Todays Content

1. Intro to Algorithms
2. DS/Algo Intro with Dijkstra's
3. Factors count optimization.

# Data Structure & Algorithm

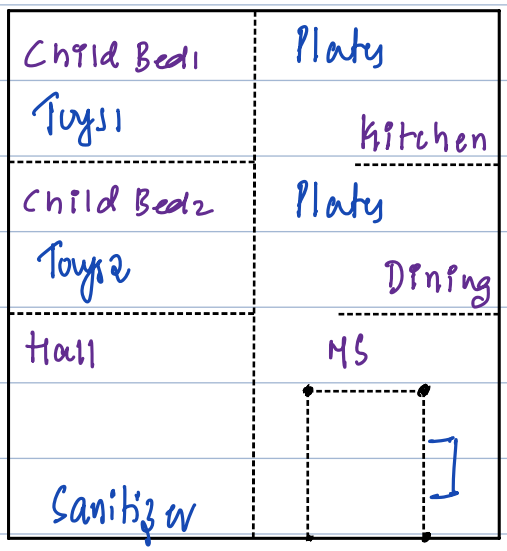
Algorithm: Step by Step process to do a task.

## Get Money from ATM

Steps:

- Enter ATM
- Insert debit card
- Enter Amount
- Enter correct Pin
- Collect Cash
- Press Cancel
- Exit ATM

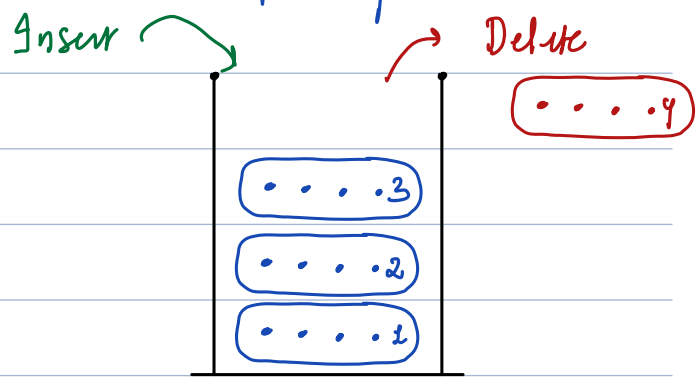
## Datastructure:




Ex: Datastructure

Stack / Linked List / Queue / Tries / Trees..

Stack: Insert & Delete from top.




Note: Arranging stuff based on our use case?

↳ Arranging data as per your requirement in memory in different structures is data structure

Stacks: 1. Recursion
2. Undo / Redo
↔
3. Evaluate expression

# Q: Fire = Petrol Bunk

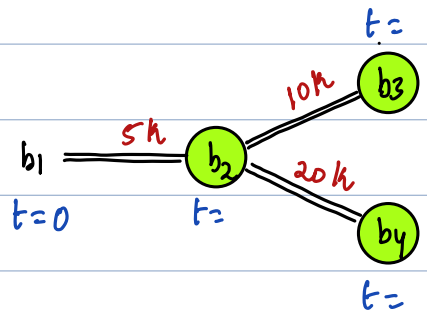
● → Representing petrol pump  ↗ Pipe filled with Petrol

a. ——— line indicates lengths of petrol pipe between 2 bunks

b. Initially say bunk 1 blasted  c. Petrol burns at 1km/min.

d. Calculate time at which each bunk is blasted ?

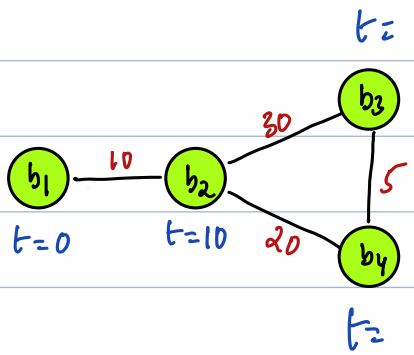
Eg1:




bunks : $b_1$ $b_2$ $b_3$ $b_4$

Time :

Eg2:

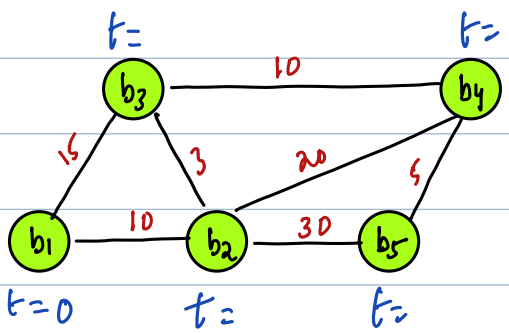



bunks : $b_1$ $b_2$ $b_3$ $b_4$

Time :

Note: If multiple ● are reaching same bunk we consider ● time.

Eg3:




bunks: $b_1$ $b_2$ $b_3$ $b_4$ $b_5$

Time :

Steps:

1.

2.

Q1) Given N > 0, return no: of factrs of N?

factor? Q: Is 4 a factr of 24? Yes

Q: Check if i is a factr of N? N%.i==0

Countfactrs:

Q1: N=24 factrs = 1 2 3 4 6 8 12 24 : 8 factrs

Q2: N=10 factrs = 1 2 5 10 : 4 factrs.

obs: All Factrs of N: {1,.. N}

```
int countfactr(int N){
    int c=0;
    for(int i=1; i<=N; i++){ // i=1 2 3 .. N : N iterations
        if(N%.i==0){
            c++;
        }
    }
    return c;
}
```

Online: 1 sec = $10^8$ iterations/sec ↳ Code loops.

Iterations: N iterations

| Input N | Iterations | Execution Time |
|---|---|---|
| $N=10^9$ | $10^9$ iterations | $10^9 * \frac{1}{10^8} = 10$ sec |
| $N=10^{18}$ | $10^{18}$ iterations | $10^{18} * \frac{1}{10^8} = 10^{10}$ sec = 317 years |

$1^{st} \to 2^{nd} \to 3^{rd} \to 4^{th} \to 5^{th} \to 6^{th}/7^{th}$

Note: $\frac{a^m}{a^n} = a^{m-n}$

$a^m * a^n = a^{m+n}$

Unitary Method:

6 apples = 60 rupees Ass: $10^8$ iterations = 1 sec 1 iteration = $1/10^8$

1 apple = 60/6 = 10

15 apples = 150 rupees

Observations:

1. $i * j = N$

   Both i & j are factor of N

   $j = N/i$

   Con: i & N/i factor of N

   Fin: if i is factor of N

   N/i is factor of N.

eg:

| N | i | N/i |
|---|---|---|
| 24 | 3 | 8 |
| 40 | 5 | 8 |

N = 24

| i | c = | N/i |
|---|---|---|
| 1 | c = | 24 |
| 2 | c = | 12 |
| 3 | c = | 8 |
| 4 | c = | 6 |
| 6 | 7 | 4 |
| 8 | 7 | 3 |
| 12 | 7 | 2 |
| 24 | 7 | 1 |

obs:

i = 1.

- $i * c = N/i$
- $i^2 * c = N$ sqrt on both
- $\sqrt{i^2} * c = \sqrt{N}$
- $i * c = \sqrt{N}$

i = 1 ; $i * c = \sqrt{N}$ ; i ↑↑ j

⇓

$i^2 * c = N$

N = 36

| i | c = | N/i |
|---|---|---|
| 1 | c = | 36 |
| 2 | c = | 18 |
| 3 | c = | 12 |
| 4 | c = | 9 |
| 6 | c = | 6 |
| 9 | 7 | 4 |
| 12 | 7 | 3 |
| 18 | 7 | 2 |
| 36 | 7 | 1 |

N = 15 : c = 0

| i = 1 | $i * i * c = N$ | N%i == 0 | i & N/i |
|---|---|---|---|
| 1 | $1 * 1 * c = 15$ | 15%1 == 0 | 1 & 15 c = c+2 |
| 2 | $2 * 2 * c = 15$ | 15%2 == 0 | |
| 3 | $3 * 3 * c = 15$ | 15%3 == 0 | 3 & 5 c = c+2 |
| 4 | $4 * 4 * c = 15$ | break | |

return c; 4.

```
int countFactorsOpt(int N){
    int c=0;
    for(int i=1; i*i<=N; i++){
        if(N%i==0){
            // i is fawr, N/i is fawr
            if(i==N/i){
                c=c+1;
            }else{
                c=c+2;
            }
        }
    }
    return c;
}
```

N = 16.

| i=1 | i*i<=16 | N%i==0 | i & N/i |
|---|---|---|---|
| 1 | 1*1<=16 | 16%1==0 | 1 & 16 c=c+2 |
| 2 | 2*2<=16 | 16%2==0 | 2 & 8 c=c+2 |
| 3 | 3*3<=16 | 16%3==0 | |
| 4 | 4*4<=16 | 16%4==0 | 4 & 4 c=c+2; * |

i==N/i c=c+1

for(int i=1; i*i<=N; i++){ ⇒ i=1; i<=√N; i++ ⇒ {1 2 3 .. √N} ⇒ √N iterations

| Input | Iterations | Execution Time |
|---|---|---|
| $N=10^{18}$ | $\sqrt{10^{18}} = 10^9$ iterations : $10^9 * \frac{1}{10^8} = 10$sec | |

Ass: $10^8$ iterations = 1 sec  1 iteration = $1/10^8$

$10^{18} = (10^9)^2$ : $\sqrt{(10^9)^2} = 10^9$