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 solve a task.

Get Money from ATM

Steps:

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

Data structure:

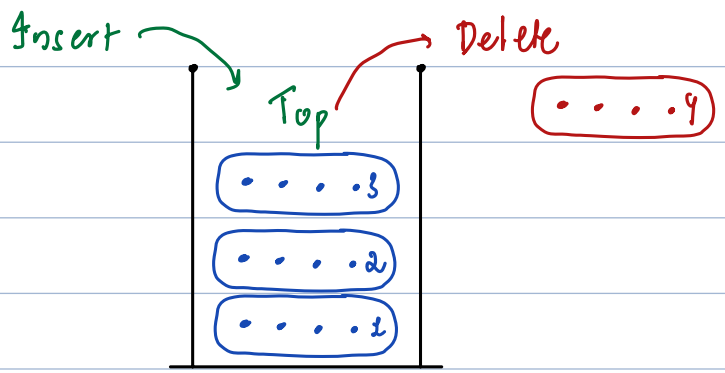
Child Bed1	Maseda's
Toys1	Kitchen
Child Bed2	Plates
Toys2	Dining
Hall	
	Drum
sanitizer	

Auto complete

Photo/Playlist

Ex: Data structure : Stack / Linked / Trees / Tries / ...

Stack: Insert & Delete occur at top.



Use case: 1. Undo Redo 2. ← → 3. Expression Evaluation

Note: Arranging stuff based on our use case?

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

Q: Fire = Petrol Bunk

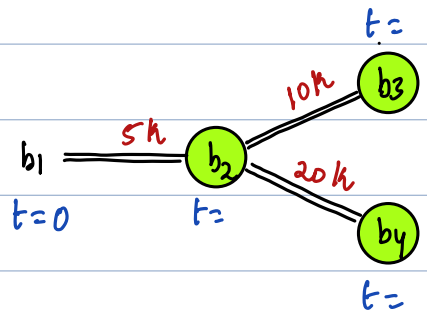
● → Representing petrol pump ↗ Pipe filled with Petrol

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 ?

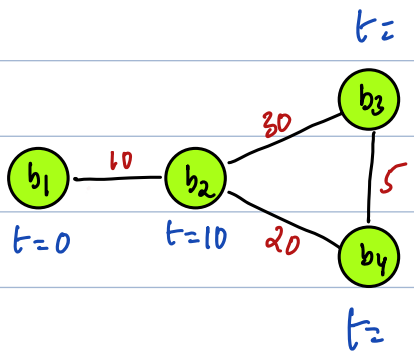
Eg1:



bunks : b_1 b_2 b_3 b_4

Time :

Eg2:

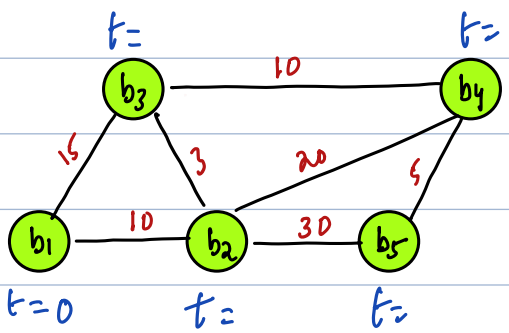


bunks : b_1 b_2 b_3 b_4

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 → modulu operatr.

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.

N: {small 1 : largest N}

Idea: for every number from 1..N check if numbr is factr or not.

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

Online = 10^8 iteratns / sec

↳ Code loops.

Iterations : N

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

Input N	Iterations	Execution Time
$N=10^9$	10^9 iteratns	$10^9 * \frac{1}{10^8} = \frac{10^9}{10^8} = 10$ sec
$N=10^{18}$	10^{18} iteratns	$10^{18} * \frac{1}{10^8} = \frac{10^{18}}{10^8} = 10^{10}$ sec = 317 years → 158 years

1st → 2nd → 3rd → 4th → 5th → 6th/7th

Unitary Method:

6 apples = 60 rupees

1 apple = 60/6 = 10

15 apples = 15*10 = 150

10^8 iteratns = 1 sec

1 iteratn = $1/10^8$

Observations: Ass:

1. $i * j = N$

   i & j are factors of N.

   $j = N/i$

   i & N/i are factors of N

Con: if i is factor of N

N/i is factor of N.

Eg:

N	i	N/i
16	2	16/2 = 8
24	4	24/4 = 6

N = 24

i	< =	N/i
1	< =	24
2	< =	12
3	< =	8
4	< =	6
6	>	4
8	>	3
12	>	2
24	>	1

Con: Iterate in green part

i = 1. Small

$i < N/i$

$\Rightarrow i^2 < N$

Sqrt in both

$\Rightarrow \sqrt{i^2} < \sqrt{N}$

$i < \sqrt{N}$: longer

i = 1; $i < \sqrt{N}$; i++

$i^2 < N$

N = 36

i	<	N/i
1	< =	36
2	< =	18
3	< =	12
4	< =	9
6	< =	6
9	>	4
12	>	3
18	>	2
36	>	1

N = 15 : C = 0

i = 1	$i * i < N$	N%i == 0	i & N/i
1	$1 * 1 < 15$	15%1 == 0	1 & 15 C = C + 2
2	$2 * 2 < 15$	15%2 == 0	
3	$3 * 3 < 15$	15%3 == 0	3 & 5 C = C + 2
4	$4 * 4 < 15$	Stop	

return C; // 4.

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

N = 16 c = 0

i=1	i*i<=N	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+1
			i==N/i	c=c+1

Iterating: i=1; i*i<=N; i^2<=N; i<=√N; i++; {i: 1 2 .. √N}

Input	Iterations	Execution Time
$N=10^{18}$	$\sqrt{10^{18}}=\sqrt{(10^9)^2}=10^9$ iterations $=\frac{10^9}{10^8}*1=10$ sec	