HW4 – Proofs and Counting

Wednesday, March 20, 2024 7:32 PM

Chapter 9 Problems:

9.4: How many alphabetical strings are there whose length is at most 5?

$26 + 26^2 + 26^3 + 26^4 + 26^5 =$

$\boxed{12356630 \text{ strings}}$

9.11: For what value(s) of k is $\binom{18}{k}$ largest? Smallest?

Since 18 is even, will be maximized at $\frac{n}{2}$ or $\frac{18}{2} = 9$, so k is largest at $\boxed{k=9}$

$\binom{18}{9} = \frac{18!}{(18-9)!\,9!} = 48620$

Will be smallest at both extreme ends, so at $\boxed{k=0 \text{ and } k=18}$

$\binom{18}{18} = \frac{18!}{(18-18)!\,18!} = 1$

$\binom{18}{0} = \frac{18!}{(18-0)!\,0!} = 1$

9.16

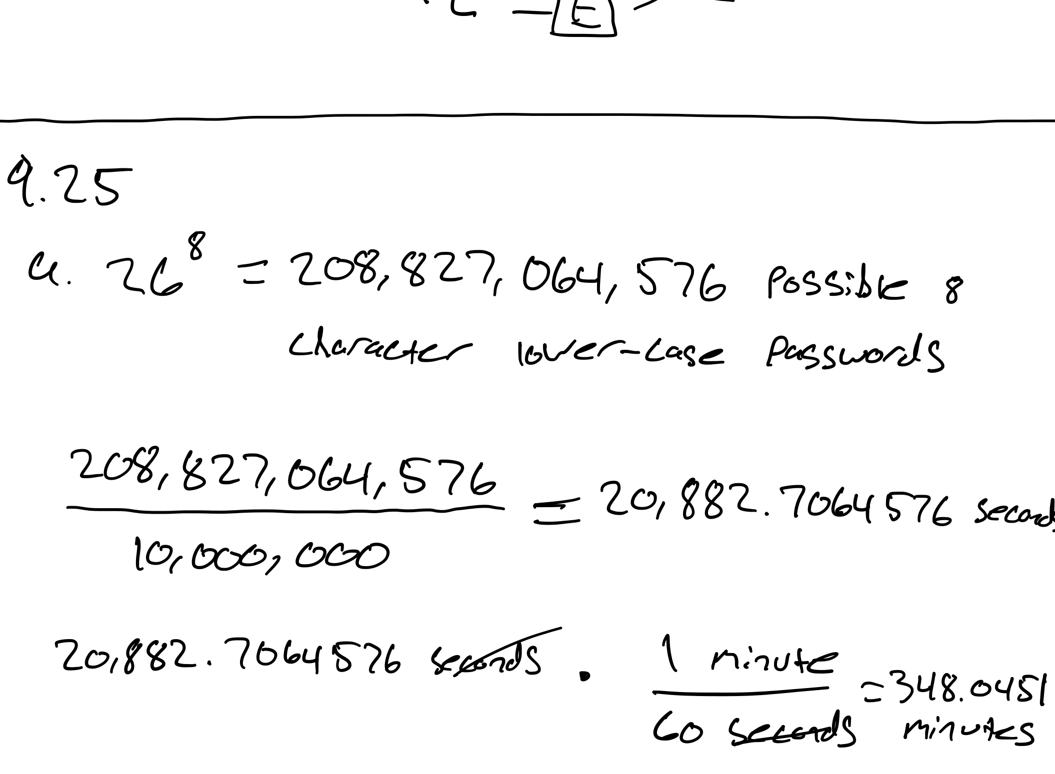
Vowels: 5 A, E, I, O, U

Consonants: 21

a. $\frac{26}{5} = 5.2$, assuming that each vowel is equally distanced from each other in an alphabet, which would give the lowest amount of consonants between each vowel, there would be a gap of 5 or 6. Moving where one vowel is will only increase another gap, so there is always going to be four consecutive consonants.

b. B, C, D, F, $\boxed{A}$, G, H, J, K, $\boxed{E}$, L, M, N, P, $\boxed{I}$, Q, R, S, T, $\boxed{O}$, V, W, X, Y, $\boxed{U}$, Z

c. Looking at the reasoning and equation in A and list in B, if we evenly spread out the vowels $(\frac{26}{5})$, the gap is 5.2. This means that if we seperate them all by 5, there will be a consonant at the end of alphabet as seen in B. If they are all connected in a circle, that last consonant will connect with the first list, making five consecutive consonants. Changing the seperation of vowels would only insure that there would be a point in the alphabet with more then five consecutive consonants. The circle with a gap of 5 between the vowels with one consonant left over to make one gap 6 is shown below.



9.25

a. $26^8 = 208{,}827{,}064{,}576$ possible 8 character lower-case passwords

$\frac{208{,}827{,}064{,}576}{10{,}000{,}000} = 20{,}882.7064576$ seconds

$20{,}882.7064576 \text{ seconds} \cdot \frac{1 \text{ minute}}{60 \text{ seconds}} = 348.0451$ minutes

$348.0451 \text{ minutes} \cdot \frac{1 \text{ hour}}{60 \text{ minutes}} = \boxed{5.801 \text{ hours}}$

b. $\underset{26^8+26^8}{52^8} = 5.345972853 \cdot 10^{13}$ Passwords

$5.345972853 \cdot 10^{13} \text{ passwords} \cdot \frac{1 \text{ second}}{10{,}000{,}000 \text{ Passwords}} =$

$5345972.853 \text{ seconds} \cdot \frac{1 \text{ minute}}{60 \text{ seconds}} = 89099.54755$ minutes

$89099.54755 \text{ minutes} \cdot \frac{1 \text{ hour}}{60 \text{ mins}} = 1484.992459$ hours

$1484.992459 \text{ hours} \cdot \frac{1 \text{ day}}{24 \text{ hours}} = \boxed{61.8746858 \text{ days}}$

c. $\underset{26^8+26^8+10^8}{62^8} = 2.183401056 \cdot 10^{14}$ Passwords

$2.183401056 \cdot 10^{14} \text{ Passwords} \cdot \frac{1 \text{ second}}{10{,}000{,}000 \text{ Passwords}} =$

$21834010.56 \text{ seconds} \cdot \frac{1 \text{ min}}{60 \text{ seconds}} = 363900.176$ mins

$363900.176 \text{ mins} \cdot \frac{1 \text{ hour}}{60 \text{ mins}} = 6065.002933$ hours

$6065.002933 \text{ hours} \cdot \frac{1 \text{ day}}{24 \text{ hours}} = \boxed{252.7084555 \text{ days}}$

d. $\underset{26^8+26^8+10^8+32^8}{94^8} = 6.095689385 \cdot 10^{15}$ Passwords

$6.095689385 \cdot 10^{15} \text{ Passwords} \cdot \frac{1 \text{ second}}{10{,}000{,}000 \text{ Passwords}} =$

$609568938.5 \text{ seconds} \cdot \frac{1 \text{ min}}{60 \text{ sec}} = 10159482.31$ minutes

$10159482.31 \text{ minutes} \cdot \frac{1 \text{ hour}}{60 \text{ mins}} = 169324.7052$ hours

$169324.7052 \text{ hours} \cdot \frac{1 \text{ day}}{24 \text{ hours}} = 7055.196048$ days

$7055.196048 \text{ days} \cdot \frac{1 \text{ year}}{365 \text{ days}} = \boxed{19.32930424 \text{ years}}$

9.28

b. $2^{32} + 2^{16} + 2^8 + 2^4 + 2^2 + 2^1 = 2^{63}$ (choosing winners of all 63 games)

$\boxed{2^{63} = 9.223372037 \cdot 10^{18} \text{ Possible brackets}}$

c. No, there are more possible brackets than people in this case, so it is not guarenteed. To guarentee a match in a bracket, by the Pigeonhole Principle, there would need to be $9.223372037 \cdot 10^{18} + 1$ people who filled out a bracket.

d. $\frac{7{,}000{,}000{,}000 \cdot k}{7{,}000{,}000{,}000} = \frac{9.223372037 \cdot 10^{18}}{7{,}000{,}000{,}000}$

$k = 1{,}317{,}624{,}577$

If everyone in the world filled out different brackets without repeats, each person would have to fill out 1,317,624,577 brackets to guarentee a perfect bracket.

Chapter 8 Problems:

8.4: Prove that the number of binary palindromes of length $2k+1$ (odd length) is 2^{k+1} for all $k \geq 0$.

Base case:

for $k=0$, $2(0)+1 = 1$ $2^{0+1} = 2^1 = 2$ "0" and "1", so holds true for base case

Hypothesis: holds true for all values $k < x$

Inductive step: Show statement holds for $x+1$

$2(x+1)+1 = 2^{(x+1)+1}$

$2(x+1)+1$ holds true when appending a bit to both sides of palindrome

ex: $2x+1$ $x=2$ $2(2)+1 = 5$ 10001

$2(x+1)+1$ $x=2$ $2(2+1)+1 = 7$ 1100011

We can either append 0 on both ends or 1 on both ends, so for each palindrome of length $2x+1$, we can create 2 new palindromes of length $2(x+1)+1$. That means the total number of palindromes of length $2(x+1)+1$ is twice the number of palindromes of length $2x+1$, which is 2^{x+1}. That means $2(x+1)+1 = 2 \cdot 2^{x+1}$.

$2(x+1)+1 = 2 \cdot 2^{x+1} = 2^{x+1+1}$ ✓

This proves that the number of binary palindromes of length $2k+1$ (odd length) is 2^{k+1} for all $k \geq 0$.

8.13

a. halfIt(n) returns the number of times n is halved before it becomes 1, plus 1 to become 0, which is $\log_2(n)+1$ for all $n > 0$. Returns 0 for $n=0$.

b. Base case: $n=1$, $\log_2(1)+1 = 1$, so holds true for base case.

Hypothesis: holds true for all values $n < x$

Inductive step: Base case $= 1$, so need that 1 plus number of additional halves needed.

$1 + (\log_2(\frac{x}{2}) + 1) = 1 + (\log_2(x) - \log_2(2) + 1) =$

$1 + (\log_2(x) - 1 + 1) = \log_2(x) + 1$ ✓

This proves that halfIt(n) returns $\log_2(x)+1$ for all $n > 0$.

c. $\text{halfIt}(n) = \log_2(x) + 1$, since this is a recursive algorithm, a recurrence relation can be defined to find the complexity of the algorithm.

The cost of $\text{halfIt}(n) = \text{cost}(n) = \text{cost}(\frac{n}{2}) + c$

$\text{cost}(\frac{n}{2}) + c$ unwind and simplify

$= (\text{cost}(\frac{n}{4}) + c) + c$

$= \text{cost}(\frac{n}{4}) + 2c$ unwind and simplify

$= (\text{cost}(\frac{n}{8}) + c) + 2c$

$= \text{cost}(\frac{n}{8}) + 3c$ can continue to unwind and simplify $\log_2 n$ times

$= \text{cost}(1) + (\log_2 n) \cdot c$

$= c \log_2 n + c'$ where $c' = \text{cost}(1)$

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

As proved in the recurrence relation above, the complexity of $\text{halfIt}(n) = O(\log n)$.

8.15

(proved in textbook)

Total number of moves $= 2^n - 1$ with 64 disks $= 2^{64} - 1 = 1.844674407 \cdot 10^{19}$ moves

With 1 move every second it will take

$\frac{1.844674407 \cdot 10^{19} \text{ moves}}{\text{seconds in a year} = 60 \cdot 60 \cdot 24 \cdot 365} = 5.849424174 \cdot 10^{11}$ years

They started 6000 years ago, so the world will end in $5.849424174 \cdot 10^{11} - 6000 =$

$\boxed{5.849424114 \cdot 10^{11} \text{ years}}$