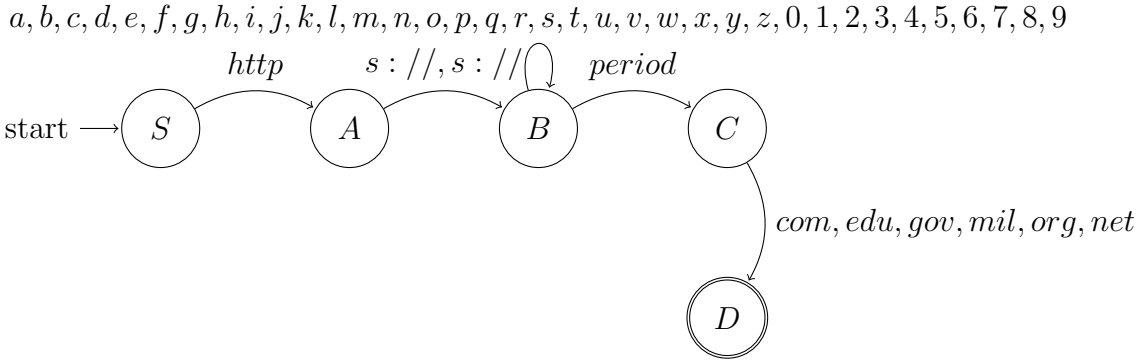
Part 1:

My langue is a representation of URLS using HTTP or HTTPS, and using the top level domains of com, edu, gov, mil, org, and net. It does not support multiple top level domains in one, nor subdomains. Matching URL's is a common important task, and while this language is very simple, it shows how to go about matching URL's. I only support the top level domains that I do because they are the original ones plus net. If I where to try to include every single top level domain my language would be massive. I could probably write code to automatically generate the language, but I believe that sort of thing is out of scope for the assignment. The language is pretty simple, first it gets http, then it matches for both https and regular http. Then at B it just matches everything in the language except for period, which goes to C, which matches the top level domain.

I use period for "." for readability
Part 2:
The language has an alphabet of a-z, all integers, :, and period.
The grammer is
S → httpA
A → s://B|://B
B → periodC|aB|bB|cB|dB|eB|fB|gB|hB|iB|jB|kB|lB|mB|nB|oB|pB|qB|rB|sB|tB|uB|vB|wB|xB|yB|zB|0B|1B|2B|3B|4B|5B|6B|7B|8B|9B
C → com|edu|gov|mil|org|net

which should be equivelent to the regex
http(s://|://)[a-z0-9]*?\.(com|edu|gov|mil|org|net)
My regex syntax I am using is () for grouping | for or, * for zero or more, ? for as little as possible, \. for period because period by itself in most engines means any character, and [a-z0-9] means anything that is a lower case letter or number.
Part 3:
Here is a automaton of my machine:



Part 4:
Note that I wrote the program before I wrote this.
My structure is a series of states, with defend transitions. These transitions contain what they consume and what state they move too. This is inputted into the dfa structure that controls and processes everything. Here are the states in the format NAME, [TRANSITIONS] where TRANSITIONS are (RESULT STATE, CONSUME) and ends with + for end state. B has transitions for everything in the language except : and ., for simplicity I am just using ALL to represent this.
S is the start state and just takes in http
S, [(A,http)]

A is used so we can support https and http.
A, [(B,s://), (B,://)]
This consumes the rest of the url up to the period, at which point we move to C, which handles the TLD
B, [(C, .),(B,ALL)]
This takes the TLD and moves the the accepting state
C, [(D,com), (D,edu), (D,gov), (D, mil), (D, org), (D, net)]
We are not handling subdomains so we are done
D, []+
Part 5:
The code for this is in the included project folder. To run just enter it and do python3 main.py, or py main.py depending on your OS. It will ask for an inputted string and return the path taken if it was successful, and tell you if it was not. It supports python 3.10+, which are all the ones still in support. Python 3.9 does not work due to use of type hints, but its support was dropped in october.
The output format is,
Accepted: [(STATE, CONSUMED)]
and
Rejected