Algo Review Part 2

By P'Time

Recursion

- Why we use it?
  - To keep it simply, recursion is just easy to understand, shorter to code and cleaner to read
- When to use it?
  - When there is no other better approach or such the algorithm complexity is the same
  - When the problem is not too strict about the space management
- When to not use it?
  - When there is better approach or using recursion would take too much spaces

Hashing

- Linear probing
  - Simplest way to hash and produce quite good result of hash table
  - Can lead to slow searching and inserting but still better than Quadratic probing
- Key dependent probing
  - Good or bad it depends on hashing function
- Separate chaining or hashing buckets
  - Best hashing algorithm so far extremely good when inserting also quite good when searching

Memory Management

- External and Internal Fragmentations
- Sequential and Non-Sequential Fit methods
- Garbage Collections
- Mark and Sweep
- Compaction
- Incremental Garbage Collection

Fragmentations

- External Fragmentation

– Occurs when memory allocated and deallocated some of it leaving gaps between used memory

- Internal Fragmentation

– Occurs when process request of memory is larger than it needed

Sequential Fit methods

- Inefficient method not widely used
- First fit
  - Allocated first suitable block found in the memory. Most efficient however external fragmentation likely to be occurred
- Best fit
  - Allocated most suitable block found in the memory. Good efficient no need to worry about fragmentation problem
- Worst fit
  - Allocated the largest block found in the memory. Worst efficient

Non-Sequential Fit methods

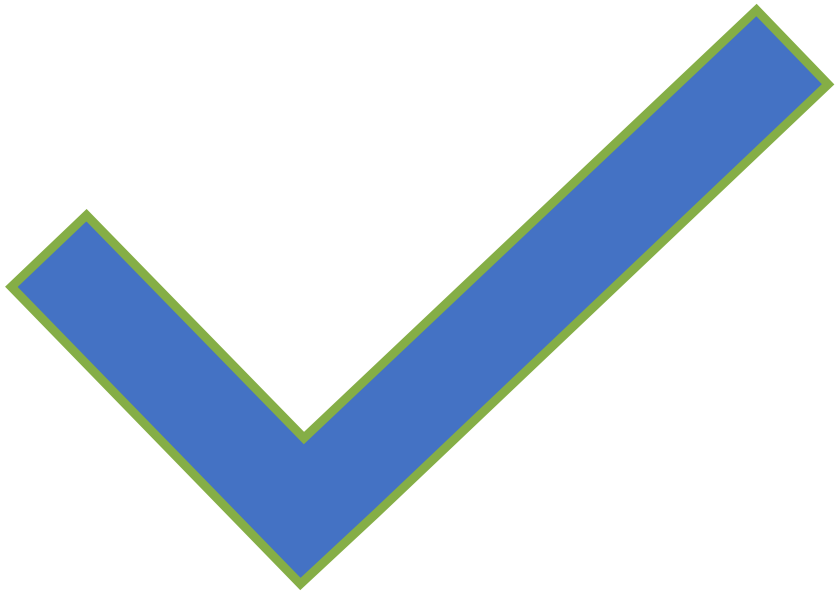
- Efficient way of managing memory widely uses method
- Buddy System (aka Binary Buddy System)
  - Divide memory into 2 sections and each of them can be divided further by 2
  - Whenever possible sections combine to make a larger section
  - If needed a section can be divided to smaller sections
  - Internal Fragmentation can be occurred

Garbage collection

- Area that no longer use will be return to the memory
- Mark and sweep method
  - Mark the used memory location
  - Delete the unmarked memory location this can lead to gaps between used memory
  - Data compaction -> copy data that sparsely scattered in memory to be more tightly locate to each other
- Incremental garbage collection
  - Instead of just interrupting the program to do mark and sweep, this method copy the data to another memory location
  - And do the mark and sweep method there instead, so that the program can keep running as it do its work

String management

- Brute force (Worst)
  - Check with the pattern if it incorrect move to next character check with the pattern again
- Hancart (Not bad)
  - If wrong move to the next i - j + 2 characters
- Knuth Morris Pratt (Good)
  - If wrong move to the next j - longest suffix characters



That's it