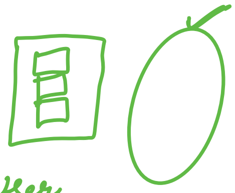
I/P ⟶ O/P

I/P → Array of length N ✓

TC ∝ f (size of input) ✓

↳ Amount of time taken by an algo. to run as function of size of input.



x → 10 machines/servers ✓

⇓

✓ x → 3 machines ✓ ∝ cost ✓

User Experience ✓

Q→ check if given input N is prime?

↳ A +ve number with exactly 2 factors 1 & N.

1 → only 1 factor ×

check if 'i' is a factor of N?

((N%i)==0) ✓

$\boxed{\text{Bruteforce}}$

-1 → 1, → -1

```
factors = 0;
for (i = 1 ; i <= N ; i++) {          1—10
    if (N % i == 0) factors++;
}
return (factors == 2);
```

	a b
	↓ ↓
N=10	1*10 2*5	{1, 2, 5, 10}
N=20	1*20 2*10 4*5	{1, 2, 4, 5, 10, 20}

$\boxed{N=13 \quad 1*13}$ ⇒ prime

N=25

1*25

5*5

$\sqrt[3]{25} = 2...$

N = 10

steps = 10 ✓

$\boxed{N = a*b}$ $\boxed{a <= b}$

$b = \frac{N}{a}$

(>1) ⇒ N is not prime ✓

$a <= b$

$a <= \frac{N}{a} \Rightarrow a^2 <= N$

$\Rightarrow \boxed{a <= \sqrt{N}}$

```
✓ for (a = 2; a*a <= N; a++) {
      if (N % a == 0)
          return false;
  }
  return true;   ✓
```

N=10

steps = $\sqrt{N} = \sqrt{10} = 3$ ✓

$A_1 \rightarrow 2^N$ steps

$\boxed{A_2 \rightarrow N^3 \text{ steps}}$ ✓

1 step = 1 ms

$\boxed{N=2}$	$2^2 = 4$ ✓	$2^3 = 8$
$\boxed{N=5}$	$2^5 = 32$ ✓	$5^3 = 125$
$\boxed{N=10}$	$2^{10} = 1024$	$10^3 = 1000$ ✓
$\boxed{N=100}$	$2^{100} \approx 10^{30}$	$100^3 = 10^6$ ✓

very large

$TC = O(2^N)$

$TC = O(N^3)$

$\boxed{\text{Rate of growth of time wrt input size}}$ ✓

$N \rightarrow \boxed{N^{1.1}} + 2 + 3N - \sqrt{N}$

$TC = O(N^{1.1})$

$N \rightarrow \cancel{2}N^2 + \cancel{N} - \cancel{2}$

$TC = O(N^2)$

$N \rightarrow \cancel{2} - \cancel{3}N + \cancel{\sqrt{N}}$

$TC = O(N)$

Quiz

```
1) for(i=0; i<N; i+=2) {
           print(i);
   }
```

0, 2, 4, 6 - - - N

$\frac{N}{2}$ steps $= \cancel{\frac{1}{2}} * N$

$TC = O(N)$

```
2) for(i=0; i<N; i++) {
       for(j=i+1; j<N; j++) {
           ans += i*j;
       }
   }
```

$i=0 \quad j \rightarrow 1 — N$

$i=1 \quad j \rightarrow 2 — N$

$i=2 \quad j \rightarrow 3 — N$

$\vdots$

$N-1 \quad j \rightarrow N-1 — N$

$N + (N-2) + (N-3) + \ldots 1 = \sum N$

$= \frac{N * (N+1)}{2}$

$\frac{N^2}{2} + \frac{N}{2}$

$\frac{1}{2}N^2 + \frac{1}{2}N$

$TC = O(N^2)$ ✓

```
3)    j=0;
      for(i=0; i<N; i++){
         while(j<N && a[i]<=a[j]){j++;}
         sum+=a[j];
      }
```

✓N times

i → N times

j → N times

N+N = 2N

j=0+2... N ✓

TC=O(N) ✓

```
4)  i=N;  a=0;
    while(i>0){ a+=i; i/=2; }
```

1/2 = 0 ✓

$N \to \frac{N}{2} \to \frac{N}{2^2} \to \frac{N}{2^3} \cdots \boxed{\frac{N}{2^k}=1}$

after k steps

$\log_b(b^k)=k$ ✓

$\Rightarrow \boxed{N=2^k} \Rightarrow k=\log_2(N)$

TC=O(log(N)) ✓

$O(\log_2(N))$

$\log_b(a)=\frac{\log(a)}{\log(b)}$

$\log_2 N = \frac{\log(N)}{\log(2)} = \frac{1}{\log(2)} * \log(N)$

```
4)  for(i=1; i<N; i=i*2){ print(i); }
```

$1 \to 2 \to 2^2 \to 2^3 \to \cdots \to 2^k = N$

after k steps

$k \approx \log_2(N)$

TC=O(log(N))

```
5)  k=0;
    for(i=0; k<N; i++){
             k=k+i;
    }
```

i→ 0 1 2 3

$k=0+0+(1)+(2)+(3)$

$(1+2+3+\cdots x) = N$

#steps = x

$\Rightarrow \frac{x*(x+1)}{2} = N$

$\Rightarrow x^2+x=2N$

$\Rightarrow x^2+x-2N=0$

$x = \frac{-1+\sqrt{1-4(1)(-2N)}}{2} = \frac{-1+\sqrt{1+8N}}{2}$

$ax^2+bx+c=0$

$x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$

TC=O(√N) ✓

Space Complexity → Extra space to convert given i/p to desired o/p.

$\propto f(\text{size of input})$

No extra space used ⇒ $SC = O(1)$

Q→ For a given input array. Find the product of freq. of every element in the array?

$|A| = N <= 10^5$

$1 \leq A[i] = X <= 10^5$ → |F|

f(A, N, X){

↓

}

Eg→ A → [3 2 1 1 3 5 5 5] (indices 0 1 2 3 4 5 6 7)

A[i] →	freq
3 →	2
2 →	1
1 →	2
5 →	3
4 →	0

$Ans = 2*1*2*3 = \underline{12}$

```
∀ i  F[i]=0          → SC=O(X)
for( i=0 ; i<N ; i++){ F[A[i]]++; }
ans=1;
for (i=0 ; i<X ; i++){
    if(F[i]>0)    ans = ans* F[i];
}
return ans;
```

Find freq. of all elements in A[].

F[] → output

$SC = O(X)$ → incorrect

$SC = O(1)$

int a = 5;

4 byte = 4*8 = 32 bits

5 = 00---0---00000101

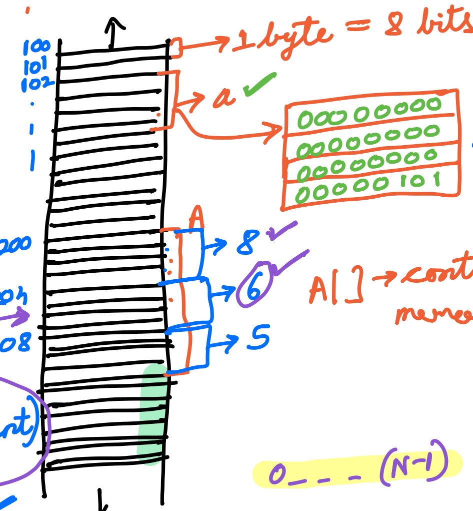
int A[] = {8, 6, 5};

4*3 = 12 bytes

A[1] = base address + (1*size of int)

= 200 + (1*4) = 204

TC=O(1)



A[] → continuous memory allocation

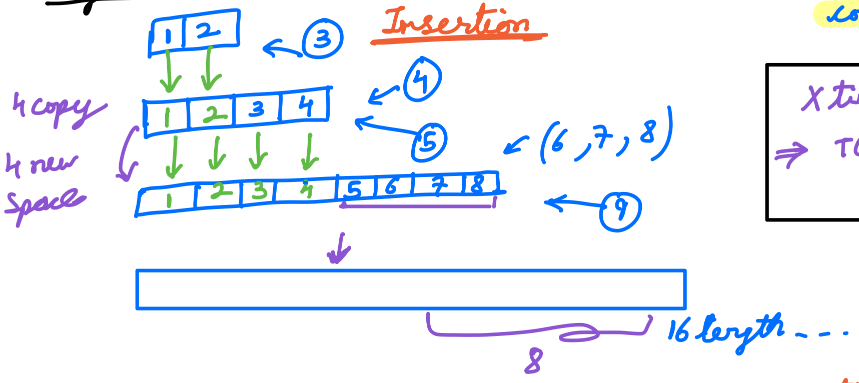
0 _ _ _ (N-1)

Dynamic Array → ArrayList/Vector

TC = O(1) Access time ✓

continuous memory ✓

Insertion



X time to copy
⇒ TC=O(1) for X insertions after copy.

TC ≈ O(2) ✓

TC = O(1) ✓

✓ Inserting X elements is taking 2X time