CS 577 # Assignment 1 – Discrete Review Fall 2024

Answer the questions in the boxes provided on the question sheets. If you run out of room for an answer, add a page to the end of the document.

Related Readings:

- Proof Primer: `https://canvas.wisc.edu/courses/412522/files/40339266`
- CS240 Readings: `http://pages.cs.wisc.edu/~hasti/cs240/readings/`

Name: _____ Wisc id: _____

Induction

1. Prove the following by induction.
   (a) $\sum_{i=1}^{n} i = n(n+1)/2$

Solution:
Base case: $n = 1$; $n(n+1)/2 = 1$
Induction step for $k+1$:

$$\begin{aligned}\sum_{i=1}^{k+1} i &= \left(\sum_{i=1}^{k} i\right) + k + 1 \\ &= k(k+1)/2 + k + 1, \text{by ind hyp,} \\ &= (k(k+1) + 2k + 2)/2 = (k+1)(k+2)/2\end{aligned}$$

□

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

Solution:
Base case: $n = 1$; $n(n+1)(2n+1)/6 = 1$
Induction step for $k+1$:

$$\begin{aligned}\sum_{i=1}^{k+1} i^2 &= \left(\sum_{i=1}^{k} i^2\right) + (k+1)^2 \\ &= k(k+1)(2k+1)/6 + (k+1)^2, \text{by ind hyp,} \\ &= (k(k+1)(2k+1) + 6(k+1)^2)/6 \\ &= (k+1)(2k^2 + 7k + 6)/6 \\ &= (k+1)(k+2)(2k+3)/6\ .\end{aligned}$$

□

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

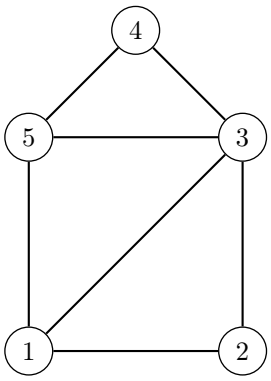
Solution:
Base case: $n = 1$; $n^2(n+1)^2/4 = 1$
Induction step for $k+1$

$$\begin{aligned}\sum_{i=1}^{k+1} i^3 &= \left(\sum_{i=1}^{k} i^3\right) + (k+1)^3 \\ &= k^2(k+1)^2/4 + (k+1)^3, \text{by ind hyp,} \\ &= (k^2(k+1)^2 + 4(k+1)^3)/4 \\ &= (k+1)^2(k^2+4k+4)/4 \\ &= (k+1)^2(k+2)^2/4\ .\end{aligned}$$

□

Graphs and Trees

2. Give the adjacency matrix, adjacency list, edge list, and incidence matrix for the following graph.



Solution:

Adjacency Matrix:

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

Adjacency List:

$$\begin{pmatrix} (2,3,5) \\ (1,3) \\ (1,2,4,5) \\ (3,5) \\ (1,3,4) \end{pmatrix}$$

Edge List:

$\big((1,2),(1,3),(1,5),(2,3),(3,4),(3,5),(4,5)\big)$

Incidence Matrix:

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$

3. How many edges are there is a complete graph of size n? Prove by induction.

Solution:

There are $h(n) = n(n-1)/2$ edges.

By induction on the number of nodes. Base case: $n = 1$. $h(1) = 0$.

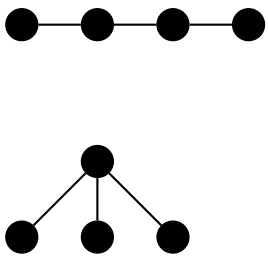
Induction step show for k_{i+1}: Consider a complete graphs k_{i+1} and an arbitrary node u. By definition, u has an edge to each of the i other nodes, and $k_{i+1} \setminus u$ gives k_i. By ind hyp, k_i has $i(i-1)/2$ edges. Hence,

$$h(i+1) = h(i) + i = i(i-1)/2 + i = \frac{i(i-1)+2i}{2} = \frac{i((i-1)+2)}{2} = (i+1)i/2 \ .$$

□

4. Draw all possible (unlabelled) trees with 4 nodes.

Solution:



5. Show by induction that, for all trees, $|E| = |V| - 1$.

Solution:

By (structural) induction on the number of nodes. Base Case: Single node; no edges

Induction step: Consider an arbitrary tree T with k nodes. Removing any leaf node will also remove 1 edge from T and produce a tree T' with $k-1$ nodes. By ind hyp, T', a tree of k nodes has $k-1$ edges. Hence,

$$|E| = k - 1 + 1 = k = |V| - 1 \ .$$

□

Proofs

6. Show that $2x$ is even for all $x \in \mathbb{N}$.

   (a) By direct proof.

   Solution: By definition, an even number is divisible by 2, and $2x$ is divisible by 2. □

   (b) By contradiction.

   Solution: Assume to the contrary that $2x$ is odd. This implies that $2x$ is not perfectly divisible by 2. However, $2x/2 = x$, a contradiction.

7. For all $x, y \in \mathbb{R}$, show that $|x + y| \leq |x| + |y|$. (Hint: use proof by cases.)

Solution:

Case 1: $x \geq 0, y \geq 0$. $|x + y| = x + y = |x| + |y|$.

Case 2: $x < 0, y < 0$. $|x + y| = -(x + y) = |x| + |y|$.

Case 3: $x = 0, y < 0$. $|x + y| = |y| < |x| + |y|$.

Case 4: $x > 0, y < 0$ **and** $|x| \geq |y|$. $|x + y| < |x| \leq |x| + |y|$.

Case 5: $x > 0, y < 0$ **and** $|x| < |y|$. $|x + y| < |y| \leq |x| + |y|$.

WLOG: Case 3 covers $x < 0, y = 0$; Case 4 covers $y > 0, x < 0$ and $|y| \geq |x|$; and Case 5 covers $y > 0, x < 0$ and $|y| < |x|$

Program Correctness (and Invariants)

8. For the following algorithms, describe the loop invariant(s) and prove that they are sound and complete.

(a)

Algorithm 1: findMin

```
Input: a: A non-empty array of integers (indexed starting at 1)
Output: The smallest element in the array
begin
    min ← ∞
    for i ← 1 to len(a) do
        if a[i] < min then
            min ← a[i]
        end
    end
    return min
end
```

Solution:

Invariant: At the end of step i of the loop, min contains the minimum value from index 1 to i of a.

[Proof: By induction on the iterations. After the first iteration, min is set to $a[1]$. Assume that min is minimum value of $a[1, \ldots, k]$ after the kth iteration. After the $(k+1)$-st iteration, min will be the minimum of min and $a[k+1]$. Hence, min is the minimum of $a[1, \ldots, k+1]$.]

Soundness: From the loop invariant, after completing the step equal to the length of the input array, min will contain the minimum element.

Completeness: With each iteration of the loop, the counter i moves one step closer to the end of the input array, and the loop (and the algorithm) terminates after reaching the end of the array.

(b)

Algorithm 2: InsertionSort

```
Input: a: A non-empty array of integers (indexed starting at 1)
Output: a sorted from largest to smallest
begin
    for i ← 2 to len(a) do
        val ← a[i]
        for j ← 1 to i − 1 do
            if val > a[j] then
                shift a[j..i − 1] to a[j + 1..i]
                a[j] ← val
                break
            end
        end
    end
    return a
end
```

Solution:

Invariant: At the end of step i of the outer loop, the items $a[1..i]$ are sorted from largest to smallest.

[Proof: By induction on the iterations. After the first iteration, the item $a[1]$ is sorted. Assume that $a[1, \ldots, k]$ is sorted after the kth iteration. After the $(k+1)$-st iteration, $a[k+1]$ will be placed at $a[j]$ after the sorted $a[1, \ldots, j-1]$ items larger than $a[k+1]$. The sorted $a[j, k]$ items less than $a[k+1]$ are shifted to $a[j+1, k+1]$. Hence, $a[1, \ldots, k+1]$ is sorted.]

Soundness: From the loop invariant, after completing the step (of the outer loop) equal to the length of the input array, a will be sorted from largest to smallest.

Completeness: With each iteration of the loop, the outer loop counter i moves one step closer to the end of the input array, and the outer loop (and the algorithm) terminates after reaching the end of the array. For each iteration, the inner loop increments by 1 from 1 to i (the counter of the outer loop), and will always terminate.

Coding Question: Hello World

Most assignments will have a coding question. You can code in C, C++, C#, Java, Python, or Rust. You will submit a Makefile and a source code file.

Makefile: In the Makefile, there needs to be a build command and a run command. Below is a sample Makefile for a C++ program. You will find this Makefile in assignment details. Download the sample Makefile and edit it for your chosen programming language and code.

```
#Build commands to copy:
#Replace g++ -o HelloWorld HelloWord.cpp below with the appropriate command.
#Java:
#       javac source_file.java
#Python:
#       echo "Nothing to compile."
#C#:
#       mcs -out:exec_name source_file.cs
#C:
#       gcc -o exec_name source_file.c
#C++:
#       g++ -o exec_name source_file.cpp
#Rust:
#       rustc source_file.rs

build:
        g++ -o HelloWorld HelloWord.cpp

#Run commands to copy:
#Replace ./HelloWorld below with the appropriate command.
#Java:
#       java source_file
#Python 3:
#       python3 source_file.py
#C#:
#       mono exec_name
#C/C++:
#       ./exec_name
#Rust:
#       ./source_file

run:
        ./HelloWorld
```

9. **HelloWorld Program Details**

   The input will start with a positive integer, giving the number of instances that follow. For each instance, there will be a string. For each string s, the program should output Hello, s! on its own line.

   A sample input is the following:

```
3
World
Marc
Owen
```

   The output for the sample input should be the following:

```
Hello, World!
Hello, Marc!
Hello, Owen!
```