Using the Lock-down Browser

You will have to answer each question using just typing Here are examples.

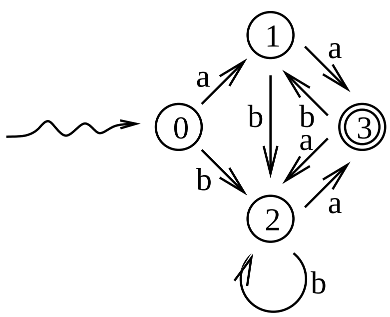
I. Design an NFA which accepts the language of all strings over a, b which have three consecutive a's.

Your answer:

States are 0, 1, 2, 3 where 3 is the only final state.

```
delta(0,a) = 0,1  delta(0,b) = 0
delta(1,a) = 2    delta(1,b) = 0
delta(2,a) = 3    delta(2,b) = 0
delta(3,a) = 3    delta(3,b) = 3
```

II. Give a minimal DFA equivalent to the following DFA:



Your answer: States 1 and 2 are equivalent. the minimal DFA has states 0, 1/2, 3, where 3 is the only final state. The transition function of the minimal DFA is:

```
delta(0,a) = 1/2  delta(0,b) = 1/2
delta(1/2,a) = 3  delta(1/2,b) = 1/2
delta(3,a) = 1/2  delta(3,b) = 1/2
```

III. Give a regular expression for the language of all strings over $\{a, b\}$ with three consecutive a's.
Your answer: `(a+b)*aaa(a+b)*`

IV. Let $L = \{a^n b^n \mid n \geq 0\}$. Design a PDA which accepts L.

Your answer:

```
Three states, 0, 1, and 2  2 is the only final state.  The bottom of stack
symbol is z.
There is a self-loop at state 0.  The labels on that loop are: z/a/az, a/a/aa.
There is an arc from 0 to 1.  The labels on that arc are: z/lambda/z,a/lambda/a.
There is a self-loop at state 1.  The label on that loop is: a/b/lambda.
There is an arc from 1 to 2.  The label on that arc is z/lambda/lambda.
```

V. Consider the following CNF grammar:
$S \to AB$
$A \to a$
$B \to b$
Use the CYK algorithm to prove that ab is in the language generated by the grammar.

Your answer:

```
V[1,1] = A, V[2,2] = B, V[1,2] = S
```

Therefore ab is generated by the grammar.

VI. Write exponents using the carat symbol. That is, a^2 should be written `a^2`

Answers to Practice Exam for February 18, 2021

1. True or False. T = true, F = false, and O = open, meaning that the answer is not known science at this time. In the questions below, $\mathcal{P}$ and $\mathcal{NP}$ denote $\mathcal{P}$-TIME and $\mathcal{NP}$-TIME, respectively.

   (i) **F** Let L be the language over $\{a, b, c\}$ consisting of all strings which have more a's than b's and more b's than c's. There is some PDA that accepts L.

   (ii) **T** The language $\{a^n b^n \mid n \geq 0\}$ is context-free.

   (iii) **F** The language $\{a^n b^n c^n \mid n \geq 0\}$ is context-free.

   (iv) **T** The language $\left\{a^i b^j c^k \mid j = i + k\right\}$ is context-free.

   (v) **T** The intersection of any three regular languages is regular.

   (vi) **F** The intersection of any two context-free languages is context-free.

   (vii) **F** If L is a language and L^* is regular, then L must be regular.

   (viii) **T** If L is a context-free language over an alphabet with just one symbol, then L is regular.

   (ix) **T** The set of strings that your high school algebra teacher would accept as legitimate expressions is a context-free language.

   (x) **T** Every language accepted by a non-deterministic machine is accepted by some deterministic machine.

   (xi) **F** Every context-free language is generated by some unambigous context-free grammar.

   (xii) **T** The problem of whether a given string is generated by a given context-free grammar is decidable.

   (xiii) **T** If G is a context-free grammar, the question of whether $L(G) = \emptyset$ is decidable.

   (xiv) **F** Every language generated by an unambiguous context-free grammar is accepted by some DPDA.

   (xv) **T** The language $\{a^n b^n c^n d^n \mid n \geq 0\}$ is decidable.

(xvi) **F** Every problem that can be mathematically defined has an algorithmic solution.

That would imply that the halting problem has an algorithmic solution, which would mean the halting problem is decidable, which is false.

(xvii) **O** $\mathcal{P} = \mathcal{NP}$.

(xviii) **O** There exists a polynomial time algorithm which finds the factors of any positive integer, where the input is given as a binary numeral.

(xix) **T** The language consisting of all strings over $\{a, b\}$ which have more a's than b's is context-free.

(xx) **T** Every context-free language is in $\mathcal{P}$.

(xxi) **O** Every context-sensitive language is in $\mathcal{P}$.

The context-sensitive membership problem is in $\mathcal{P}$-SPACE. In fact, it is $\mathcal{P}$-SPACE-complete. It is not known whether $\mathcal{P}$-TIME $=$ $\mathcal{P}$-SPACE.

(xxii) **F** Every language generated by a general grammar is decidable.

(xxiii) **F** The problem of whether two given context-free grammars generate the same language is decidable.

(xxiv) **F** Every bounded function is recursive (that is, computable).

(xxv) **T** Recall that if $\mathcal{L}$ is a class of languages, co-$\mathcal{L}$ is defined to be the class of all languages that are not in $\mathcal{L}$. Let $\mathcal{RE}$ be the class of all recursively enumerable languages. If L is in $\mathcal{RE}$ and also L is in co-$\mathcal{RE}$, then L must be decidable.

(xxvi) **O** If a language L is both $\mathcal{NP}$ and co-$\mathcal{NP}$, L must be $\mathcal{P}$.

(xxvii) **T** Every language is enumerable. That means, either L is finite or there is a one-to-one function from the positive integers to L.

(xxviii) **T** There is a non-recursive function which grows faster than any recursive function. (Recursive function means computable function.)

For example, the "busy beaver" function.

(xxix) **T** There exists a machine[1] that runs forever and outputs the string of decimal digits of π (the well-known ratio of the circumference of a circle to its diameter).

(xxx) **F** For every real number x, there exists a machine that runs forever and outputs the string of decimal digits of x.

There are uncountably many real numbers, but there are only countably many machines.

[1]As always in automata theory, "machine" means abstract machine, a mathematical object whose memory and running time are **not** constrained by the size and lifetime of the known (or unknown) universe, or any other physical laws. A computer program, which can have any length, can be considered to be a machine; by assuming it runs on an abstract machine. If we want to discuss the kind of machine that exists (or could exist) physically, we call it a "physical machine."

(xxxi) **O** There is a polynomial time algorithm which determines whether any two regular expressions are equivalent.

This is beyond what I have lectured on up to this point.

(xxxii) **T** If two regular expressions are equivalent, there is a polynomial time proof that they are equivalent.

This is beyond what I have lectured on up to this point.

(xxxiii) **T** The halting problem is undecidable.

2. Let L be the language consisting of all strings over the binary alphabet whose last three symbols are '`010`.' Design an NFA with four states which accepts L.

Call the four states 0, 1, 2, 3, where 0 is the start state, and 3 is the only final state. The transitions are as follows:

```
delta(0,0) = {0,1}      delta(0,1) = {0}
delta(1,0) = empty set  delta(1,1) = {2}
delta(2,0) = {3}        delta(2,1) = empty set
delta(3,0) = empty set  delta(3,1) = empty set
```

3. Let L be the language consisting of all strings over $\{a, b\}$ which do not contain the substring aab. Write a regular expression for L and draw a minimal DFA which accepts L. (Hint: 3 states.)

```
regular exprssion:  (b+ab+aab)*(lamba+a+aa)
```

The DFA has states 0, 1, 2 where 0 is the start state and 0, 1, and 2 are all final. (There is actually a fourth, dead, state, which is not final.) Each state has a "meaning":
State 0 means that the current string does not end with a.
State 1 means that the current string ends with a, but not with aa.
State 2 means that the current string ends with aa.
The transitions are as follows:

```
delta(0,a) = 1  delta(0,b) = 0
delta(1,a) = 2  delta(1,b) = 0
delta(2,a) = 2  delta(2,b) = dead state
```

4. Consider the NFA shown below.

(i) Write a transition table for a minimal DFA equivalent to that NFA.

```
Let Q = {1,2,3,4}.  The reachable subsets of Q are {1}, {3}, {2,3}, {1,2,3},
{4}, {2,3,4} and {1,4}, the last three of which are final.  The start state is
{1}. The states {3} and {2,3} are equivalent.  The transition table is:
  delta(1,a) = 123      delta(1,b) = 3/23
  delta(3/23,a) = 3/23  delta(3/23,b) = 4
  delta(123,a) = 123    delta(123,b) = 234
  delta(14,a) = 123     delta(14,b) = 234
  delta(234,a) = 3/23   delta(234,b) = 14
```

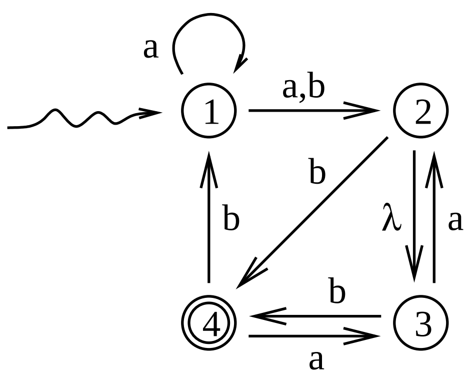
(ii) Write a regular expression which describes the language accepted by that NFA.

```
a*(a+b)a*b(aa*b+ba*(a+b)a*b)*
```

(iii) Give a regular grammar which generates the language accepted by that NFA.

The variables are S, A, B, C, corresponding to states 1, 2, 3, and 4, respectively. The productions are as follows:

```
S -> aS | aA | bA
A -> B | bC
B -> aA | bC
C -> bS | aC | lambda
```



5. Consider the language L generated by the CF grammar given below. $S->wS$
$S->iS$
$S->iSeS$
$S->a$

(i) Give a Chomsky Normal Form grammar for L.

There is more than one CNF grammar. Here is one:

```
S -> WS
W -> w
S -> IS
S -> AB
A -> IS
I -> i
B -> ES
E -> e
```

(ii) Use the CYK algorithm to decide whether $iiwaea \in L$. Show all values of $V_{i,j}$.

```
V11 = {I}
V22 = {I}
V33 = {W}
V44 = {S}
```

```
V55 = {E}
V66 = {S}
V12 = empty set
V23 = empty set
V34 = {S}
V45 = emptyset
V56 = {B}
V13 = emptyset
V24 = {S,A}
V35 = emptyset
V46 = emptyset
V14 = {A,S}
V25 = emptyset
V36 = emptyset
V15 = emptyset
V26 = {S}
V16 = {A,S}
```

The string is a member of L because S is a member of the top set V16.

6. Let L be the set of all algebraic expressions, where the only operations permitted are addition, subtraction, and multiplication, where there are only two variables, x and y and no constants, and where multiplication is indicated by concatenation, as in the expression $x+y(xy+x)$. Parentheses can be used. Write a context-free grammar for L. Your grammar may be ambiguous.

   Here is a simple ambiguous grammar, where E is the start variable:

   ```
   E -> E+E | E-E | EE | (E) | x | y
   ```

   Here is an unambiguous grammar that respects the usual priority of operators:

   ```
   E -> E+T | E-T | T
   T -> TF | F
   F -> (E) | x | y
   ```

7. Design a PDA which accepts the Dyck language. To make it easier to grade, use a and b for left and right parentheses.

   The bottom-of-stack symbol is z. The start state is 0, and there is one other state, 1. Only 1 is final. There is a self loop at 0, with labels:

```
z/a/a
a/a/aa
a/b/lambda
```

   There is an arc from 0 to 1 which has the label:

```
z/lambda/lambda
```