Algorithm → Analysis

Search
⇒ linear search → O(N) Ω(1)
⇒ Binary Search O(log n)

Sort
bubble sort → O(N²) Ω(N)
selection sort
Merge Sort → O(n log n)
Complexity time
Space time O(n)

Travelling Salesperson (Talbot)

O(N!)

5×4×3×2×1 = 120

To find the minimum Path

Breadth First Search
A* Algorithm
① Shortest Path A, B, ?
② Shortest Path

Greedy Algorithm → "Optimizations" Find the Optimal Solution

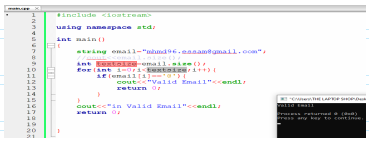
Cashier
"Minimize Number of Coins"
⇒ 100
⇒ 3
97 Coins
50 20 20 5 1 1
5 coins

Dollar → 100¢
Half dollar → 50¢
Quarter → 25¢
Dime → 10¢
Nickel → 5¢
Penny → 1¢

Pseudo Code:
buy 2.7 dollar
Pay 5$
① 5×100 − 2.7×100 = 230¢
② 230/100 = 2 (dollar = 2)
③ 230% 100 = 30
5 strong
2 People 2

Modulus
10/3 = 3
10%3 = 1

Dynamic Programing: saving Computed Results for Reuse later.



Recursion

Recursive Case → Item → Base Case

```
int countdown(int n){
    cout<<n<<endl;
    if(n<=0){
        return 0;
    }
    else{
        return countdown(n-1);
    }
}

int main()
{
    countdown(10);
}
```

Factorial n = 5 → 5×4×3×2×1 — stop
n-1

```
int factorial(int n){
    if(n==1){
        return 1;
    }
    else{
        return n*factorial(n-1);
    }
}

int main()
{
    cout<<factorial(5);
}
```