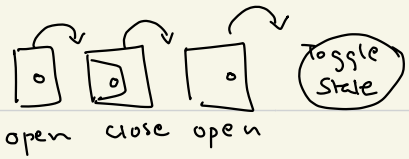
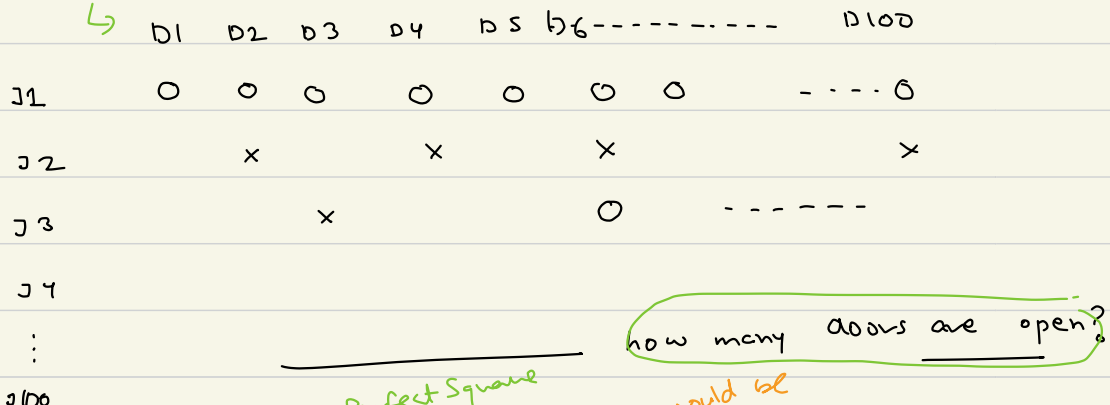
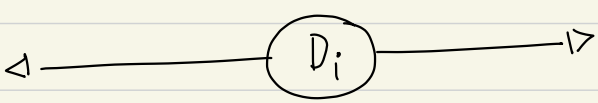
Lecture - 2



warmup · Circular Jail

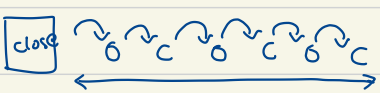




Door 12 → 1, 2, 3, 4, 6, 12 ✓ ✓ ✓ ✓ ✓ ✓

even

6 Rounds

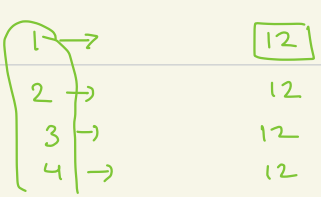


Toggling a door even times will never change state of the door

→ odd of factors of Di → open

Jailer

Door → 12



Div => Rounds

D1 → 1

D2 → 1, 2

D3 → 1, 3

D4 → 1, 2, 4 → 3 → odd

D5 → 1, 5 → even

D6 → 1, 2, 3, 6 → even

D7 → 1, 7 → even

D8 → 1, 2, 4, 8 → even

D9 → 1, 3, 9 → 3 → odd

D10 → 1, 2, 5, 10

6 →
12 →
→ ~~open~~ closed

a × b = 30

even

1 × 30 = 30
2 × 15 = 30
3 × 10 = 30
5 × 6 = 30

Non-perfect Sq → 30 → 1, 2, 3, 5, 6, 10, 15, 30 → pairs → even

perfect Square → 36 → 1, 2, 3, 4, 6, 9, 12, 18, 36 → 2x+1 → odd

odd

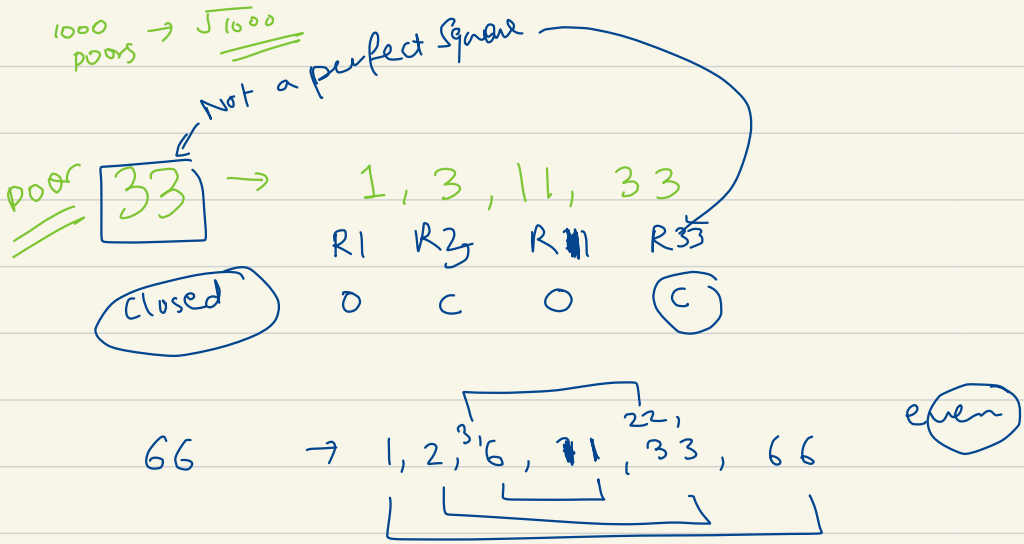
$\sqrt{N} \times \sqrt{N} = N$

$1^2, 2^2, 3^2, 4^2, 5^2, 6^2, 7^2, 8^2, 9^2, 10^2$

10 doors

→ 1, 4, 9, 16, 25, 36, 49, 64, 81, 100 ← open

open



Pseudo Code

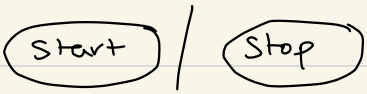
→ Human readable desc. of an algorithm.

↓

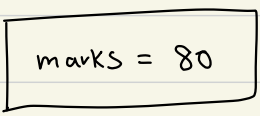
Sequence of steps in a particular order to solve a problem

- Lang Independent
- Structure
- Fast / verify / Review

Instructions

1. 

Start / Exit

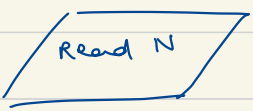
2. 

Assignment

marks = 80

SI = P*R*T /100

3. Input / Output



5
N

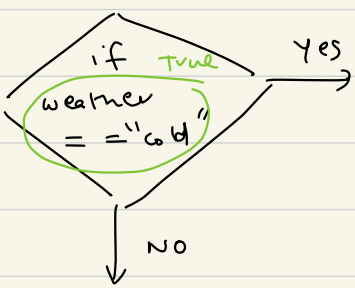
Read N

Print N

Print " Hello "

double quotes

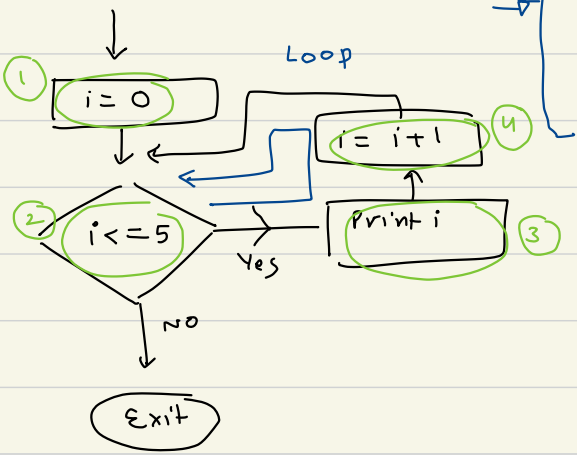
Print " N "

4. 

True

```
if (Condition) {
    ===
    Sweater
}
else {
    ===
}
```

5 Repeat




```
i = 0          ← init        ↙ stop
while (i <= 5) {←
    ==
    (3)
    i = i + 1   ← update
}
```

for Loop

```
while  condition  do
    ==
endwhile
```

→ X = int("5")

for ↔ while

(1) Flowchart → (2) Pseudocode (3) Code (Java)

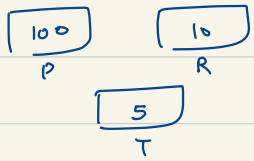
Given P, R, T and we want to compute SI

1 Read P, R, T

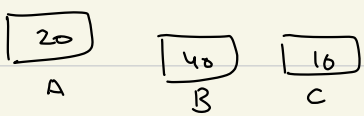
2 SI = $\frac{P*R*T}{100}$

3 Print SI

4 Exit



Largest of 3 No's



1. Read A, B, C

2 if A >= B and B >= C {

print "A is largest"

exit

3

3 if ~~B >= A~~ and B >= C {

print "B is largest"

exit

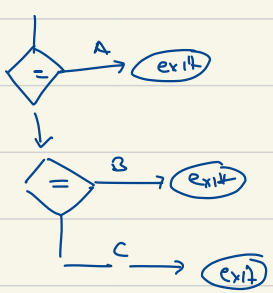
3

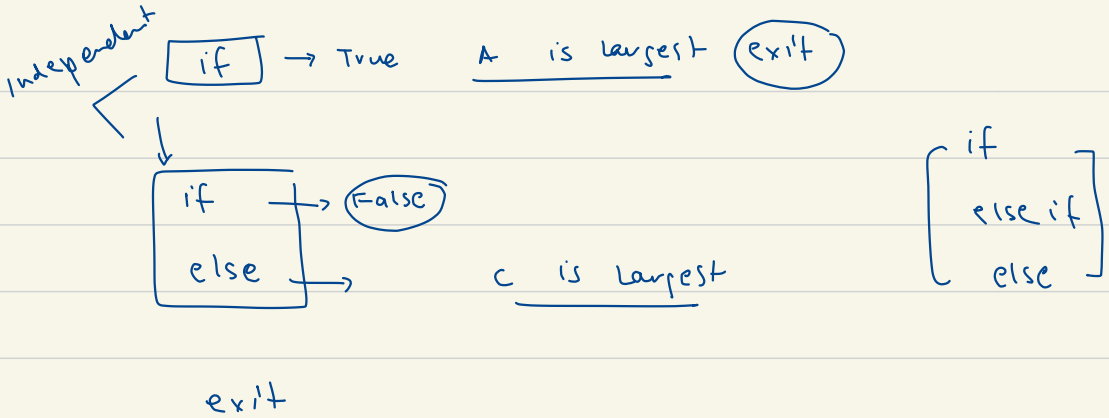
4 else {

print "C is largest"

exit

3





#3 Read N, Numbers (followed by N), and find their smallest. & sum

Input

N = 5

1, 3, 2, 0, 5

N-1 numbers

Output

Min → 0

Sum → ~~10~~ 11

Calc your ans as you take input one by one.

1, 3, 2, 0, 5

Solution

Read N

Read num

Sum = num

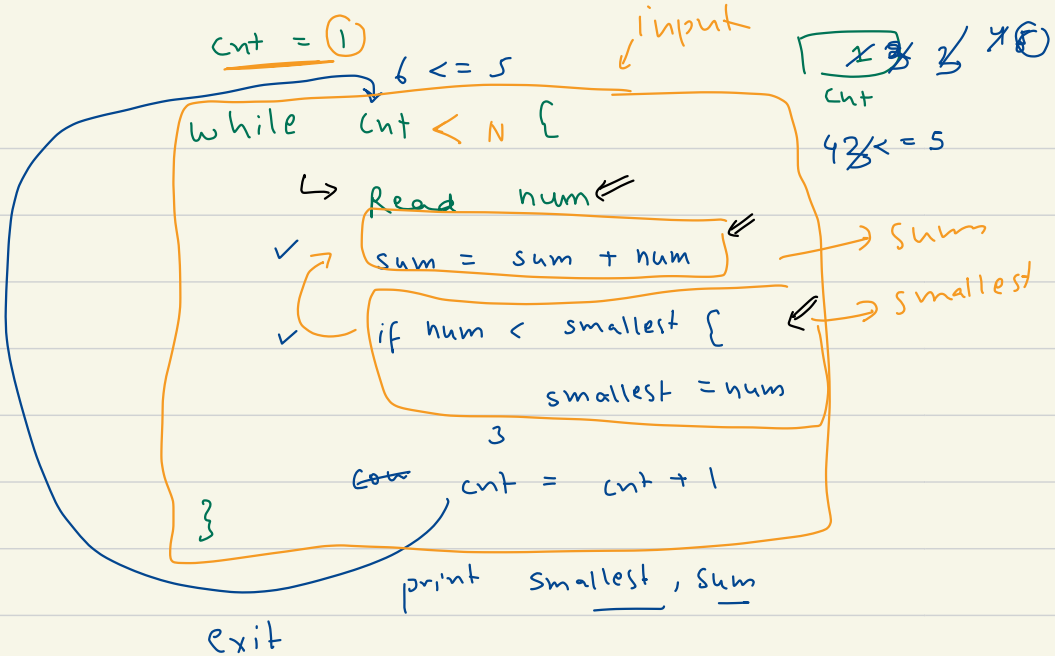
Smallest = num

5 (N)

~~11~~ (Sum)

~~10~~ (Smallest)

~~3~~ (num)



cnt 0 ——— N-1 N=5

0, 1, 2, 3, 4

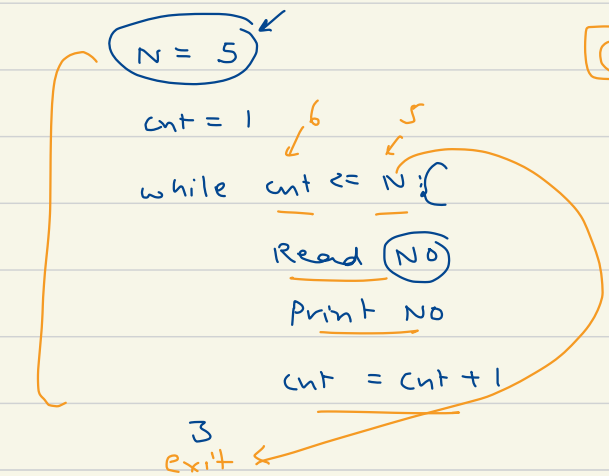
cnt 1 ——— N N=5

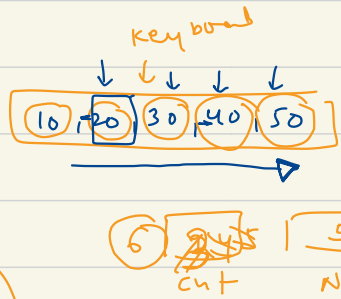
1, 2, 3, 4, 5

100 105

6 times

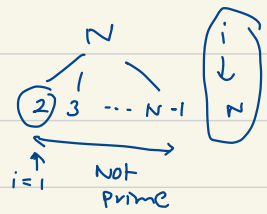
Read 5 No's and print them





10 20 30 40 50

(2) Check if a No N is prime or Not.



Read N

let i = 2 — init

i <= √N

while i <= N-1 — stop

if N % i == 0 {

print "NOT PRIME"

exit

}

i = i + 1 — update

N = 7

i = 2

2 <= 6 — 7 % 2

3 <= 6 — 7 % 3

4 <= 6 — 7 % 4

5 <= 6 — 7 % 5

6 <= 6 — 7 % 6

7 <= 6

⇒ print "PRIME"

⇒ exit

Few Optimisations

≈ N steps

N = 10,000 ≈ 10000 steps

N/2 ≈ 5000 steps

Why → √N ≈ 100 steps

30 → 1, 2, 3, 5, 6, 10, 15, 30

small | Large

≤ √N		> √N	
1	×	30	= 30
2	×	15	= 30
3	×	10	= 30
5	×	6	= 30

a × b = N

√N × √N ≥ N

if a no doesn't have a div. less than ≤ √N it will never have a divisor > √N

36 → 1 × 36 = 36

2 × 18 = 36

3 × 12 = 36

4 × 9 = 36

6 × 6 = 36

atleast 1 div 2 to $\sqrt{N}$

1, 2, 3, 4, 6, 9, 12, 18, 36 =

6 × 6 = 36

4 × 9 = 36

15 → 1, 3, 5, 15

2 to $\sqrt{29}$

29 → 2, 3, 4, 5

$\sqrt{29}$ = 5

Prime

29 % 2

29 % 3

29 % 4

29 % 5

2, 3, 4, 5 ---- 28

43 → 2, 3, 4, 5, 6

$\sqrt{43}$ = 6

Prime

```
Read N
i = 2
while i*i <= N {
    if N%i == 0 {
        print "Not prime"
        exit
    }
    i = i+1
}
print Prime
exit
```

saved rounds of steps

$i <= \sqrt{N}$

$i^2 <= N$

$6 \times 6 <= 43$

$7 \times 7 <= 43$

$\sqrt{N}$ → Binary search

Library

Linear S.

```
for (i=2; i^2 <= N; i++){
    if N%i == 0
        Not prime
        exit
}
```

Pattern Problem

Print a pattern for given N.

input

N = 4

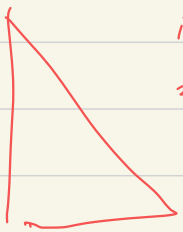
output

R1 1

R2 2 3

R3 4 5 6

R4 7 8 9 10



Observations

1) No of Rows = N

2) No of elements in each Row → Row No

3) Every time the no is inc by 1.

Read N

r = 1, ~~val = 1~~

while r <= N {

cnt = 1

while cnt <= r {

~~print val +" "~~

~~val = val + 1~~

cnt = cnt + 1

}

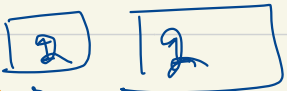
print("\n")

r = r + 1

}

exit

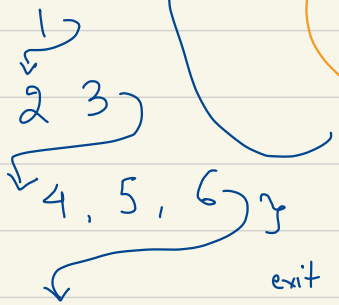
N = 3

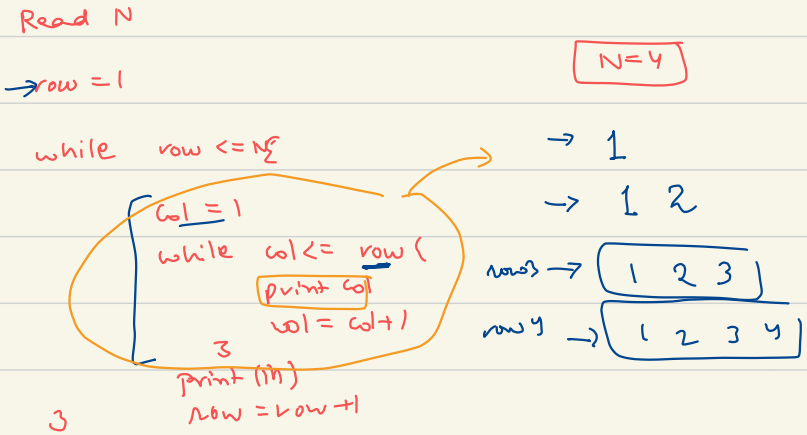


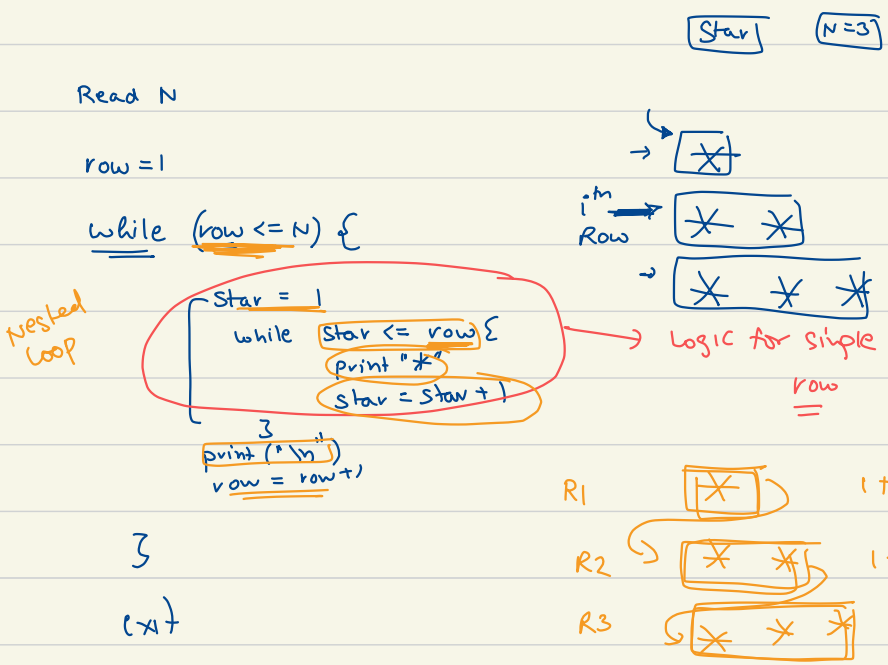
1<=3

print cnt 1

→ Enter




Read N
row = 1
while row <= N{
col = 1
while col <= row {
print col
col = col+1
}
print ("\n")
row = row+1
}
N=4
→ 1
→ 1 2
row3 → 1 2 3
row4 → 1 2 3 4

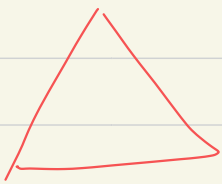

Star
N=3
Read N
row = 1
while (row <= N) {
Star = 1
while Star <= row {
print "*"
Star = Star + 1
}
print ("\n")
row = row+1
}
(x)
Nested Loop
i.e Row
Logic for simple row
R1
R2
R3
1 to 1
1 to 2
1 to 3

Homework

Q1. Print Star Pattern for N.

N = 3

```
  *
 * * *
* * * * *
```



Q2

N = 4

```
1
0 1
1 0 1
0 1 0 1
```

Q3 Pattern

N = 4

```
   1
  2 3 2
 3 4 5 4 3
4 5 6 7 6 5 4
```