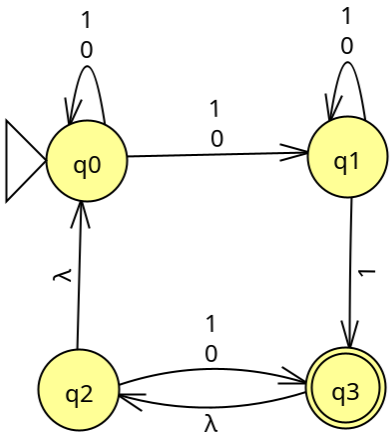
# Assignment 2

Christopher Tait - S3899475

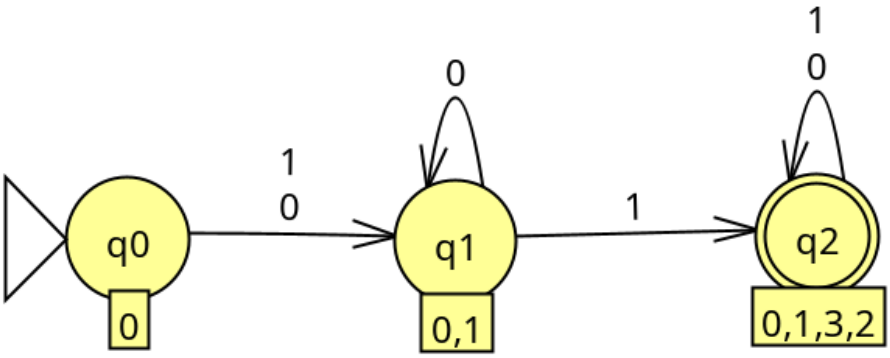
# 2. Nondeterminism

## a)

I constructed this NFA by slowly adding transitions, and making sure that the converted DFA Form after each transition was added did not have many states.

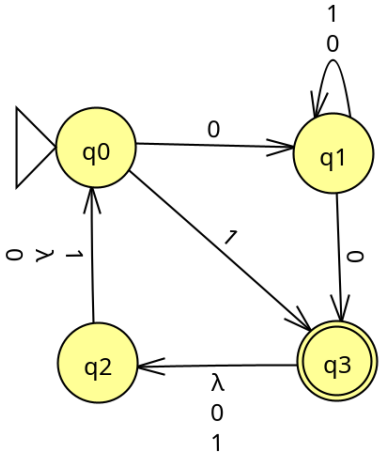



DFA form:

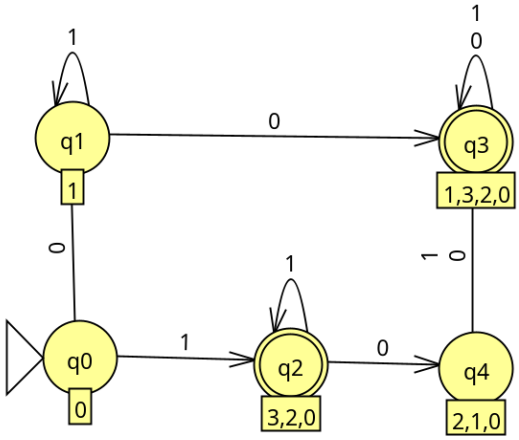



## b)

This NFA was constructed by chance when I was trying to constuct one that had a DFA with 2 or 3 states.

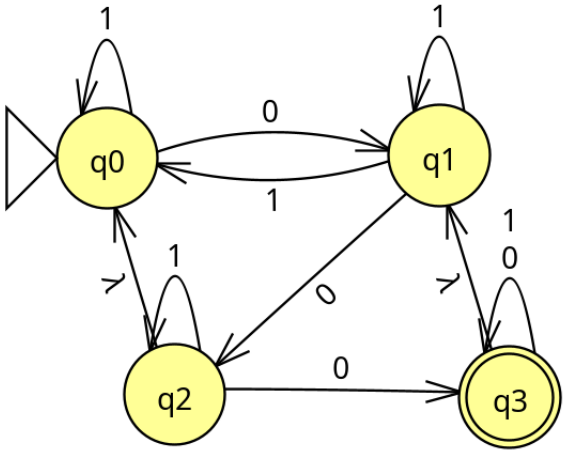


DFA form:

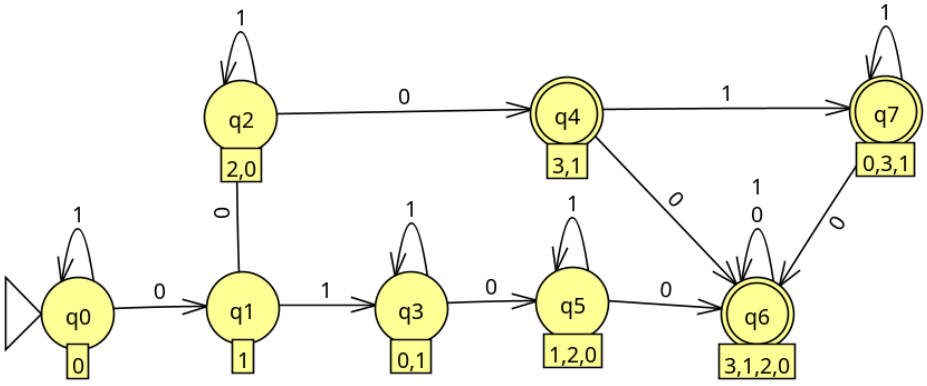



**c)**

To construct this NFA, I first removed the bottom lambda transition and then tested adding transitions to see what would increase the final state cound the most.




DFA Form:




# 3. Pumping Lemma

## a)

**Claim:** The language $L_1 = \{1^i2^j3^{2i}2^j1^i\}$ is not regular.

To prove that this language is not regular, I will first prove that a part of it is not regular.

**Claim:** The language $L_2 = \{1^i2^j3^{2i}\}$ is not regular.

**Proof:**

Assume L is regular, then the Pumping Lemma will apply.
This means there is an integer n ≥ 1 that for some string w ∈ $L_2$, where |w| ≥ n, w=xyz such that:

1. |xy| ≤ n
2. y ! = λ
3. $xy^iz$ ∈ L, for i ≥ 0

When w = $1^n2^n3^{2n}$ by the Pumping Lemma, w=xyz=$1^n2^n3^{2n}$ and |xy| ≤ n
Using y = $1^j$ for some 1 ≤ j ≤ n
Choose i=3 and consider xyyyz = $1^{n+2j}2^n3^{2n}$
By the Pumping Lemma, xyyyz ∈ $L_2$ but this is not the case as 2(n+2j) ! = 2n, so we have a contradiction
Hence $L_2$ is not regular and $L_1$, by extension is not either

## b)

Same as (a) except using w=$1^n3^{2n}$ and i=0

**Claim:** The language $L_1$ = {$1^i2^j3^{2i}2^j1^i$} is not regular.
To prove that this language is not regular, I will first prove that a part of it is not regular.
**Claim:** The language $L_3$ = {$1^i2^j3^{2i}$} is not regular.
**Proof:**
Assume L is regular, then the Pumping Lemma will apply.
This means there is an integer n ≥ 1 that for some string w ∈ $L_3$, where |w| ≥ n, w=xyz such that:

- |xy| ≤ n
- y ! = λ
- $xy^iz$ ∈ L, for i ≥ 0

When w = $1^i2^03^{2i}$ = $1^n3^{2n}$, by the Pumping Lemma, w=xyz=$1^n3^{2n}$ and |xy| ≤ n
Using y = $1^j$ for some 1 ≤ j ≤ n
Choose i=0 and consider xz = $1^{n-j}3^{2n}$
By the Pumping Lemma, xz ∈ $L_2$ but this is not the case as 2(n-j) ! = 2n, so we have a contradiction.
Hence $L_3$ is not regular and $L_1$ by extension is not regular either.

Most steps remain the same as the $2^j$ is not used in the original proof. Only the last few steps are changed, because w is different.

## c)

### Correct Proof

**Claim:** The language L={$1^i2^i\#3^{2i}@2^i1^i$} is not context free
**Proof:** Assume L is context-free. Then the Pumping Lemma applied and so for some n ≥ 1 such that for all w ∈ L such that |w| ≥ n, w=xyzuv where:

- |yzu| ≤ n
- y ! = λ or u ! = λ
- $xy^izu^iv$ ∈ L for all i ≥ 0

Choose w = $1^n2^n\#3^{2n}@2^n1^n$ and so w ∈ L and |w| ≥ n. So by the Pumping Lemma w = xyzuv = $1^n2^n\#3^{2n}@2^n1^n$ and |yzu| ≤ n. This means that y and u can contain at most two of 1, 2 and 3. We will refer to the part of the string before # as zone 1, the part of the string between # and @ as zone 2, and the part of the string after @ as zone 3.
Now if y or u contains # or @, then clearly $xy^2zu^2v$ ∉ L, as this would contain two occurrences of either # or @.
Hence neither u nor v contains either # or @.

Now note that both y and u can only cover at most two of zone 1, zone 2, and zone 3, as |yzu| ≤ n. This means that $xy^2zu^2v \notin L$, as one of the zones will have the same string as w, but the other two will have longer strings, and hence $xy^2zu^2v$ is not of the form $1^i2^i\#3^{2i}@2^i1^i$.
This is a contradiction, and so L2 is not context-free.

## 4. Intracability

For this section I used the website https://tspvis.com/ to simulate the Travelling Salesperson Problem. I used the depth first search algorithm (a brute force algorithm) to find the complete solution. The input data was constructed by placing a point on the capital of each australian state and territory, in order of decreasing population of state. This means that the order of cities added was:

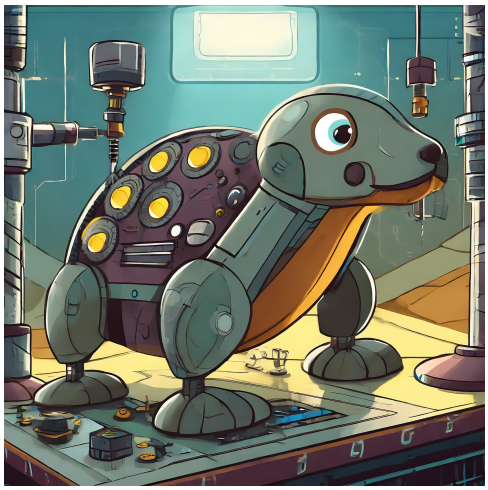
3. Melbourne, Brisbane, Sydney
4. Perth
5. Adelaide
6. Hobart
7. Canberra
8. Alice Springs
9. Norfolk Island

Complete solution: Depth first search

The two estimations I used were the Nearest Neighbour algorithm, which is a simple greedy algorithm, and the Furthests Insetion algorithm. Unfortunately, both algorthims did not take long to complete, even with the maximum number of 200 points. The Nearest Neighbour algorithm still took about 1s and the Furthest Insertion took 3s.

| n | Brute Force | Nearest Neighbour | Furthest Insertion |
|---|---|---|---|
| 3 | 1 | 1 | 1 |
| 4 | 3 | 1 | 1 |
| 5 | 2 | 1 | 1 |
| 6 | 7 | 1 | 1 |
| 7 | 48 | 1 | 1 |
| 8 | 337 | 1 | 1 |
| 9 | 2707 | 1 | 1 |

## 5. Creative Cloud



This image was created using Adobe Firefly with the prompt "A game that includes that platypus, which simulates a turing machine". I used this prompt to try and get something mechanical looking, and slected the final image from the four generated based on the one that had the correct number of legs.

# 6. Get Creative!

For this section I have chosen to extend my Langton's Ant simulator with 'boundaries'. The boundaries essentially make the turning machine that is Langton's Ant have a 2d looping tape to move on. These boundaries can be adusted using the mouse, and allow a lot more variations for different Langton's Ants, that have the same definition (e.g. turn left on a black square, turn right on a white one). The simulation runs in the browser and is now packaged into a single html file that will run on all modern browsers. The simulation can be downloaded here: https://github.com/s3899475/COSC1107-Assignment-2/releases/download/v1/out.html

## Reflection

From this creative project, I have learned of another example of how simple rules can turn into a complex, winding pattern. This phenomenon seems common in the world of mathematics and computing, and leads to many rabbit hole topics. I have never looked into Langton's Ant before, but have been interesting in generating images of fractals such as the mandelbrot set. If I were to do the project again, I would perhaps use pure JavaScript instead of the Nim language, as cross compiling to Javascript can lead to some loss of performance. Another idea would be to implement the Langton's ant using a GUI library, to acheive greater simulation performance.

# 7. The Platypus Game

## a)

| Class | Number | Percentage |
|---|---|---|
| Reachable | 1499 | 75% |
| Unreachable | 176 | 9% |
| None | 325 | 16% |

### Reachable

Time taken for tournament: 1444.152 seconds
Wins: 1042279
Winless Machines: 0

Top 10 Machines:

| Number | Wins | For | Against |
|---|---|---|---|
| 200911315 | 1356 | 88142 | 36299 |
| 150566290 | 1356 | 87515 | 37061 |

| Number | Wins | For | Against |
|---|---|---|---|
| 180142529 | 1340 | 87599 | 38683 |
| 180289115 | 1335 | 78374 | 29264 |
| 157242778 | 1324 | 87490 | 21387 |
| 144071213 | 1319 | 82836 | 24544 |
| 169484218 | 1317 | 91729 | 28405 |
| 196206039 | 1316 | 83546 | 38475 |
| 169624512 | 1311 | 92039 | 28866 |
| 178908893 | 1307 | 80706 | 38185 |

**Unreachable**
Time taken for tournament: 34.296 seconds
Wins: 12335
Winless Machines: 0

Top 10 Machines:

| Number | Wins | For | Against |
|---|---|---|---|
| 176855009 | 165 | 25835 | 15461 |
| 138502054 | 156 | 25750 | 17362 |
| 143680804 | 156 | 25750 | 17362 |
| 159390644 | 156 | 25750 | 17362 |
| 163642809 | 156 | 25750 | 17362 |
| 174154661 | 156 | 25750 | 17362 |
| 186661941 | 156 | 25750 | 17362 |
| 243414685 | 156 | 23728 | 16920 |
| 197303116 | 155 | 23408 | 16594 |
| 150037820 | 151 | 21358 | 16441 |

**None**
Time taken for tournament: 120.339 seconds
Wins: 47922
Winless Machines: 1

Top 10 Machines:

| Number | Wins | For | Against |
|---|---|---|---|
| 179585518 | 299 | 49595 | 31093 |
| 185106832 | 295 | 46523 | 32289 |
| 179053738 | 293 | 48075 | 31494 |
| 176990952 | 292 | 45640 | 30095 |
| 176433994 | 291 | 46440 | 29059 |
| 145036604 | 290 | 47145 | 31417 |
| 184568622 | 290 | 47145 | 31417 |
| 172660075 | 290 | 44395 | 30246 |

| Number | Wins | For | Against |
|---|---|---|---|
| 196346980 | 287 | 48435 | 30118 |
| 134269205 | 285 | 45640 | 32030 |

## b)

For this task I have chosen to do an alternative tournament with the 'Green', 'Animal' and 'Check' combination of options. This tournament took 2686.145 seconds to run. The top 10 machines are as follows:

| Number | Wins | For | Against |
|---|---|---|---|
| 162814060 | 1995 | 160047 | 77183 |
| 179585518 | 1992 | 236395 | 119365 |
| 179457744 | 1991 | 176197 | 86780 |
| 172660075 | 1977 | 217505 | 113988 |
| 36324288 | 1966 | 227044 | 117949 |
| 11880700 | 1952 | 265714 | 133586 |
| 42748637 | 1945 | 194674 | 103927 |
| 179053738 | 1936 | 200424 | 118886 |
| 180289115 | 1936 | 99903 | 49022 |
| 42613953 | 1927 | 204141 | 112958 |

As seen above, the top 10 machines are completely different from the tournament run using the 'tree', 'tiebreaker' and 'terminate' options.

## c)

If the tournament were to be run again, I would reccommend that:

- The tournament be run in a bracket based fashion, where each student's top 10 machines would face off against another student's machines, and the winner would enter in to the next round.
- Each player has a set number of lives, perhaps 3. When a player has a platypus on a green cell, they lose a life but the cell gets reverted to its initial state. The game ends when one player's lives are reduced to 0.