Week 2

1. Countably infinite.
2. No; *L* covers all strings with *n* amount of 1’s followed by the same amount of 2’s. ‘12’ and ‘1122’ are valid, but ‘122’ is not.
   1. No; the string must have a positive number of a’s, b’s and c’s – it cannot be empty.
   2. Yes; there are an equal number of a’s, b’s and c’s.
   3. No; there are not an equal number of a’s and b’s
   4. Yes; there are an equal number of a’s and b’s, and a positive number of c’s – *n* does not need to be equal in *L1* and *L2*
3. *L\** = {*w : w* = ε v ∃*k* > 0 ( ∃*w1, w2, … wk* ∈ *L*( *w = w1w2…wk*) )}
4. {apple, peach, cherry, applepie, peachpie, cherrypie, applecobbler, peachcobbler, cherrycobbler}
5. {*L1* X *L2* = {(peach, pie), (peach, cobbler), (peach, ε), (apple, pie), (apple, cobbler), (apple, ε), (cherry, pie), (cherry, cobbler), (cherry, ε)}}
6. **P***L1* = { Ø, (peach), (apple), (cherry), (peach, apple), (peach, cherry), (apple, cherry), (peach, apple, cherry) }   
   **P***L2* = { Ø, (pie), (cobbler), (ε), (pie, cobbler), (pie, ε), (cobbler, ε), (pie, cobbler, ε) }  
   **P***L* = {*w* : *w* ⊆ *L}***P***L1* ∩ **P***L2* = Ø, therefore the intersection is not empty.
7. *L* = {*w* ∈ {a, b}\* : ∃*x*, *y* (*x* ≠ *y*, |*x*| ≥ 2,|y| ≥ 2, *substr*(*x, w*), *substr*(*y, w*)) }

Week 3

Week 4

1. No; for each NDFSM, there is an equivalent DFSM that can be produced using the *ndfsmtodfsm* algorithm.
   1. True
   2. False
   3. True
   4. 
   5. 
   6. 

Week 5

Week 6

Week 7

Week 8

Week 9

Week 10

Week 11

Week 12

a∈a