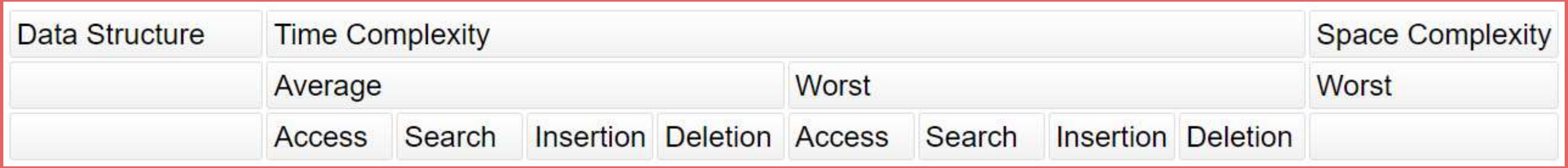
Data Structures & Algorithms



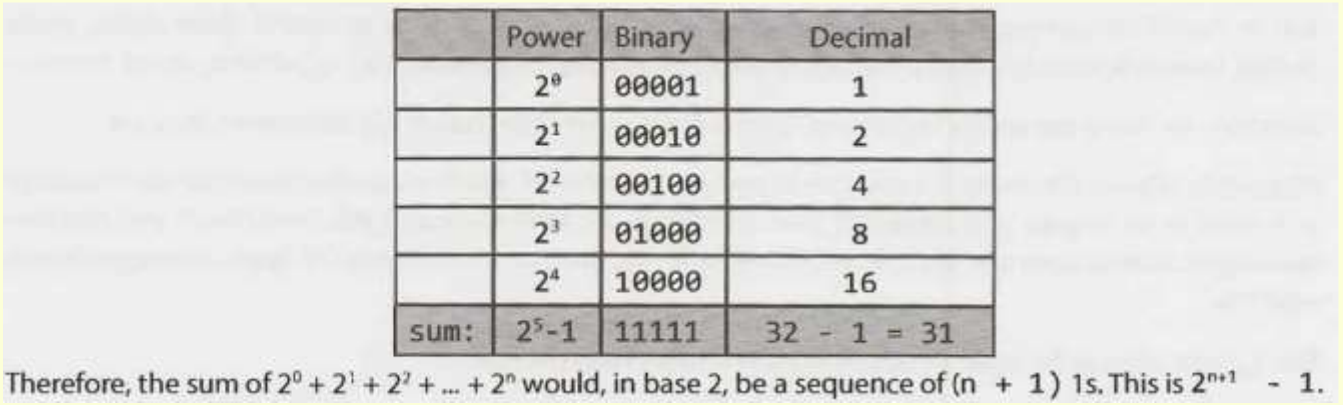
Data Structure	Time Complexity								Space Complexity
	Average				Worst				Worst
	Access	Search	Insertion	Deletion	Access	Search	Insertion	Deletion	

Time/Space Complexity

What is the big-O for the following series:
1 + 2 + 3 + … + n = ?
1 + 2 + 4 + 8 + … + n = ?
2^0 + 2^1 + 2^2 + … + 2^n = ?

1+2+...+n = (n(n+1))/2 = O(n^2)

1 + 2 + 4 + 8 + … + n = O(2n) = O(n)



	Power	Binary	Decimal
	2^0	00001	1
	2^1	00010	2
	2^2	00100	4
	2^3	01000	8
	2^4	10000	16
sum:	2^5-1	11111	32 - 1 = 31

Therefore, the sum of $2^0 + 2^1 + 2^7 + ... + 2^n$ would, in base 2, be a sequence of (n + 1) 1s. This is $2^{n+1} - 1$.

What is the difference between combinations and permutations, and how do you compute them? What's the intuition behind the equation?

Permutations: Order matters

- Represents multiplying n*(n-1)*...*(n-k-1)
- If n=k, becomes n!

$$P_k^n = \frac{n!}{(n-k)!}$$

Combinations: Order invariant

- Same as permutation, but divide out the duplicates with same elements but different ordering
- If n=k, becomes 1

$$C_k^n = \frac{n!}{k!(n-k)!}$$

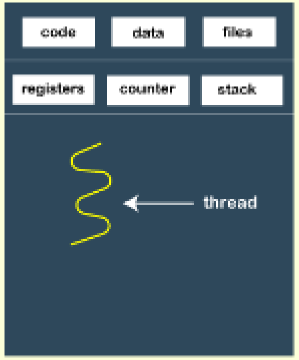
What is the time complexity of nCk?

- In general, O(n^min{k,n-k})
- In most cases, k is very small, so O(n^k)

Misc

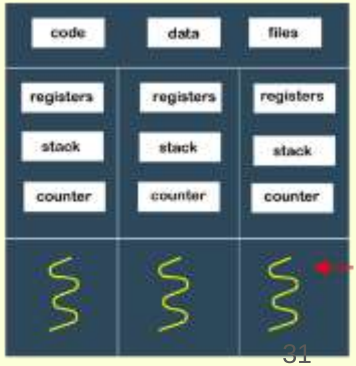
What are processes and threads?

- Each process must start with least one thread, but can also later create multiple.
- Processes/threads are implemented on the OS level, and uses the underlying CPU's cores/processors
  - For example, a Intel Core i5-1335U has 10 cores and 12 hardware threads. This means that 10 processes can be run at once (true parallel), and 12 threads scheduled at once. You can have more software processes/threads – they will be scheduled using context switching.
- In CPython, the Global Interpreter Lock (GIL) allows only one thread to hold control of the Python interpreter.
  - So, only one thread can be executing at any point in time. This was done for simplicity/compatibility
  - However, other python implementations don't have a GIL



Single-threaded process

	Processes	Threads
Memory Space	Each has a separate memory space	Threads of a process run in a shared memory space
Communication	Harder to share objects between processes; more isolated from each other. Need inter-process communication (IPC)	Easier to share objects in the same memory, but need to be careful to avoid race conditions
Python Package	threading	multiprocessing
Overhead	Larger memory footprint	Lightweight, low memory footprint
True parallelism?	In python, takes advantages of multiple CPUs/cores	In python, threads cannot be run in parallel using multiple CPUs/cores, due to the GIL.
Use Case	**Use case in python:** For when you want to really do more than one thing at a given time on the CPU (CPU-bound).	**Use case in python:** enables applications to be responsive, when execution is I/O bound (eg from internet, database retrieval, etc). Things aren't being done in parallel on the CPU, but concurrently due to context switching.



Multi-threaded process

Question Goes Here

- Answer goes here