {элемент алфавита(переход), состояние куда оно ведет}...}

   Примеры: 1) из первого номера $\{g_0|0\{0, g_1\}, \{5, g_3\}, \{1..4, 6..9, g_2\}\}$
$\{g_1|1\{0..9, g_4\}, \}\{g_2|0\{0, 5g_3\}, \{1..4, 6..9g_2\}\}\{g_3|1\{1..4, 6..9g_2\}\{0, 5g_3\}\}\{g_4|0\{0..9, g_4\}\}$
2)



   оканчивается ли строка на "aa"
$\{0|0\{a, 1\}, \{b, 0\}\}\{1|0\{a, 2\}\{b, 0\}\}\{2|1\{a, > 2\}\{b, 0\}\}\{> 2|0\{a, > 2\}\{b, 0\}\}$
3)



есть ли подстрока "ab"$\{|0\{a,a\}\{b,b\}\}\{a|0\{a,a\}\{b,ad\}\}\{b|0\{a,a\}\{b,b\}\}\{ab|1\{a,b,ab\}\}$

5.
```
  bool del(Expression const &request) {
    try {
      pqxx::work W{*C};
#ifdef LOGGING
      std::cout << "DELETE FROM " + table_ + " WHERE " + request.expr_
                << std::endl;
#endif
      pqxx::result R =
          W.exec("DELETE FROM " + table_ + " WHERE " + request.expr_);
      W.commit();
      return true;
    } catch (std::exception const &e) {
      std::cerr << e.what() << std::endl;
      return false;
    }
  }
```

На всякий случай, если у вас дефолтные ц вета отличаются(Для примера было взято ORM с весеннего проекта. Весь файл есть в репозитории)
Подсветка:

(a) зарезервированные слова языка такие как return, try, catch, const, int, if, else и т.д.

(b) синтаксис для SQL запросов(чтобы их было легче видеть в остальном коде и не сливались с переменными и строками)

(c) использование структур данных/функций из std

(d) свои собственные классы

(e) классы и важные функции, которые берутся из библиотеки драйвера
(последние три пункта чтобы понимать с каким объектом мы работаем и куда лезть в случае неполадок)