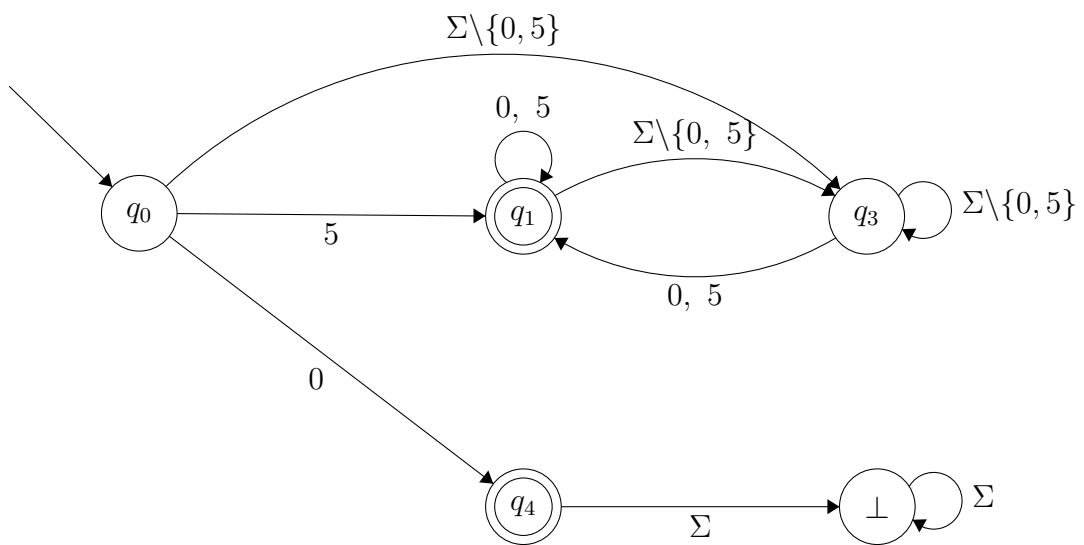


Домашняя работа

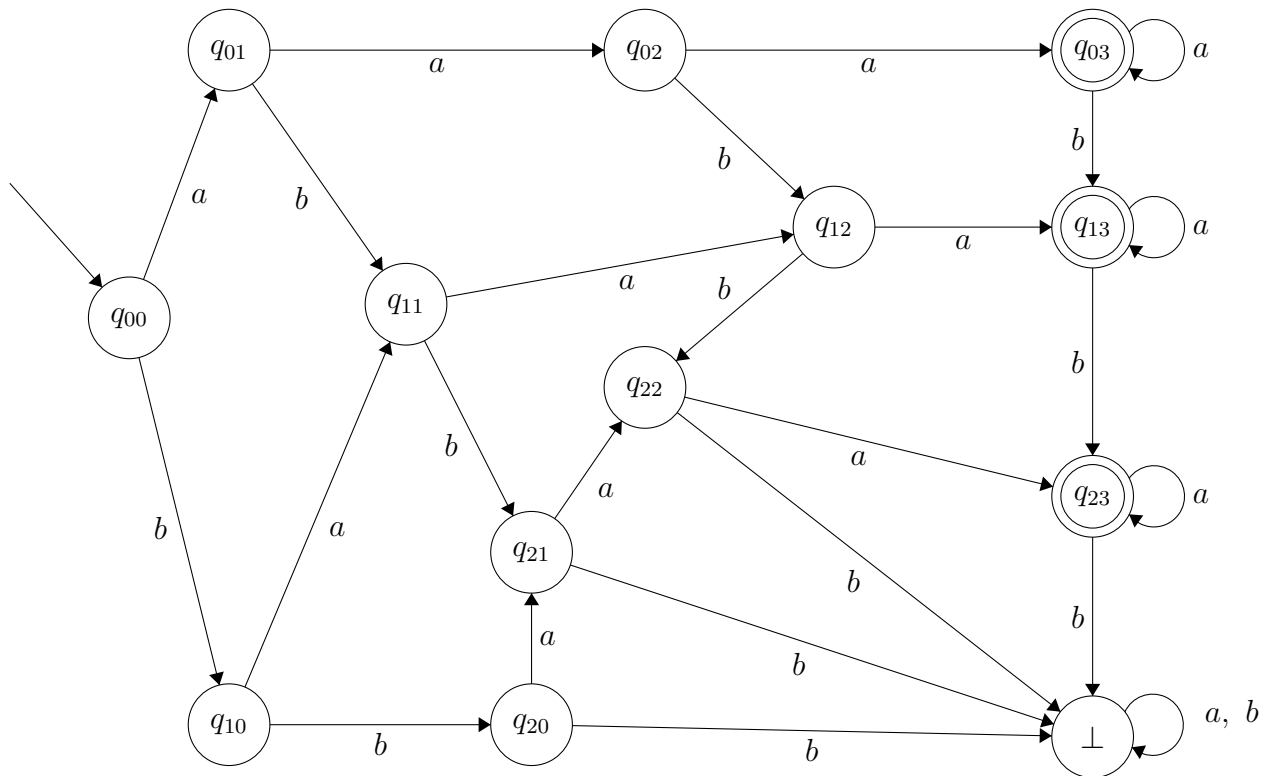
Суворов Вячеслав

13 сентября 2021 г.

1. Начальная вершина q_0



2. q_{cd} показывает, сколько букв a . s показывает, сколько букв b , причем если $d == 3$, то это значит что букв a больше или равно трем. Начальная вершина q_{00}



3. Для описания лексического синтаксиса, я выбрал язык Kotlin.
Можно кликнуть по названию пункта и перейти на официальную документацию.

(a) The `!!` operator

Я узнал, что существует особый оператор, который помогает нам обрабатывать и избегать проблемы с `Null Pointer Exception`. Это оператор `!!`. Если мы хотим получить ненулевой тип, или получить extension то мы можем использовать его. В примере по ссылке можно увидеть: `val l = b!!.length`. Это бывает очень полезно, когда мы хотим проверить ввод пользователя

(b) "Safe" (nullable) cast operator

Стоит сказать, что после некоторых типов в Kotlin мы можем написать символ вопроса (`'?'`) этот символ помогает нам понять, лежит ли тип, или `null`. Если этот оператор не присовить, будет ошибка компиляции. Но есть оператор безопасного приведения типов.

```
val x: String? = y as? String
```

В таком случае результат nullable

(c) `noinline`

В языке Kotlin есть возможность использования встроенных лямбд в инлайн функции, то у нас есть такая возможность, написав перед ними прототип слово `noinline` без использования никаких других функций.

(d) Data classes

Удобно, что есть синтаксический сахар над обычными классами, вместо того, чтобы писать полностью свой класс можно использовать их и автоматически будут написаны некоторые функции: `equals+hashCode+toString()`

(e) Unary operations

Вместо унарных операторов можно указывать конкретные методы. если на плюсах нужно указывать ключевое слово, то теперь можн опереопредять их с помощью имени метода, вот пример: `operator fun Point.unaryMinus() = Point(-x, -y)`

4. Для описания конечного автомата будем использовать русский язык (он является естественным языком). Сначала мы зададим все вершины, причем самой первой записывается начальная вершина, потом все вершины остальные. Просто записываем имя вершины. Если вершина терминальная, то справа от нее пишем терминальная. Все вершины кладем внутри блока ВЕРШИНЫ НАЧАЛИСЬ *вершины* ВЕРШИНЫ КОНЧИЛИСЬ. Теперь зададим все ребра в блоке РЕБРА НАЧАЛИСЬ *все ребра* РЕБРА КОНЧИЛИСЬ.

Записываем ребро так:

'из какой вершины', 'в какую вершину', {'перечисляем по каким символам'};

Записываем вершину так: 'имя', 'является ли вершина терминальной ДА/НЕТ';

Примеры:

Является ли строка строкой ab (алфавит a, b, c)

ВЕРШИНЫ НАЧАЛИСЬ

q_0 , НЕТ;

q_1 , НЕТ;

q_2 , ДА;

\perp , НЕТ;

ВЕРШИНЫ КОНЧИЛИСЬ

РЕБРА НАЧАЛИСЬ

$q_0, q_1, \{a\}$;

$q_0, \perp, \{b\}$;

$q_1, q_2, \{b\}$;

$q_1, \perp, \{a\}$;

$q_2, \perp, \{a, b\}$;

$\perp, \perp, \{a, b\}$;

РЕБРА КОНЧИЛИСЬ

Является ли число корректным (без ведущих нулей) (алфавит 0..9)

ВЕРШИНЫ НАЧАЛИСЬ

q_0 , НЕТ;

q_1 , ДА;

q_2 , ДА;

\perp , НЕТ;

ВЕРШИНЫ КОНЧИЛИСЬ

РЕБРА НАЧАЛИСЬ

$q_0, q_1, \{1..9\}$;

$q_0, q_2, \{0\}$;

$q_1, q_1, \{0..9\}$;

$q_2, \perp, \{0..9\}$;

$\perp, \perp, \{0..9\}$;

РЕБРА КОНЧИЛИСЬ

Заканчивается ли число на 1 (алфавит 0, 1)

ВЕРШИНЫ НАЧАЛИСЬ

q_2 , НЕТ;

q_0 , НЕТ;

q_1 , ДА;

ВЕРШИНЫ КОНЧИЛИСЬ

РЕБРА НАЧАЛИСЬ

$q_2, q_1, \{1\}$;

$q_2, q_0, \{0\}$;

$q_1, q_0, \{0\}$;

$q_1, q_1, \{1\}$;

$q_0, q_1, \{1\}$;

$q_0, q_0, \{0\}$;

РЕБРА КОНЧИЛИСЬ

5. я делал подсветку кода в Sublime. Для того, чтобы применить ее, зайдите в `/.config/sublime-text/Package/User` и в эту папку закиньте файл `My_syntax.sublime – syntax`. Потом создайте любой файл и примените мою подсветку `View->Syntax->My_syntax.sublime – syntax`. В комментариях можно посмотреть что я подсвечивал и почему.