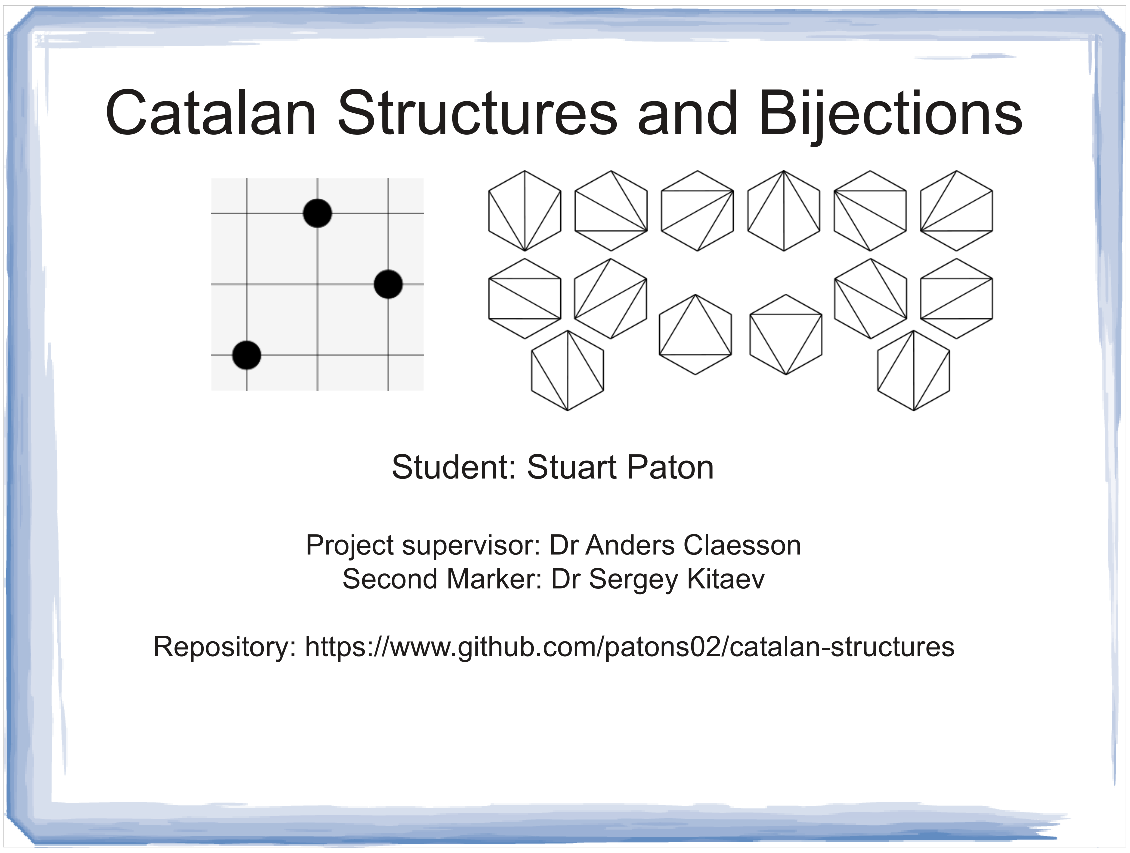
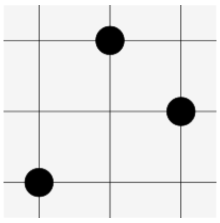
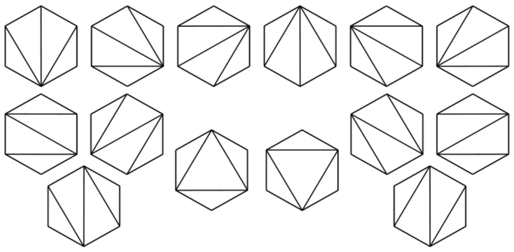


# Catalan Structures and Bijections





Student: Stuart Paton

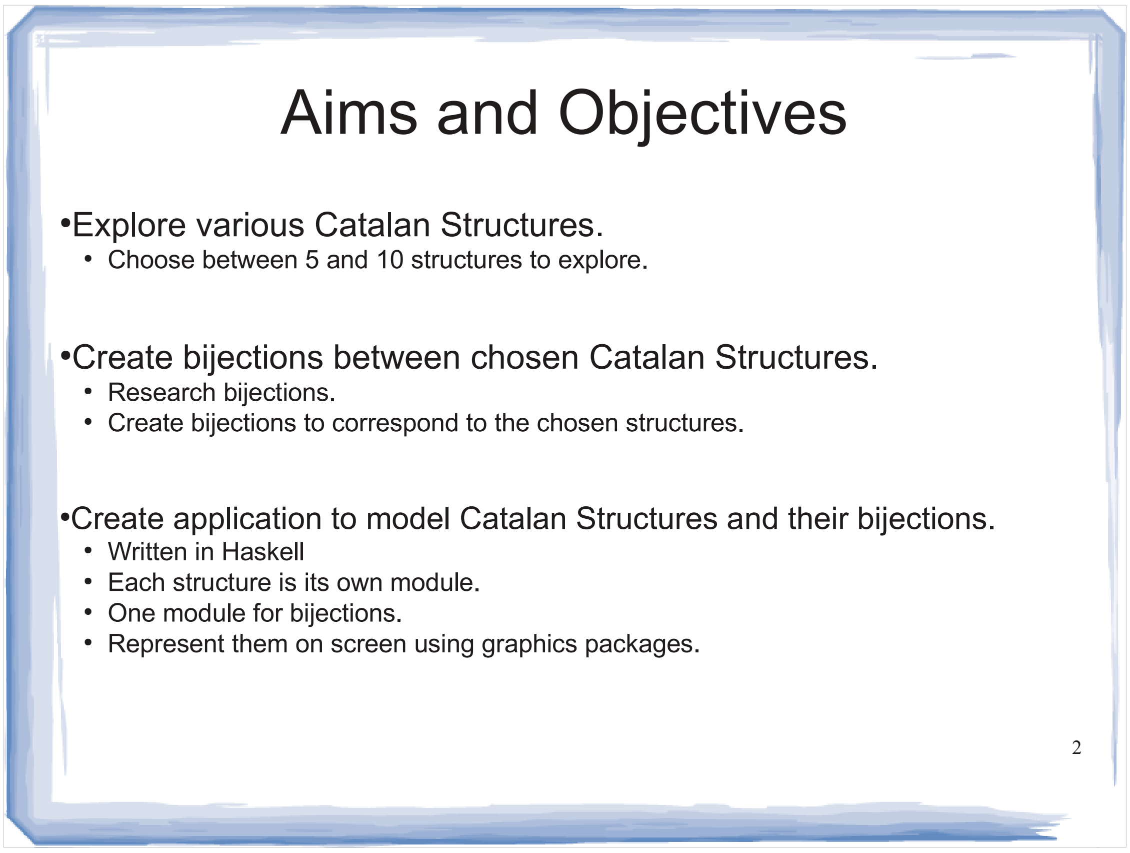
Project supervisor: Dr Anders Claesson
Second Marker: Dr Sergey Kitaev

Repository: https://www.github.com/patons02/catalan-structures

Good afternoon I'm Stuart Paton and this is my project poster presentation on Catalan Structures and their bijections.

My project supervisor is Dr Anders Claesson and my second marker is Dr Sergey Kitaev.

All my project information and code is being stored on github at https://www.github.com/patons02/catalan-structures



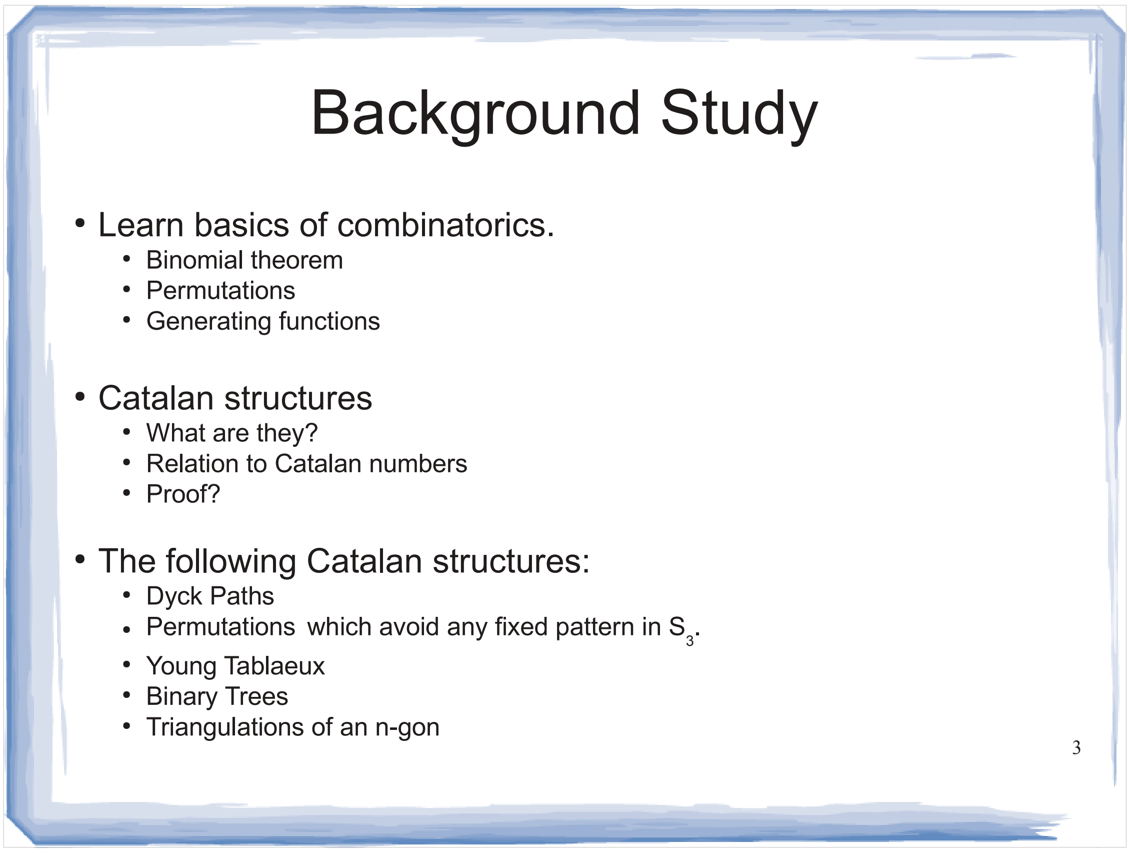
## Aims and Objectives

- Explore various Catalan Structures.
  - Choose between 5 and 10 structures to explore.
- Create bijections between chosen Catalan Structures.
  - Research bijections.
  - Create bijections to correspond to the chosen structures.
- Create application to model Catalan Structures and their bijections.
  - Written in Haskell
  - Each structure is its own module.
  - One module for bijections.
  - Represent them on screen using graphics packages.

2

For my project I have three key aims which I shall explain:
The first aim is to explore different Catalan structures, of which I shall choose between 5 and 10 of them depending on time constraints.
The structures I have chosen will be explained in the next slide.

My second aim is to create bijections between my chosen Catalan Structures. In order to succeed with this I must firstly research bijections which have currently been found, and then create bijections between my chosen structures. Currently I have a bijection from Stack sortable permutations to Dyck paths known as the standard bijection which I shall discuss later.
I have been using the paper: Classification of 312- and 132-avoiding permutations by Anders Claesson and Sergey Kitaev for bijections between permutations.
Lastly my third aim is to create an application to model the chosen Catalan structures and their bijections.
This will be done in a modular fashion with one module for each structure and a separate model to hold all of the bijections.
Finally I will represent this on screen using the Haskell graphics package Diagrams.



## Background Study

- Learn basics of combinatorics.
  - Binomial theorem
  - Permutations
  - Generating functions
- Catalan structures
  - What are they?
  - Relation to Catalan numbers
  - Proof?
- The following Catalan structures:
  - Dyck Paths
  - Permutations which avoid any fixed pattern in $S_3$.
  - Young Tablaeux
  - Binary Trees
  - Triangulations of an n-gon

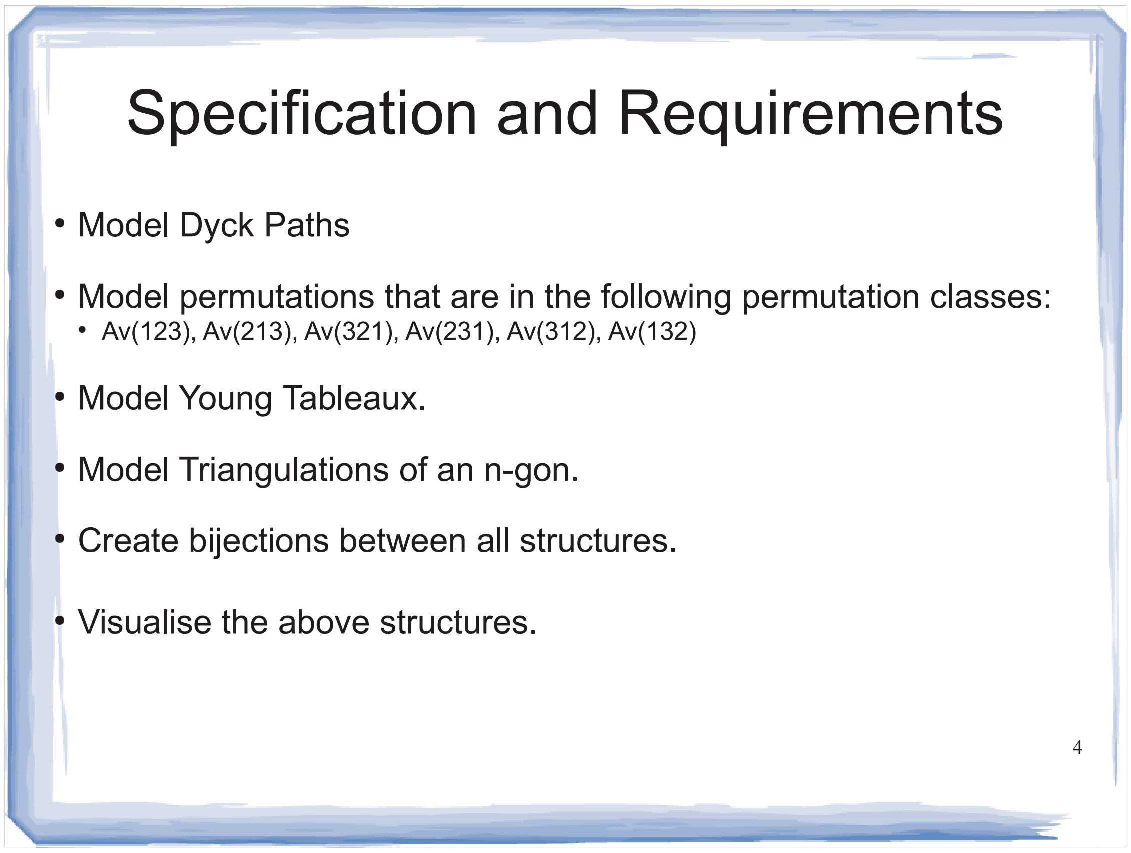
3

Before fully starting the project I had to do some background study into the area of combinatorics. To do this I bought a text book and went through it and covered the basic areas such as the binomial theorem, permutaitons and generating functions.

Next I started to explore what exactly a Catalan Structure is. When you create the generating function for a structure, by taking its formal power series coefficients you get the catalan sequence of numbers.

Proof (on sheet...)

Speak about structures.



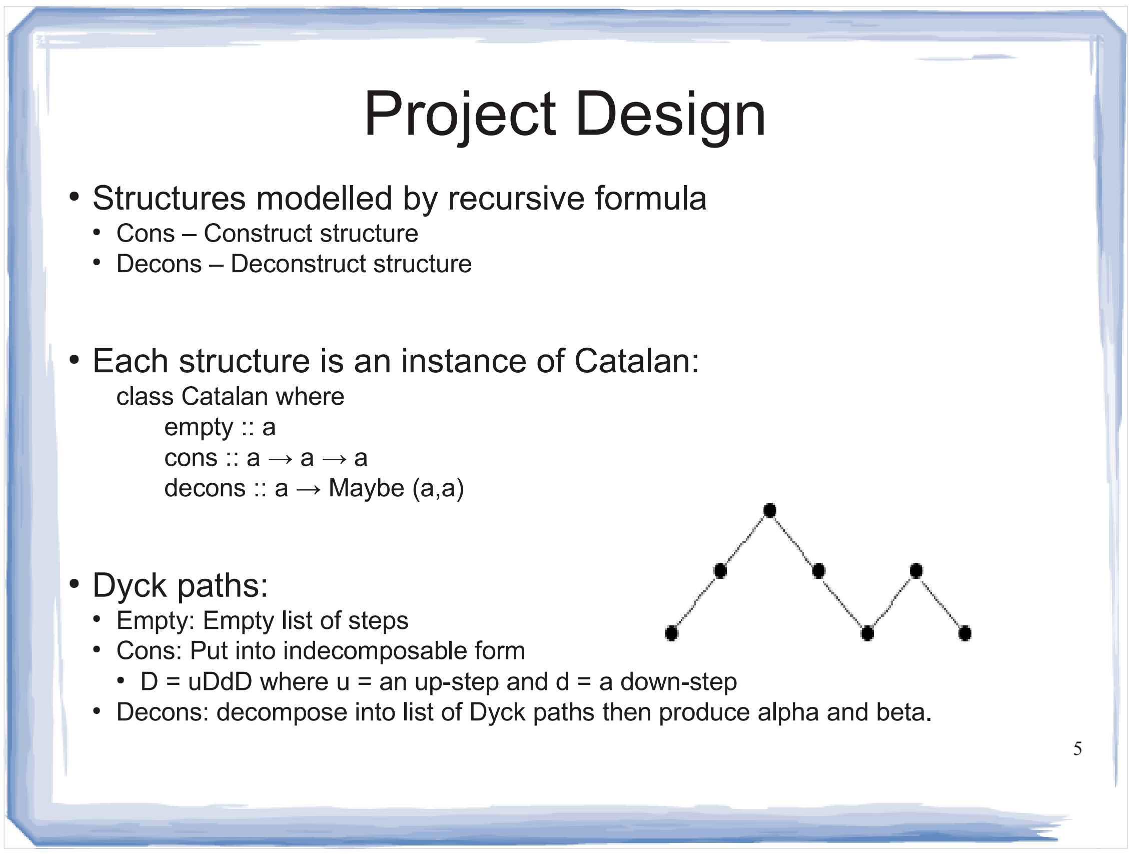
## Specification and Requirements

- Model Dyck Paths
- Model permutations that are in the following permutation classes:
  - Av(123), Av(213), Av(321), Av(231), Av(312), Av(132)
- Model Young Tableaux.
- Model Triangulations of an n-gon.
- Create bijections between all structures.
- Visualise the above structures.

Speak about:

Encoding of each structure, and then that it'll be done for each in Haskell and finally that it'll be visualised.

Dyck paths in Haskell
Permutations in Haskell so far
Fact they'll be visualised using Diagrams.

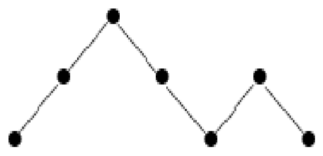


## Project Design

- Structures modelled by recursive formula
  - Cons – Construct structure
  - Decons – Deconstruct structure

- Each structure is an instance of Catalan:

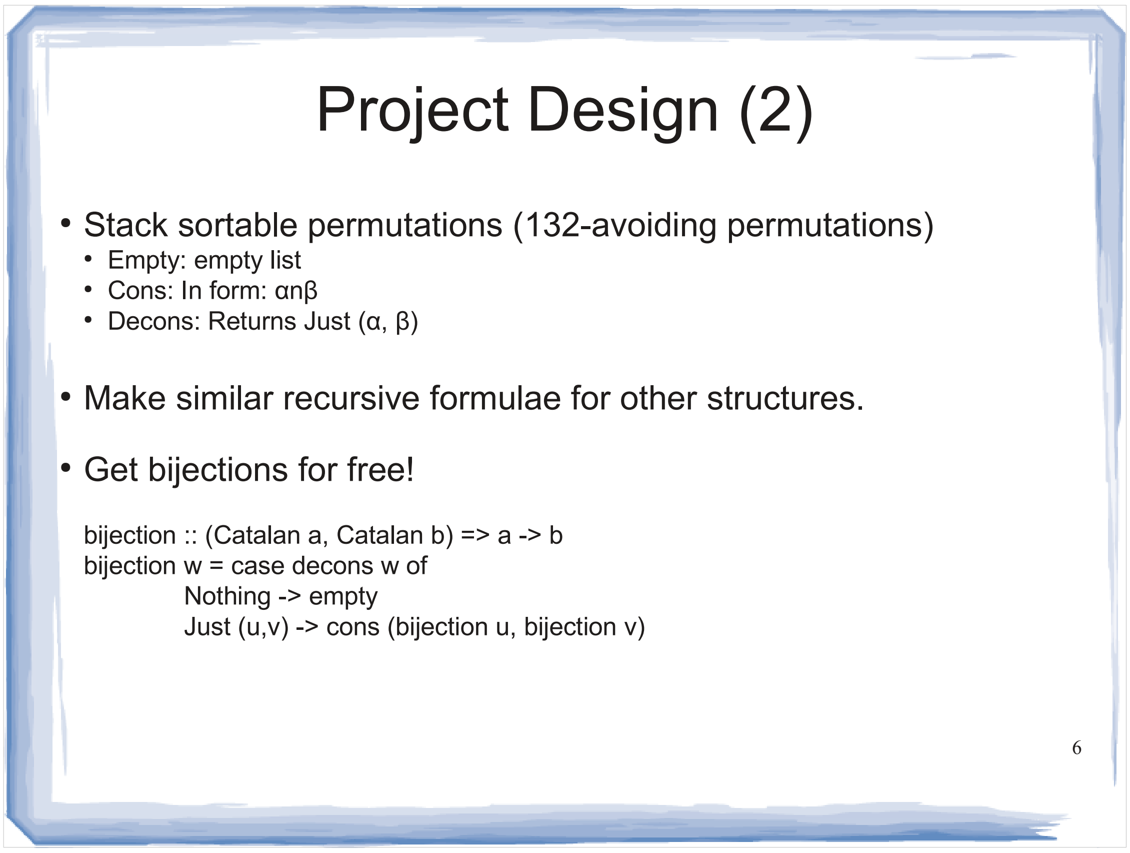
```
class Catalan where
      empty :: a
      cons :: a → a → a
      decons :: a → Maybe (a,a)
```



- Dyck paths:
  - Empty: Empty list of steps
  - Cons: Put into indecomposable form
    - D = uDdD where u = an up-step and d = a down-step
  - Decons: decompose into list of Dyck paths then produce alpha and beta.

Speak about how to model I will make a recursive mathematical model, and implement in Haskell.
Speak about Catalan typeclass and how each structure is an instance of it (Shown on slide.)

Dyck Path: indecomposable form = U + alpha + D
For the decons part of Dyck paths speak about how it works exactly by getting the height of each element of the Dyck path by their partial sums and how it extracts the alpha and disregards the U and D surrounding alpha, and then also extracts beta.



## Project Design (2)

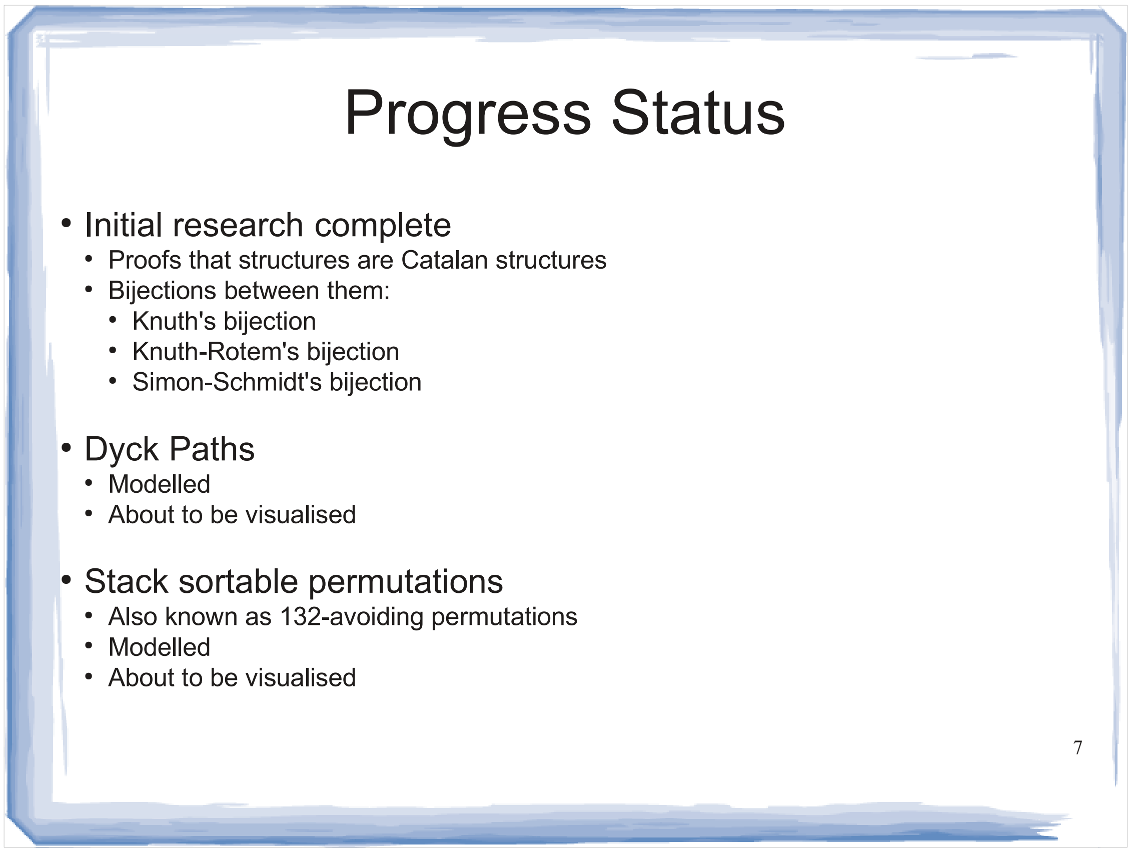
- Stack sortable permutations (132-avoiding permutations)
  - Empty: empty list
  - Cons: In form: αnβ
  - Decons: Returns Just (α, β)
- Make similar recursive formulae for other structures.
- Get bijections for free!

```
bijection :: (Catalan a, Catalan b) => a -> b
bijection w = case decons w of
          Nothing -> empty
          Just (u,v) -> cons (bijection u, bijection v)
```

6

SSP: Cons: make alpha n beta form
Decons: just get alpha and beta, n is just the length of both + 1.
Speak about expansion for other permutations and structures.

Speak about getting the bijection for free and how we do, instances of Catalan are defined in the type line so we can then run cons and decons on them both.



## Progress Status

- Initial research complete
  - Proofs that structures are Catalan structures
  - Bijections between them:
    - Knuth's bijection
    - Knuth-Rotem's bijection
    - Simon-Schmidt's bijection
- Dyck Paths
  - Modelled
  - About to be visualised
- Stack sortable permutations
  - Also known as 132-avoiding permutations
  - Modelled
  - About to be visualised

Speak about overall progress so far:

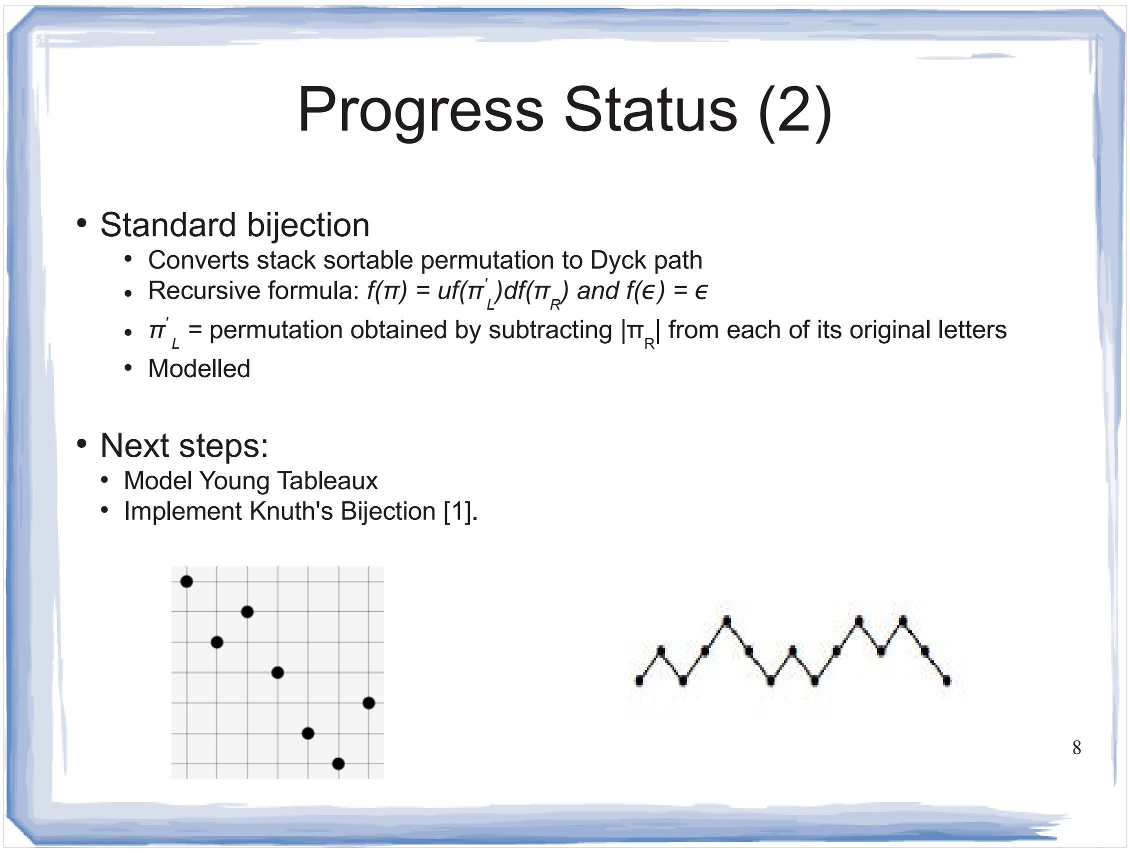
Mention proofs of each structure and that I've written notes on each of them so far in the form of a paper.

Mention the bijections between Dyck Paths and the permuations of Sym(3).

State how the models of Dyck Paths corresponds with my design.

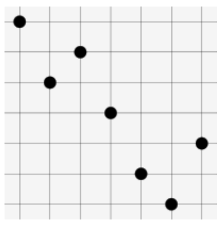
State how the models of permutations corresponde with my design

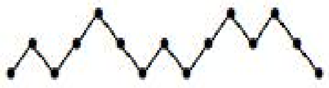
Speak a bit about how they're visualised.



## Progress Status (2)

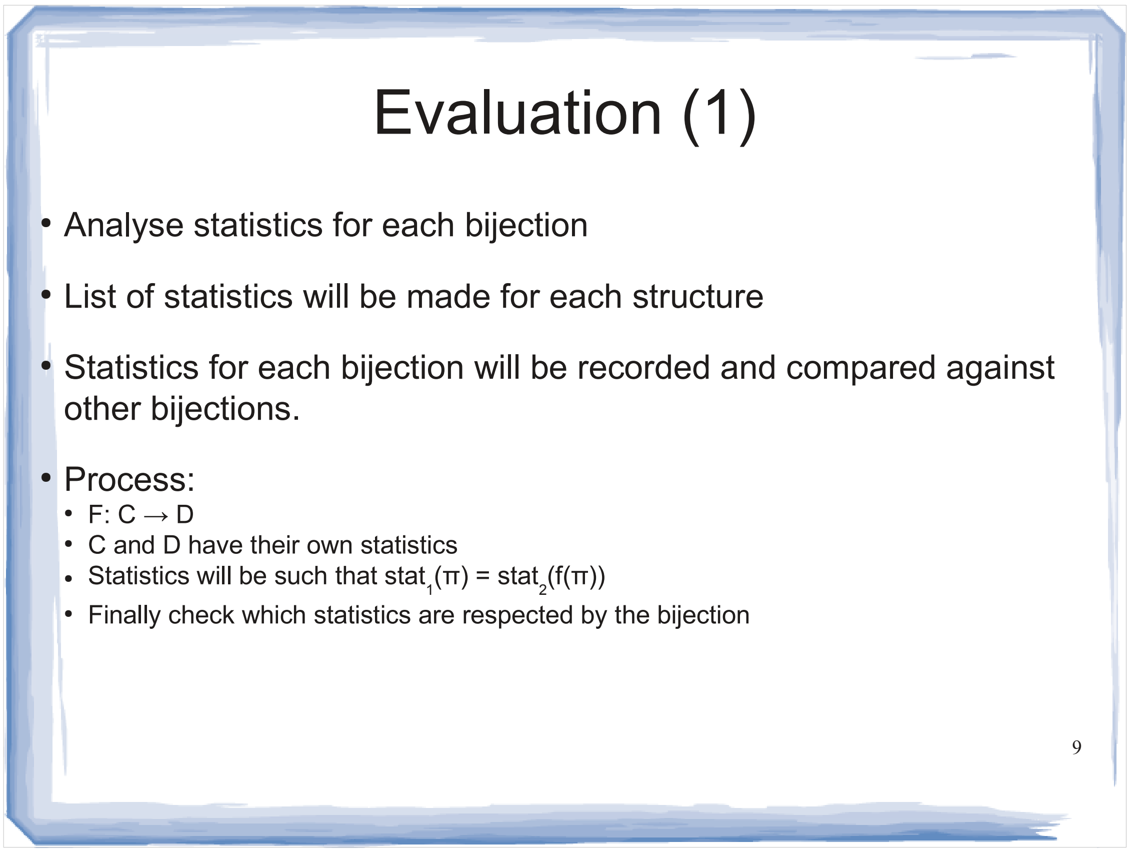
- Standard bijection
  - Converts stack sortable permutation to Dyck path
  - Recursive formula: $f(\pi) = uf(\pi'_L)df(\pi_R)$ *and* $f(\epsilon) = \epsilon$
  - $\pi'_L$ = permutation obtained by subtracting $|\pi_R|$ from each of its original letters
  - Modelled
- Next steps:
  - Model Young Tableaux
  - Implement Knuth's Bijection [1].





Speak about the standard bijection and it's implementation.

Speak about how I will model young tableaux and knuths bijection using the Robinson-Schensted-Knuth correspondence.



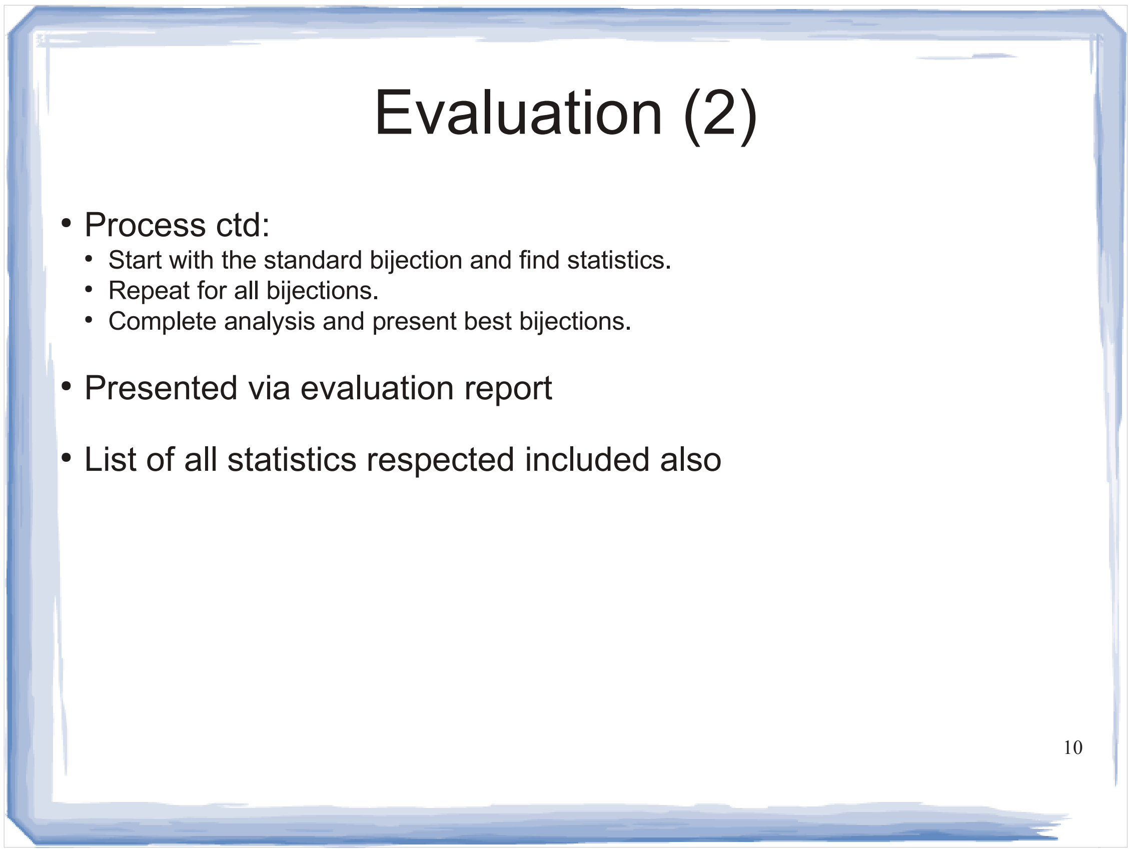
## Evaluation (1)

- Analyse statistics for each bijection
- List of statistics will be made for each structure
- Statistics for each bijection will be recorded and compared against other bijections.
- Process:
  - F: C → D
  - C and D have their own statistics
  - Statistics will be such that $stat_1(\pi) = stat_2(f(\pi))$
  - Finally check which statistics are respected by the bijection

Speak about the base set of permutation statistics and how they will be referenced against.

Speak about how I will start off finding statistics for each bijection.

Mention process of evaluation

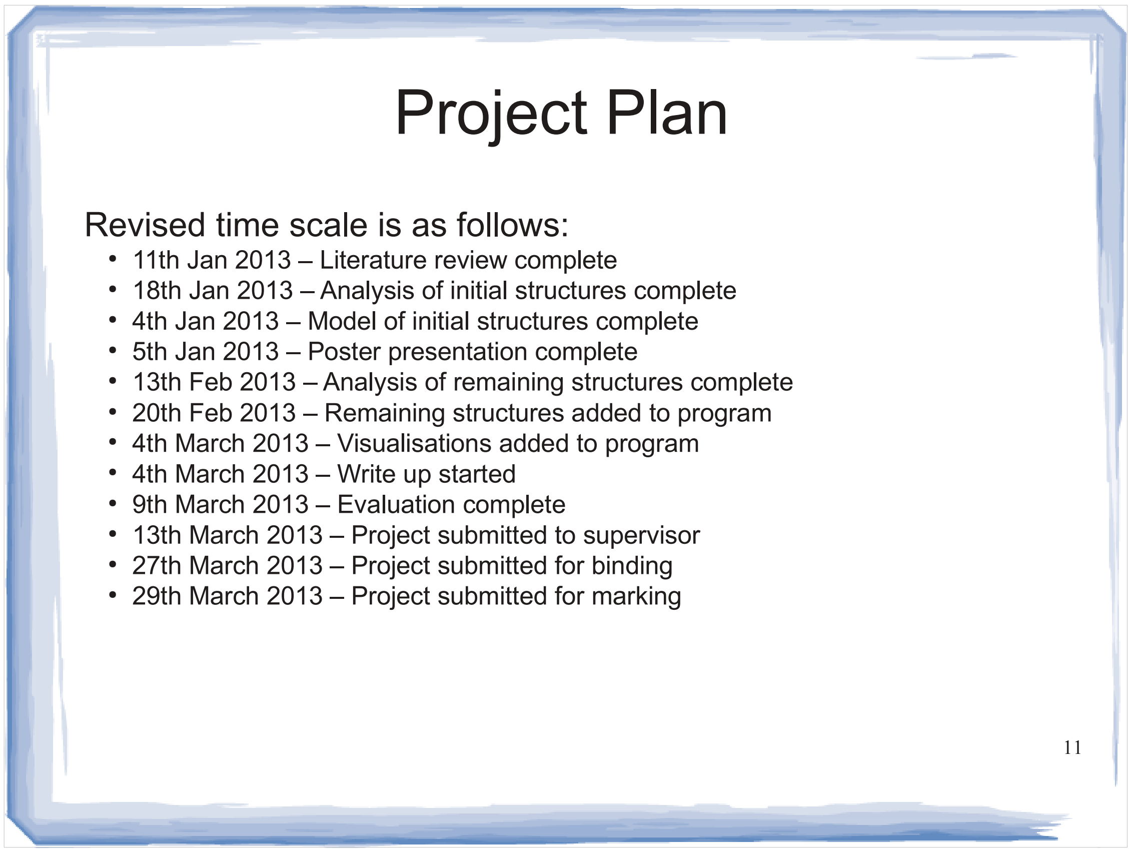


## Evaluation (2)

- Process ctd:
  - Start with the standard bijection and find statistics.
  - Repeat for all bijections.
  - Complete analysis and present best bijections.
- Presented via evaluation report
- List of all statistics respected included also

Speak about ways to presenting the findings.

Speak about base statistics also.



## Project Plan

Revised time scale is as follows:

- 11th Jan 2013 – Literature review complete
- 18th Jan 2013 – Analysis of initial structures complete
- 4th Jan 2013 – Model of initial structures complete
- 5th Jan 2013 – Poster presentation complete
- 13th Feb 2013 – Analysis of remaining structures complete
- 20th Feb 2013 – Remaining structures added to program
- 4th March 2013 – Visualisations added to program
- 4th March 2013 – Write up started
- 9th March 2013 – Evaluation complete
- 13th March 2013 – Project submitted to supervisor
- 27th March 2013 – Project submitted for binding
- 29th March 2013 – Project submitted for marking

Speak of how I've revised my plan from original plan.

## References

[1] – Anders Claesson and Sergey Kitaev, Classification of bijections between 321- and 132-avoiding permutations, 2008.