Kyle Berkoski

Formal Languages and Computability

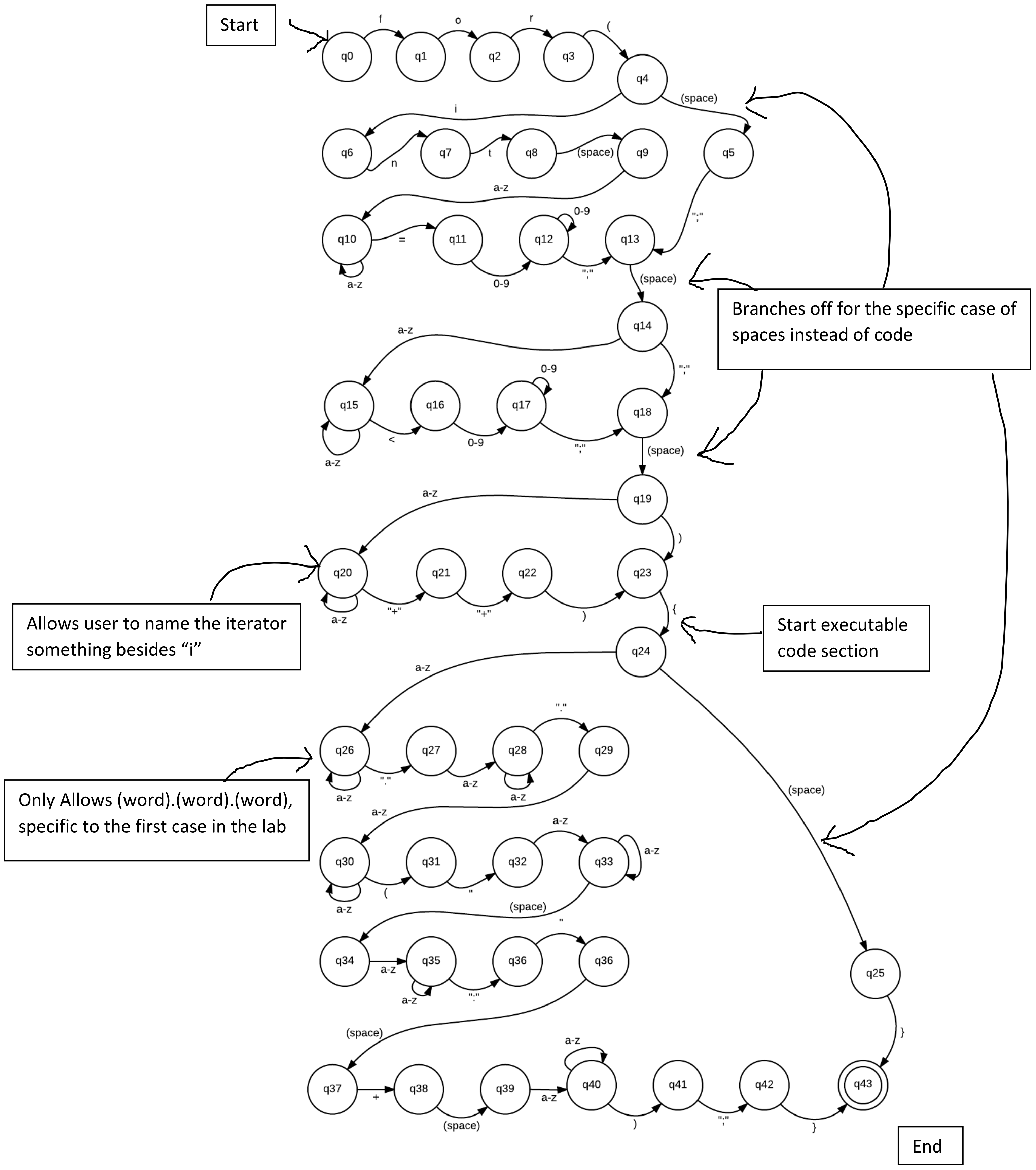
1 February, 2016

Essay 1

One of the main problems of creating a DFA diagram that encompasses any java function is not branching off for each individual scenario. This is a problem because in most cases there are infinite scenarios that are possible. One way to get around this in diagraming two specific scenarios (one with no code, and one with actual java code that will compile and run) is to branch off two different cases, one with the spaces, and one that branched off for the scenarios that contain code. In a for loop, typically the variable used to keep track of the iteration is “i,” however by circling around after the first letter as many times as needed, the user will be able to name the variable anything as long as it has at least one letter. Another problem is when the diagram gets to the code that the for loop will execute it will only handle (some word).(someword).(someword), when theoretically there could be none, or as many as the user needs. These kinds of diagrams require critical and abstract thinking to not make them infinitely big. These diagrams are able to branch off and have different outcomes, they are not required to be strictly linear. It is also very possible (but also very difficult) to do other functions in Java. One could easily map out a diagram for a while loop, an if statement, and even a void.

Essay 2

If one were to implement this kind of list in actual Java code, the most similar way of the diagram would be to build a linked list. A linked list is a linear collection of elements, similar to an array. The difference is that each element of the linked list has a pointer to another element in the list. Much like the arrows in the diagrams, the pointer dictates what comes after the initial start. There are also methods that come with a linked list that can be applied to actually build a linked list. Void add(int index, Object element) for example, will add whatever object to whatever index is required. There is also Boolean add(Object o) which adds the element to the end of the list. Void addFirst(Object o) adds an element to, surprise, the beginning of the list. There are various other commands that can be used in a linked list to help add (And even remove) elements to/from the list. A string has many different definitions depending on the programming language, however, in Java a string is a sequence of characters. A string is also classified as an object. Typically in Java, a string always starts with an open quotation mark. In order to implement a string validator using DFA, the very first input of the string must be an open quotation mark, otherwise it will not classify as a string.


Start
Branches off for the specific case of spaces instead of code
Allows user to name the iterator something besides “i”
Start executable code section
Only Allows (word).(word).(word), specific to the first case in the lab
End
q0
q1
q2
q3
q4
q5
q6
q7
q8
q9
q10
q11
q12
q13
q14
q15
q16
q17
q18
q19
q20
q21
q22
q23
q24
q25
q26
q27
q28
q29
q30
q31
q32
q33
q34
q35
q36
q36
q37
q38
q39
q40
q41
q42
q43
f
o
r
(
(space)
i
n
t
a-z
=
0-9
";"
"+"
)
{
}
"."
"
<
+