Assignment 1 – Discrete Review

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: `http://pages.cs.wisc.edu/~hasti/cs240/readings/`

Name: Kayley Seow Wisc id: kseow

Logic

1. Using a truth table, show the equivalence of the following statements.

(a) $P \overset{\text{or}}{\vee} (\neg P \overset{\text{and}}{\wedge} Q) \equiv P \vee Q$

Solution:

P	$\neg P$	Q	$\neg P \wedge Q$	$P \vee (\neg P \wedge Q)$
T	F	T	F	T
T	F	F	F	T
F	T	T	T	T
F	T	F	F	F

P	Q	$P \vee Q$
T	T	T
T	F	T
F	T	T
F	F	F

(b) $\neg P \vee \neg Q \equiv \neg(P \wedge Q)$

Solution:

P	Q	$\neg P$	$\neg Q$	$\neg P \vee \neg Q$
T	T	F	F	F
T	F	F	T	T
F	T	T	F	T
F	F	T	T	T

P	Q	$P \wedge Q$	$\neg(P \wedge Q)$
T	T	T	F
T	F	F	T
F	T	F	T
F	F	F	T

(c) $\neg P \overset{\text{or}}{\vee} P \equiv \text{true}$

Solution:

P	$\neg P$	$\neg P \vee P$
T	F	T
F	T	T

$\Rightarrow$ true

(d) $P \overset{\text{or}}{\vee} (Q \overset{\text{and}}{\wedge} R) \equiv (P \overset{\text{or}}{\vee} Q) \overset{\text{and}}{\wedge} (P \overset{\text{or}}{\vee} R)$

Solution:

P	Q	R	$Q \wedge R$	$P \vee (Q \wedge R)$
T	T	T	T	T
T	T	F	F	T
T	F	T	F	T
T	F	F	F	T
F	T	T	T	T
F	T	F	F	F
F	F	T	F	F
F	F	F	F	F

P	Q	R	$P \vee Q$	$P \vee R$	$(P \vee Q) \wedge (P \vee R)$
T	T	T	T	T	T
T	T	F	T	T	T
T	F	T	T	T	T
T	F	F	T	T	T
F	T	T	T	T	T
F	T	F	T	F	F
F	F	T	F	T	F
F	F	F	F	F	F

Sets

cardinality union intersection

2. Based on the definitions of the sets A and B, calculate the following: $|A|$, $|B|$, $A \cup B$, $A \cap B$, $A \setminus B$, $B \setminus A$.

elements in A not in B

   (a) $A = \{1, 2, 6, 10\}$ and $B = \{2, 4, 9, 10\}$

   Solution:
   $|A| = 4$
   $|B| = 4$
   $A \cup B = \{1, 2, 4, 6, 9, 10\}$
   $A \cap B = \{2, 10\}$
   $A \setminus B = \{1, 6\}$
   $B \setminus A = \{4, 9\}$

   x, x is a number a number, x is even

   (b) $A = \{x \mid x \in \mathbb{N}\}$ and $B = \{x \in \mathbb{N} \mid x \text{ is even}\}$

   Solution:
   $|A| = \infty$ $|B| = \infty$
   $A \cup B = \{1, 2, 3, 4 \ldots\}$
   $A \cap B = \{\ \}$
   $A \setminus B = \{x \in N \mid x \text{ is odd}\}$
   $B \setminus A = \{\ \}$

Relations and Functions

3. For each of the following relations, indicate if it is reflexive, antireflexive, symmetric, antisymmetric, or transitive.

   (a) $\{(x, y) : x \leq y\}$

   Solution: reflexive, symmetric, transitive

   (b) $\{(x, y) : x > y\}$

   Solution: antireflexive, antisymmetric, transitive

(c) $\{(x, y) : x < y\}$

Solution: antireflexive, antisymmetric, transitive

(d) $\{(x, y) : x = y\}$

Solution: reflexic, symmetric, transitive

4. For each of the following functions (assume that they are all $f : \mathbb{Z} \to \mathbb{Z}$), indicate if it is surjective (onto), injective (one-to-one), or bijective.

(a) $f(x) = x$

Solution: bijective

(b) $f(x) = 2x - 3$

Solution: injective

(c) $f(x) = x^2$ negative numbers -not inj b/c can have multiple -not sur b/c can never map to negative

Solution: none

5. Show that $h(x) = g(f(x))$ is a bijection if $g(x)$ and $f(x)$ are bijections.

Solution:

h(x) is one to one, injective

because g(x) is a bijection, no matter the output for f(x), the output of g(x) will be a one to one function.

$h(x_1) = h(x_2)$, $g(f(x_1)) = g(f(x_2))$, $f(x_1) = f(x_2)$, one to one $x_1 = x_2$

h(x) is onto, surjective

Because g(x) is a bijection, there will be an inverse

$h^{-1}(x) = f^{-1}(g^{-1}(x))$, $h^{-1}(h(x)) = f^{-1}(g^{-1}(g(f(x)))) = f^{-1}(f(x)) = x$

$h^{-1}(h(x)) = g(f(f^{-1}(g^{-1}(x)))) = g(x)$ so $h^{-1}(x)$ is inverse of $h(x)$

Induction

6. Prove the following by induction. $n=k \quad = \frac{k(k+1)}{2} = \frac{k^2+k}{2}$

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

Solution: Predicate: $P(n)$ be that $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ holds

Base case: plug in $i=1$, see if it holds (first step in the sum)

$\sum_{i=1}^{1} 1 = \frac{1(1+1)}{2} \Rightarrow 1 = \frac{2}{2}$, base case holds

Inductive: prove that $P(k+1)$ holds

$\sum_{i=1}^{k+1} i = \frac{k+1\,(k+1+1)}{2} = \frac{(k+1)(k+2)}{2} \qquad k+1 \quad \sum_{i=1}^{n} \frac{k(k+1)}{2} = \frac{k^2+3k+2}{2}$

inductive hypothesis:

$k+1 + \frac{k(k+1)}{2} = \frac{k^2+3k+2}{2} \rightarrow k+1+\frac{k^2+k}{2} = \frac{k^2+3k+2}{2} \qquad k+1 = \frac{2k+2}{2}$

(b) $\sum_{i=1}^{n} i^2 = n(n+1)(2n+1)/6$ $\quad 2(k+1)+1 = 2k+3$

Solution: predicate: let $p(n)$ be that $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$ holds

Base Case: plug in $i=1$

$1(2)(2+1)/6 = 6/6 = 1$

Inductive: prove $p(k+1)$ holds

$\sum_{i=1}^{k+1} = \frac{(k+1)(k+2)(2k+3)}{6} \Rightarrow (k+1)^2 + \sum_{i=1}^{k} i^2 = \frac{(k+1)(k+2)(2k+3)}{6}$

$(k^2+k)(2k+1)$

$2k^3+k^2+2k^2+k$

inductive hypothesis:

$(k+1)^2 + \frac{k(k+1)(2k+1)}{6} = \frac{(k+1)(k+2)(2k+3)}{6}, \quad \frac{(k+1)(2k+3)(k+2)}{6} = \frac{(k^2+3k+2)(2k+3)}{6}$

$k^2+2k+1 + \frac{2k^3+k^2+2k^2+k}{6} = \frac{(k^2+3k+2)(2k+3)}{6},$

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

Solution: predicate: let $p(n)$ be that $\sum_{i=1}^{n} i^3 = \frac{n^2(n+1)^2}{4}$ holds

Base case: plug $i=1$ $\quad 1^3 = \frac{1^2(2)^2}{4} \Rightarrow 1 = \frac{4}{4} = 1$

Inductive: prove $p(k+1)$ holds

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

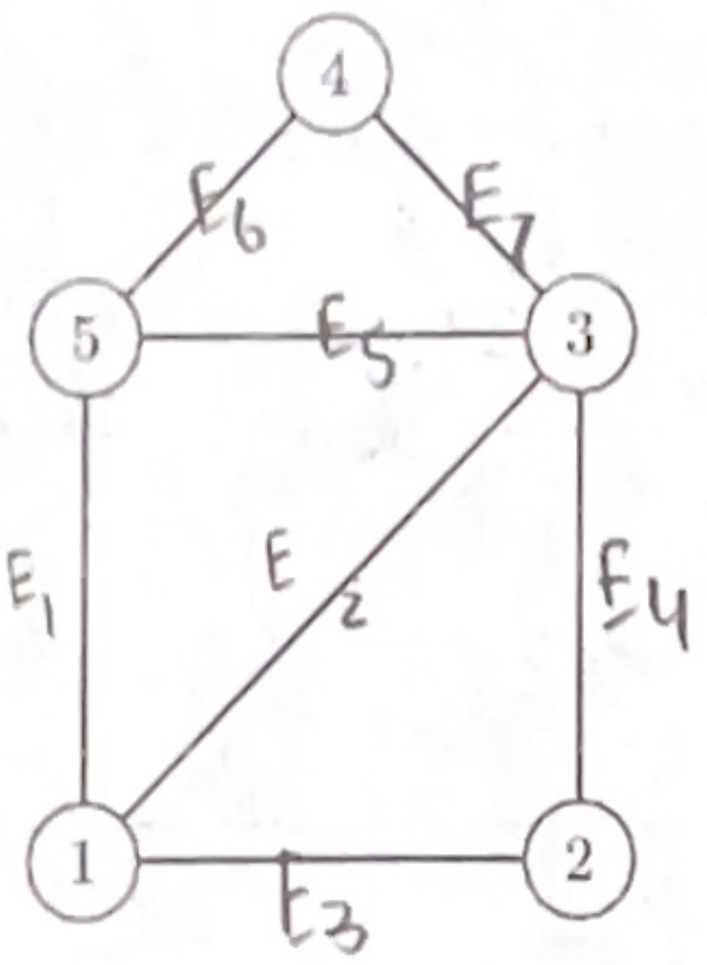
inductive hypothesis

$\frac{(k+1)^2(k+2)^2}{4} = (k+1)^3 + \frac{(k)^2(k+1)^2}{4}$

$\frac{(k+1)^3 + k^2(k+1)^2}{4} = \frac{(k+1)^2(k+2)^2}{4} \Rightarrow \frac{(k+1)^2(k+2)^2}{4}$

Graphs and Trees

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



Solution:

adjacency matrix:

$$\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 1 & 1 & 0 & 1 \\ 2 & 0 & 0 & 1 & 0 & 0 \\ 3 & 1 & 1 & 0 & 1 & 1 \\ 4 & 0 & 0 & 1 & 0 & 1 \\ 5 & 1 & 0 & 1 & 1 & 0 \end{array}$$

adjacency list

$1 \to 2 \to 3 \to 5$

$2 \to 1 \to 3$

$3 \to 1 \to 2 \to 4 \to 5$

$4 \to 3 \to 5$

$5 \to 1 \to 3 \to 4$

Edge List:

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

incidence matrix:

$$\begin{array}{c|ccccccc} & E_1 & E_2 & E_3 & E_4 & E_5 & E_6 & E_7 \\ \hline 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 4 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 5 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \end{array}$$

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

Solution: prove K_n has $\frac{n(n-1)}{2}$ edges

Base cases: $n=1$

$\frac{1(1-1)}{2} = 0$, proves equation works

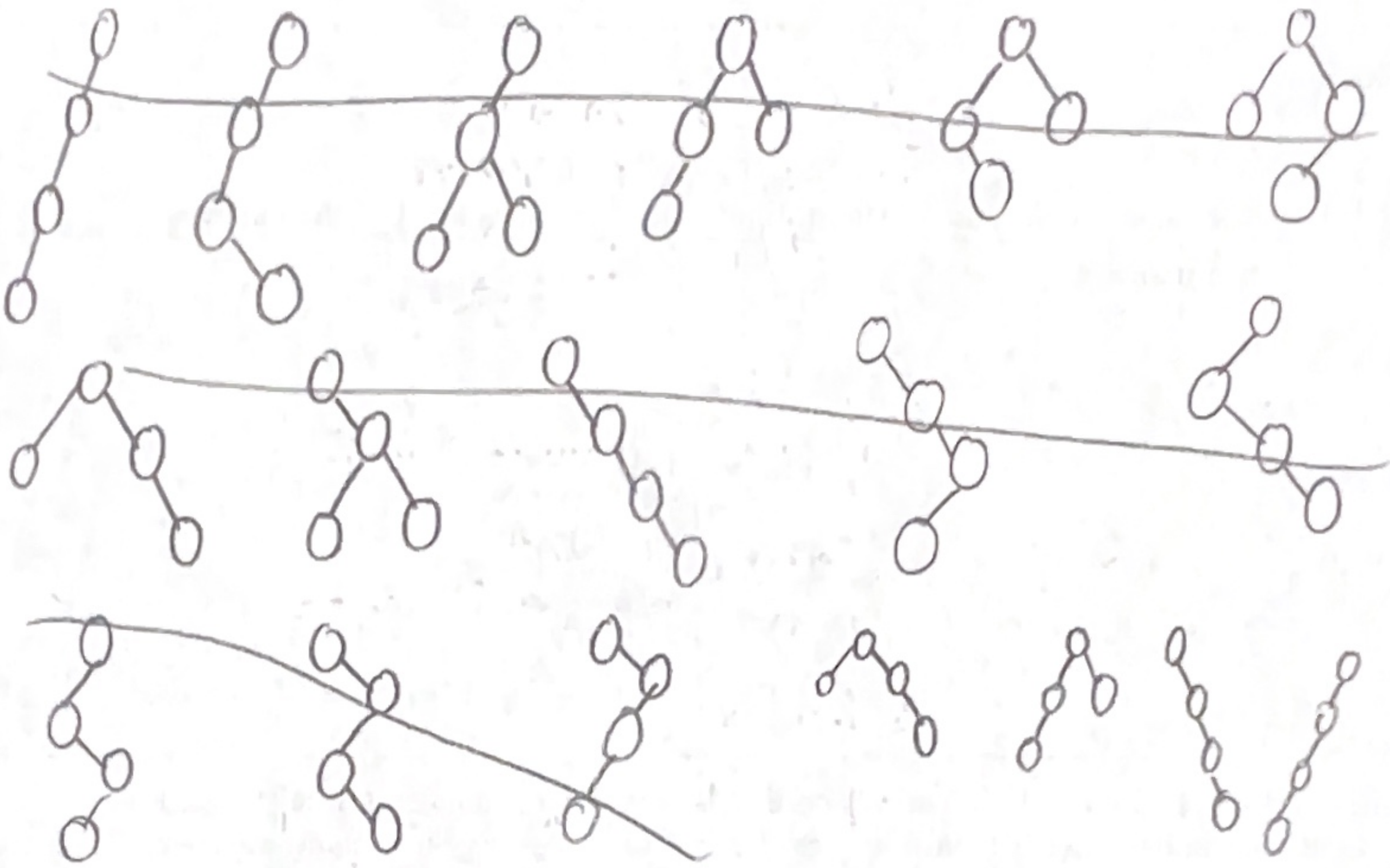
Induction hypothesis: K_n has $\frac{k(k-1)}{2}$ edges, complete graph, has K_{k+1} has $\frac{k(k+1)}{2}$ edges

$$\frac{k(k-1)}{2} + \overset{k+1}{k} = \frac{k^2-k}{2} + k = \frac{k^2-k}{2} + \frac{2k}{2} = \frac{k^2-k+2k}{2} = \frac{k^2+k}{2} = \frac{k(k+1)}{2}$$

every node in a completegraph of k+1 has $k(k+1)/2$ edges, every node has a degree of k

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

Solution:



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

Solution:

Base: $|V| = 1$ A one vertex graph has no edges $|E| = 0$ a one edge graph has two vertices $|V| = 2$

$|E| = 1 - 1 = 0$

Induction Hypothesis: For $|V| = k$, $|E| = k - 1$

Prove when $|V| = k+1$, $|E| = k$

in a tree, add a node/vertex means adding an edge

from induction, $|E| = k-1$ when $|V| = k$, adding vertex to make the tree $k+1$ would add an edge, making $|E| = k-1+1, = k$, same result as from original equation.

Counting

11. How many 3 digit pin codes are there? 10^3

Solution: 1000

12. What is the expression for the sum of the ith line (indexing starts at 1) of the following:

1 — 1
4 — 5 — 2 3
6, 10 — 15 — 4 5 6
9, 19 — 34 — 7 8 9 10
12, 3, 31 — 65 — 11 12 13 14 15
⋮

Solution:

$2a=3, a=1.5$

$3a+b=4, 4.5+b=4, b=-0.5$

$a+b+c=1, 1.5-0.5+c=1, 1+c=1, c=0$

$u_n = 1.5n^2 + -0.5n$

$= \frac{3n^2-n}{2} = \frac{n(3n-1)}{2}$

$2a=3, a=\frac{3}{2}$

$3a+b=4, \frac{9}{2}+b=4, b=\frac{8}{2}-\frac{9}{2}=\frac{-1}{2}$

$a+b+c=1, \frac{3}{2}-\frac{1}{2}+c=1, 1+c=1, c=0$

Σ in: elements in nth row

starting value: $\frac{n}{2}(n-1)+1$

last value: $\frac{n}{2}(n-1)+n$

gauss trick: pair values $n(n-1)+1+n = n^2+1$

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

$3a =$

13. A standard deck of 52 cards has 4 suits, and each suit has card number 1 (ace) to 10, a jack, a queen, and a king. A standard poker hand has 5 cards. For the following, how many ways can the described hand be drawn from a standard deck.

(a) A royal flush: all 5 cards have the same suit and are 10, jack, queen, king, ace.

Solution: 4

(b) A straight flush: all 5 cards have the same suit and are in sequence, but not a royal flush.

Solution: $9 \times 4 = 36$

(c) A flush: all 5 cards have the same suit, but not a royal or straight flush.

Solution: -36 $5148 - 40$

$4 \; {}_{13\,\text{cards}}C_5 - 36 = 1287 \times 4 - 36 = 5108$

(d) Only one pair (2 of the 5 cards have the same number/rank, while the remaining 3 cards all have different numbers/ranks):

Solution:

$4(1287) - (48 \cdot 44 \cdot 40) = 5148 -$

$4 \; {}_{13}C_2 - 48 \cdot 44 \cdot 40$

Proofs

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

(a) By direct proof.

Solution:

$2 | 2x = x \in N$

Since $2x$ is divisible by 2, it is a multiple of 2.

$2x$ is even

(b) By contradiction.

Solution:

$2x = 2 \cdot k + 1$, $k \in N$ and $x \in N$

$2x - 2k = 1$ closure $x - k \in N$

$2(x-k) = 1$

not possible, $2x$ cannot be odd

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

Solution:

case 2: x and y are both positive means absolute value leaves us with the same numbers $x+y \leq x+y \Rightarrow x+y = x+y$ ✓ true

Case 2: x is negative, y is positive. $-x+y$ is the same as $y-x$
$y - x \leq x + y$ subtract from both
$|-x| \leq |x|$ ✓ true

case 3: y is neg. x is pos. By same logic as case 2, holds true.

case 4: x and y are both negative $-x + y = -(x+y)$ the absolute of $-(x+y)$ is $x+y$. $|-x|$ & $|-y|$ is x and y. $|-(x+y)| \leq |-x| + |-y|$ $= x+y \leq x+y \Rightarrow x=y$ ✓ this holds true.

Program Correctness (and Invariants)

16. 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 ← ∞ - initializing ∞
    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 loop i, the variable min contains the smallest element from array [2] to [i].

Base case:
The smallest iteration is $i=1$, and because minimum contains the largest possible value it can store, a[i] is bound to be less and is set to be the new value at min.

Induction Hypothesis: $p(n)$ holds where $p(n)$=min= min of subarray from first index (0) to last index (n), and we want to prove $p(n+1)$

Case 1: min > a[n+1]
min = a[n+1] This correct, assuming smallest value stored in min so if a[n+1] < min is the smallest value in the previous subarray, becomes the new correct value.

Case 2: min ≤ a[n+1]
This is correct because nothing happens and smallest element remains the same, increment it 1+1.

Terminate: length is a finite number, and loop starts at 1 until length-1, the program terminates.

(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 $i \leftarrow 2$ **to** $len(a)$ **do** i at 2 to length of a
    $val \leftarrow a[i]$ insert value at i into val
    for $j \leftarrow 1$ **to** $i - 1$ **do** from j at 1 to i=1
      if $val > a[j]$ **then** value at array greater than next
        shift $a[j..i-1]$ to $a[j+1..i]$
        $a[j] \leftarrow val$
        break
      end
    end
  end
  return a
end

Solution: loop invariant: At the end of loop i, the variable array a contains sorted numbers from largest to smallest at the end of the iteration.

Base case: p(1) when 1, the minimum element is stored at a[0] because it is the minimum element, so holds the base case

Iterative hypothesis - 1 to j subarray is sorted from largest to smallest. we want to prove that this is true.

Case 1: val > a[j]
In this case, the code puts a[j] in its place in the subarray.

Case 2: Val ≤ a[j]
Because of the inductive hypothesis, nothing happens and array remains sorted.

Termination: i starts at 2, increments until length -1, length is a finite number. so by this conclusion, loop terminates. j starts at 1, goes to (i-1), the program terminates.

Recurrences

17. Solve the following recurrences.

(a) $c_0 = 1; c_n = c_{n-1} + 4$

Solution:

$c_0 =$

$C_1 = 1+4 = 5$

$C_2 = 5+4 = 9$

$C_3 = 9+4 = 13$

$\boxed{C_n = 4n+1}$

$O(n)$

(b) $d_0 = 4; d_n = 3 \cdot d_{n-1}$

Solution:

$d_0 = 4$

$d_1 = 12$

$d_2 = 36$

$d_3 = 108$

$\boxed{C_n = 3^n \cdot 4}$

$O(3^n)$

(c) $T(1) = 1; T(n) = 2T(n/2) + n$ (An upper bound is sufficient.)

Solution:

$T(1) = 1$

$T(2) = 2(1) + 2 = 4$

$T(3) =$

$T(n/2) = 2T(\frac{n}{4}) + \frac{n}{2}$

$T(\frac{n}{4}) = 2T(\frac{n}{2^3}) + \frac{n}{4}$

$T(\frac{n}{2^k}) = T(1)$

$\frac{n}{2^k} = 1$

$n = 2^k$

$\log_2 n = k$

$\rightarrow 2(2T(\frac{n}{4}) + (\frac{n}{2}) + n$

$= 4 + (\frac{n}{4}) + 2n$

$= 4(2T(\frac{n}{4}) + \frac{n}{4})$

$= 8T(\frac{n}{8}) + 3n$

$kn = 2^k T(\frac{n}{2^k}) + kn$

$= 2^{\log_2 n}(1) + \log_2 n(n)$

$= n + \log_2 n(n)$

$= \boxed{O(n \log n)}$

(d) $f(1) = 1; f(n) = \sum_1^{n-1} (i \cdot f(i))$
(Hint: compute $f(n+1) - f(n)$ for $n > 1$)

Solution:

$f(n+1) - f(n)$

$f(n+1) = \sum_1^{n} (i + f(i)) - \sum_1^{n-1} (i f(i))$

$f(n+1) - f(n) = 2 \cdot f(n)$

$f(n+1) = nf(n) + f(n) = (n+1)fn$

$f(k) = k\, f(k-1) \rightarrow k(k+1)f(k-2)$

$= k(k+1)(k-2)f(k-3) \frac{k!}{(k-3)!}$

$f(k-1) = (k-1)f(k-2)$

$f(k-2) = (k-2)f(k-3)$

$\frac{k!}{(k-m)!} f(k-m)$

$T(k-m) = T(1)$

$k - m = 1$

$k = 1 + m$

$k - 1 = m$

$\frac{k!}{k-(k-1)}$

$= \frac{k!}{1} = k! \rightarrow (n+1)!$

$f(n+1) = O(n+1!)$

$= O(n!)$

$n! = \frac{n \cdot (n-1) \cdot (n-2)}{(n-3)!}$

Coding Question

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
```

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!
```