

# COT3100 Exam 2 review

Section 1ZED (Carson, Zac)

Feb 27, 2024

# Topics

# Exam details

- Time: 8:20 to 10:20 PM
- Topics:
  - 2.3 to 2.5 (no matrices!)
  - 3.1 to 3.3
  - Somewhat cumulative (proof techniques, etc.)
- Things to bring:
  - Writing utensils
  - Handwritten reference sheet (8.5x11)
  - 4 function calculator
  - ID (UF ID, state ID, or ID on phone)

# Common mistakes!

- 1 What is the parity of 0?
- 2 List all primes less than 10.
- 3 What is  $0!$  ?
- 4 What is  $\lfloor -3.2 \rfloor$ ?

# Topics

# Functions

## Definition

A *function*  $f$  is written as

$$f: A \rightarrow B,$$

where  $A$  is the *domain* of  $f$  and  $B$  is the *codomain* of  $f$ . Every element of the domain is mapped to exactly one element of the codomain.

## Definition

The *range* of a function  $f$  is the set of all values of the codomain that are actually mapped to by some value of the domain.

Be careful in using codomain and range! The codomain is the *possible* values that can be mapped to, but they aren't necessarily all mapped to.

# One-to-one and onto

## Definition

A function  $f$  is *one-to-one* or *injective* if every element of the codomain has *at most* one domain element mapped to it.

## Definition

A function  $f$  is *onto* or *surjective* if every element of the codomain has *at least* one domain element mapped to it.

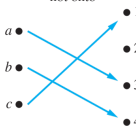
See the relationship?

# Bijections

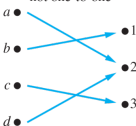
## Definition

A function  $f$  is a *one-to-one correspondence* or *bijection* if it is both one-to-one and onto. Equivalently, each codomain element has *exactly one* domain element mapped to it.

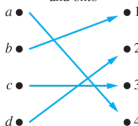
(a) One-to-one,  
not onto



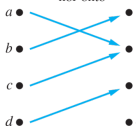
(b) Onto,  
not one-to-one



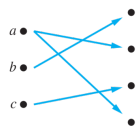
(c) One-to-one  
and onto



(d) Neither one-to-one  
nor onto



(e) Not a function





# Mathematical definitions

## Definition

A function  $f$  is *one-to-one* if for any two elements  $x_1$  and  $x_2$  in the domain of  $f$ ,

$$f(x_1) = f(x_2) \implies x_1 = x_2.$$

## Definition

A function  $f$  is *onto* if for any element  $y$  in the codomain of  $f$ , there exists an  $x$  in the domain where

$$f(x) = y.$$

## Note

An onto function has the same range and codomain (why?).

# Practice

## Problem

Given  $f: \mathbb{R} \rightarrow \mathbb{R}^+$ , where  $f(x) = x^4$ , prove or disprove if  $f$  is one-to-one. Prove or disprove if  $f$  is onto. What if the domain was  $\mathbb{R}^+$ ?

# Topics

# Important series

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}$$

$$\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}$$

$$\sum_{i=1}^n i^3 = \frac{n^2(n+1)^2}{4}$$

$$\sum_{i=1}^n 1 = ?$$

# Important series

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}$$

$$\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}$$

$$\sum_{i=1}^n i^3 = \frac{n^2(n+1)^2}{4}$$

$$\sum_{i=1}^n 1 = n$$

# Practice

## Problem

Evaluate

$$\sum_{j=0}^4 \sum_{k=98}^{100} \frac{1}{25} (k - 97)(j + 1)^3.$$

# Topics

# Countability

## Definition

A *countable* set  $S$  has a bijection

$$f: \mathbb{Z}^+ \rightarrow S.$$

A set where this bijection does not exist is *uncountable*.

## Note

The integers  $\mathbb{Z}$ , the set of even integers, and (surprisingly) the set of rationals  $\mathbb{Q}$  are all examples of countable sets.

## Note

The set of reals,  $\mathbb{R}$  is uncountable. Any interval of  $\mathbb{R}$  is also uncountable. The set of irrationals is also uncountable (why?).



# Practice

## Problem

Prove that the subset of a countable set is countable.

# Topics

# Searching and sorting complexities

Search algorithms:

- Linear search -  $O(n)$
- Binary search -  $O(\log_2 n)$

Sorting algorithms:

- Bubble sort -  $O(n^2)$
- Selection sort -  $O(n^2)$
- Insertion sort -  $O(n^2)$

Refer to the textbook and discussion slides for details of each algorithm.

# Pseudocode

When writing pseudocode, getting the message across is most important

- Will likely look similar to Python
- Can replace technical syntax (`array[i][j]=n`) with description (“set array element at  $(i,j)$  to  $n$ ”)

# Pseudocode

When writing pseudocode, getting the message across is most important

- Will likely look similar to Python
- Can replace technical syntax (`array[i][j]=n`) with description (“set array element at  $(i,j)$  to  $n$ ”)

## Warning!

You will likely need to write algorithms beyond searching/sorting!  
Be ready to work with arrays (lists), strings, and numbers.

## Problem

For an integer array `nums` of size at least 2, let a *peak* be an index  $i$  of `nums` where the element at  $i$  is strictly greater than the elements at indices  $i - 1$  (if it exists) and  $i + 1$  (if it exists). Describe an algorithm `peak_count` that finds the number of peaks of a given `nums`.

# Practice

## Problem

For an integer array `nums` of size at least 2, let a *peak* be an index  $i$  of `nums` where the element at  $i$  is strictly greater than the elements at indices  $i - 1$  (if it exists) and  $i + 1$  (if it exists). Describe an algorithm `peak_count` that finds the number of peaks of a given `nums`.

## Example

If `nums=[2,1,5,3,1]`, then `nums` has a peak at  $i = 0$  (2) since  $2 > 1$ , and a peak at  $i = 2$  (5) since  $5 > 1$  and  $5 > 3$ .

# Topics



# Growth rates

## Note

The following order represents the growth rates of functions from slowest to fastest:

$$1 \ll \log n \ll n \ll n \log n \ll n^2 \ll (\text{polynomials}) \ll 2^n \ll n! \ll n^n$$

# Big $O$

## Definition

A function  $f(x)$  is  $O(g(x))$  if there are  $C$  and  $k$  such that

$$|f(x)| \leq C|g(x)|$$

for all  $x > k$ .

We think of big  $O$  as an *upper bound* for the growth of  $f$ . In other words,  $g$  either grows faster or at the same rate as  $f$ .

# Big $O$

## Definition

A function  $f(x)$  is  $O(g(x))$  if there are  $C$  and  $k$  such that

$$|f(x)| \leq C|g(x)|$$

for all  $x > k$ .

We think of big  $O$  as an *upper bound* for the growth of  $f$ . In other words,  $g$  either grows faster or at the same rate as  $f$ .

## Example

If  $f(x) = x^2 + 3$ , then  $f(x)$  is  $O(x^2)$ . However,  $f(x)$  is also  $O(x^3)$ ,  $O(2^x)$ ,  $O(x!)$ , and many more.

# Big $\Omega$

## Definition

A function  $f(x)$  is  $\Omega(g(x))$  if there are  $C$  and  $k$  such that

$$|f(x)| \geq C|g(x)|$$

for all  $x > k$ .

We think of big  $\Omega$  as a *lower bound* for the growth of  $f$ . In other words,  $g$  either grows slower or at the same rate as  $f$ .

# Big $\Omega$

## Definition

A function  $f(x)$  is  $\Omega(g(x))$  if there are  $C$  and  $k$  such that

$$|f(x)| \geq C|g(x)|$$

for all  $x > k$ .

We think of big  $\Omega$  as a *lower bound* for the growth of  $f$ . In other words,  $g$  either grows slower or at the same rate as  $f$ .

## Example

If  $f(x) = x^2 + 3$ , then  $f(x)$  is  $\Omega(x^2)$ . However,  $f(x)$  is also  $\Omega(x)$ ,  $\Omega(1)$ , etc.

# Big $\Theta$

## Definition

A function  $f(x)$  is  $\Theta(g(x))$  if it is both  $O(g(x))$  and  $\Omega(g(x))$

We think of big  $\Theta$  as giving a “class” of functions growing at a similar rate.

## Note

The “optimal” big  $O$  refers to the *slowest* growing big  $O$  possible. Similarly, the “optimal” big  $\Omega$  refers to the *fastest* growing big  $\Omega$  possible. These both end up the same as big  $\Theta$ .

# Big $\Theta$

## Definition

A function  $f(x)$  is  $\Theta(g(x))$  if it is both  $O(g(x))$  and  $\Omega(g(x))$

We think of big  $\Theta$  as giving a “class” of functions growing at a similar rate.

## Note

The “optimal” big  $O$  refers to the *slowest* growing big  $O$  possible. Similarly, the “optimal” big  $\Omega$  refers to the *fastest* growing big  $\Omega$  possible. These both end up the same as big  $\Theta$ .

## Example

If  $f(x) = x^2 + 3$ , then  $f(x)$  is  $\Theta(x^2)$ . Its “optimal” big  $O$  and big  $\Omega$  are  $O(x^2)$  and  $\Omega(x^2)$ , respectively.

# Practice

## Problem

Order the following functions of  $n$  by their growth rates from *fastest* to *slowest*.

- |     |      |              |             |            |            |
|-----|------|--------------|-------------|------------|------------|
| (1) | 2024 | $(1.0001)^n$ | $\log(n^n)$ | $4n^2 - 1$ | $n^{2024}$ |
| (2) | 2024 | $(1.0001)^n$ | $\log(n^n)$ | $4n^2 - 1$ | $n^{2024}$ |
| (3) | 2024 | $(1.0001)^n$ | $\log(n^n)$ | $4n^2 - 1$ | $n^{2024}$ |
| (4) | 2024 | $(1.0001)^n$ | $\log(n^n)$ | $4n^2 - 1$ | $n^{2024}$ |
| (5) | 2024 | $(1.0001)^n$ | $\log(n^n)$ | $4n^2 - 1$ | $n^{2024}$ |

## Problem

For  $f(n) = 4n^3 + n - 7$ , prove that  $f(n)$  is  $\Theta(n^3)$ .



# Practice

## Problem

Order the following functions of  $n$  by their growth rates from *fastest* to *slowest*.

- |     |      |              |             |            |            |
|-----|------|--------------|-------------|------------|------------|
| (1) | 2024 | $(1.0001)^n$ | $\log(n^n)$ | $4n^2 - 1$ | $n^{2024}$ |
| (2) | 2024 | $(1.0001)^n$ | $\log(n^n)$ | $4n^2 - 1$ | $n^{2024}$ |
| (3) | 2024 | $(1.0001)^n$ | $\log(n^n)$ | $4n^2 - 1$ | $n^{2024}$ |
| (4) | 2024 | $(1.0001)^n$ | $\log(n^n)$ | $4n^2 - 1$ | $n^{2024}$ |
| (5) | 2024 | $(1.0001)^n$ | $\log(n^n)$ | $4n^2 - 1$ | $n^{2024}$ |

## Problem

For  $f(n) = 4n^3 + n - 7$ , prove that  $f(n)$  is  $\Theta(n^3)$ .

# Topics

# Practice

## Problem

Find the time complexity of the `peak_count` algorithm.