CIT 5940

Course Introduction

Welcome to CIT 5940!

A course about **data structures** and **software design.**

Programs as Information Processors

Computers are good at reading, writing, and processing batches of information.

- Individual units of data in a system are called **records** (or perhaps **nodes** or **items**).
- Further information is encoded by the **structural relationships** among the records in a system.

Data Structures

Data Structures are systems of structural relationships used to store records. The relationships used for one data structure make certain operations:

- easier or harder to program
- more or less computationally efficient

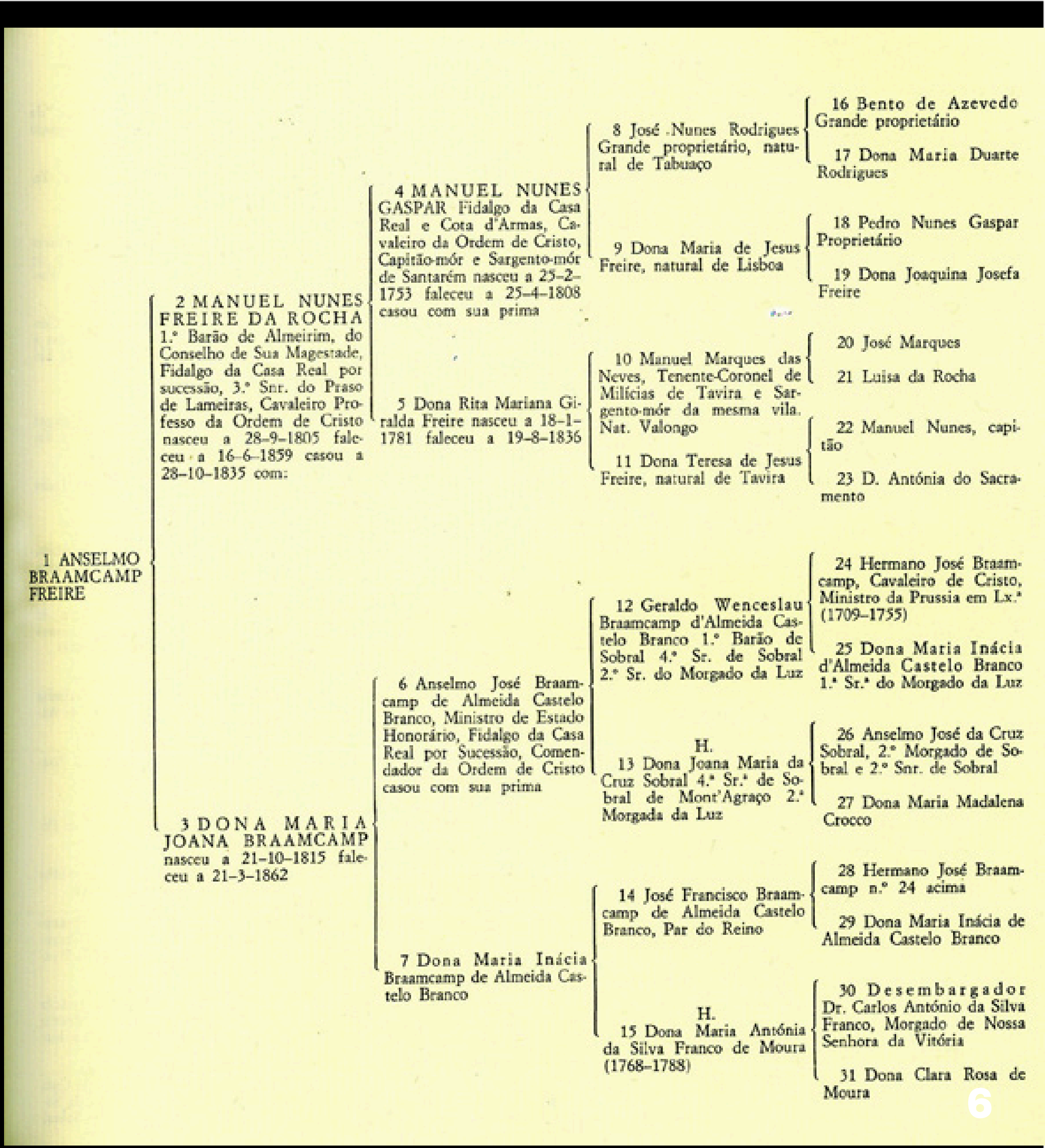
Data Structures & Affordances

The choices of relationships provide different **affordances,** or ways in which users are able use the systems.

- One data structure might make it easier to design an algorithm that finds the oldest or youngest people in a group compared to another
- The same algorithm might work more efficiently when calling the same operation on one data structure vs. another

One Common Data Structure

- What are the records?
- What information is encoded by the structural relationships?
- What affordances does this structure provide you?



1 ANSELMO BRAAMCAMP FREIRE

2 MANUEL NUNES FREIRE DA ROCHA 1.º Barão de Almeirim, do Conselho de Sua Magestade, Fidalgo da Casa Real por sucessão, 3.º Snr. do Praso de Lameiras, Cavaleiro Professo da Ordem de Cristo nasceu a 28-9-1805 faleceu a 16-6-1859 casou a 28-10-1835 com:

3 DONA MARIA JOANA BRAAMCAMP nasceu a 21-10-1815 faleceu a 21-3-1862

4 MANUEL NUNES GASPAR Fidalgo da Casa Real e Cota d'Armas, Cavaleiro da Ordem de Cristo, Capitão-mór e Sargento-mór de Santarém nasceu a 25-2-1753 faleceu a 25-4-1808 casou com sua prima

5 Dona Rita Mariana Giralda Freire nasceu a 18-1-1781 faleceu a 19-8-1836

6 Anselmo José Braamcamp de Almeida Castelo Branco, Ministro de Estado Honorário, Fidalgo da Casa Real por Sucessão, Comendador da Ordem de Cristo casou com sua prima

7 Dona Maria Inácia Braamcamp de Almeida Castelo Branco

8 José .Nunes Rodrigues Grande proprietário, natural de Tabuaço

9 Dona Maria de Jesus Freire, natural de Lisboa

10 Manuel Marques das Neves, Tenente-Coronel de Milícias de Tavira e Sargento-mór da mesma vila. Nat. Valongo

11 Dona Teresa de Jesus Freire, natural de Tavira

12 Geraldo Wenceslau Braamcamp d'Almeida Castelo Branco 1.º Barão de Sobral 4.º Sr. de Sobral 2.º Sr. do Morgado da Luz

H.
13 Dona Joana Maria da Cruz Sobral 4.ª Sr.ª de Sobral de Mont'Agraço 2.ª Morgada da Luz

14 José Francisco Braamcamp de Almeida Castelo Branco, Par do Reino

H.
15 Dona Maria Antónia da Silva Franco de Moura (1768-1788)

16 Bento de Azevedo Grande proprietário

17 Dona Maria Duarte Rodrigues

18 Pedro Nunes Gaspar Proprietário

19 Dona Joaquina Josefa Freire

20 José Marques

21 Luisa da Rocha

22 Manuel Nunes, capitão

23 D. Antónia do Sacramento

24 Hermano José Braamcamp, Cavaleiro de Cristo, Ministro da Prussia em Lx.ª (1709-1755)

25 Dona Maria Inácia d'Almeida Castelo Branco 1.ª Sr.ª do Morgado da Luz

26 Anselmo José da Cruz Sobral, 2.º Morgado de Sobral e 2.º Snr. de Sobral

27 Dona Maria Madalena Crocco

28 Hermano José Braamcamp n.º 24 acima

29 Dona Maria Inácia de Almeida Castelo Branco

30 Desembargador Dr. Carlos António da Silva Franco, Morgado de Nossa Senhora da Vitória

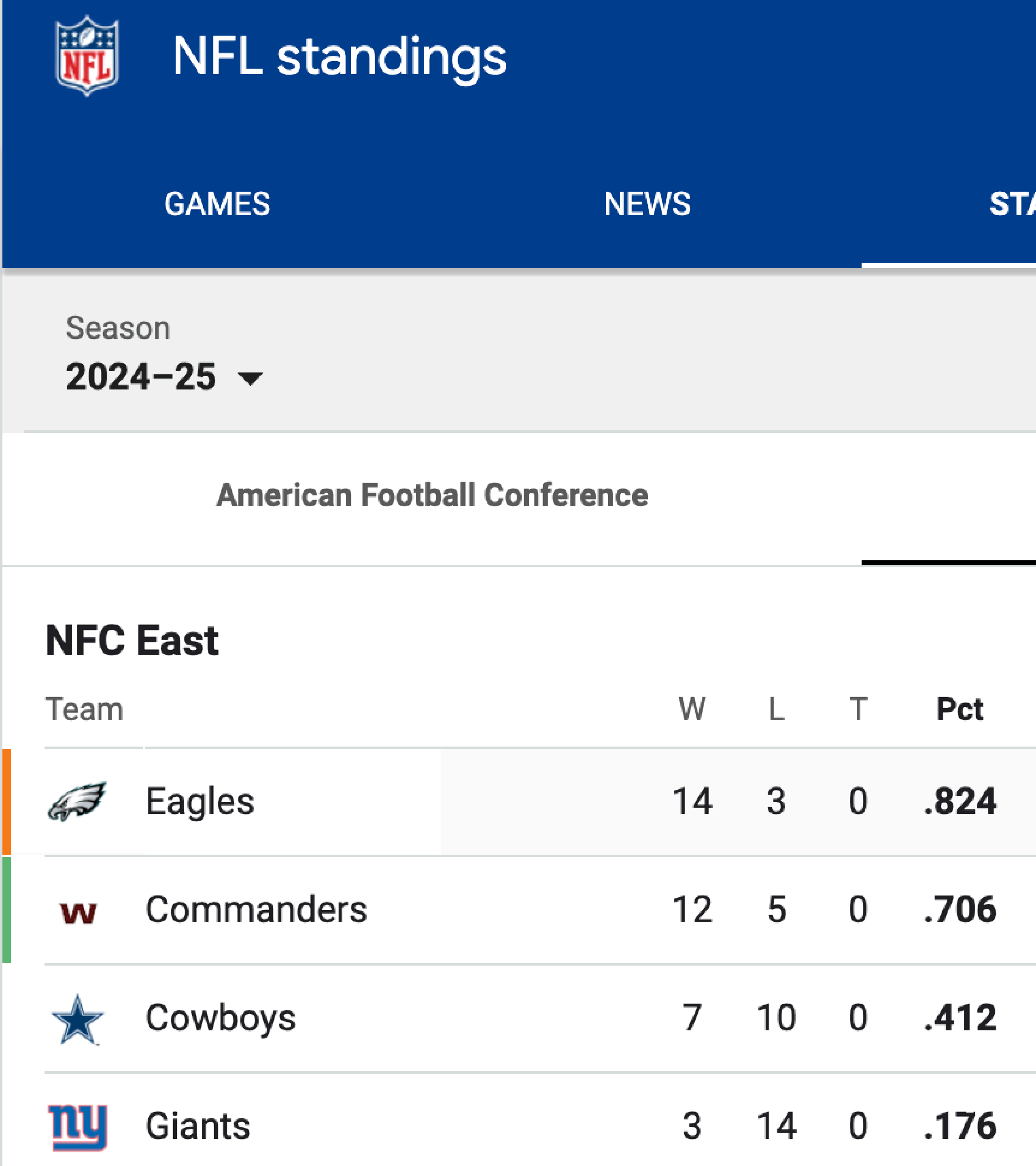
31 Dona Clara Rosa de Moura

INTRODUCTION

Another Common Data Structure

- What are the records?
- What information is encoded by the structural relationships?
- What affordances does this structure provide you?

🦅🦅🦅🦅🦅🦅🦅🦅🦅🦅🦅🦅🦅🦅



NFL standings

Season
2024–25

American Football Conference

NFC East

Team	W	L	T	Pct
Eagles	14	3	0	.824
Commanders	12	5	0	.706
Cowboys	7	10	0	.412
Giants	3	14	0	.176

What You Will Learn in CIT 5940

- Find the syllabus here.
- You will learn:
  - Commonly used data structures and algorithms and their guarantees and tradeoffs
  - How to measure the effectiveness of a data structure or algorithm

Case Study: Searching in an Array

How do you determine where a value is stored inside of an array?

Case Study: Searching in an Array

An array is a simple data structure that stores an ordered—but not necessarily sorted—sequence of values.

```
[54, 74, 31, 53, 38, 9, 34, 90, 60, 42, 24, 7, 3, 99, 7, 55]
```

A reasonable procedure to search over this array:

```java
public static boolean contains(int[] array, int target) {
  for (int i = 0; i < array.length; i++) {
    if (array[i] == target) {
      return true;
    }
  }
  return false;
}
```

Example: Searching in an Array

Our approach requires us to do:

- one iteration of the for loop to confirm that `54` is present,
- three iterations to confirm that `31` is present, and
- eighteen iterations to confirm that `55` is present or that `-15` is not present.

```
[54, 74, 31, 53, 38, 9, 34, 90, 60, 42, 24, 7, 3, 99, 7, 55]
```

Example: Searching in an Array

What if we knew that our array was **sorted?**

- Challenges: we have to sort the array, and then we have to be careful about how we add new elements to a sorted array
- Advantages: we can use a **binary search** to find elements in the array much more quickly!

Example: Searching in a Sorted Array

```java
public static boolean contains(int[] arr, int target) {
  int low = 0;
  int high = arr.length - 1;
  while (low <= high) {
    int mid = (low + high) / 2;
    if (arr[mid] == target) {
      return true;
    } else if (arr[mid] < target) {
      low = mid + 1;
    } else {
      high = mid - 1;
    }
  }
  return false;
}
```

Example: Searching in an Array

- The operations that an array supports are:
  - accessing an element at a position
  - changing an element at a position
  - querying its length
- Imposing additional **invariants** on an array allows us to make other assumptions about what information our operations can give us.
- Finding an element in a Sorted Array can be much faster than finding an element in an Array since we can use a binary search to rule out half of the positions in the Sorted Array at each step.

Goals

We want you to be able to:

- design algorithms that are easy to understand, code, & debug by using data structures
- design software that makes efficient use of the computer's resources

Administrative Stuff

Pre-requisites and Co-requisites

- Programming: CIT 5900/5910 or CIS 1200
  - Comfort with writing & testing medium size programs in an object-oriented language
  - Java experience is very helpful
- Math: CIT 5920 or CIS 1600
- Algorithms: CIT 5960 (co-requisite)

Homeworks

- HW1: Catch a Plagiarist
- HW2: Algorithm analysis (written)
- HW3: File compression
- HW4: Autocomplete
- HW5: Search Engine
- HW6: Graphs

Also: a group project!

Assignment Expectations

- Assignments are largely autograded
  - Instant feedback on submission! 🎉
  - Transparent grading criteria! 🎉
  - Defines a narrow specification that must be conformed to! 🙃
- You will likely need help during office hours
  - We have several OH per week, but there are more of you than there are of us.
  - You are expected to be **writing your own test cases!**
    - (it's part of your grade)
    - it helps streamline office hours questions & keep queues short.

Assignment Expectations

1. Understand the Problem
    i. What are the relevant concepts and how do they relate?
2. Formalize the Interface
    i. How should the program interact with its environment?
3. Write Test Cases
    i. How does the program behave on typical inputs?
    ii. How does the program behave on **unusual** inputs, or invalid ones?
4. Implement the Required Behavior
    i. Decompose the problem into simpler ones & apply this process to each.

CIT
5940

Questions & Website Tour