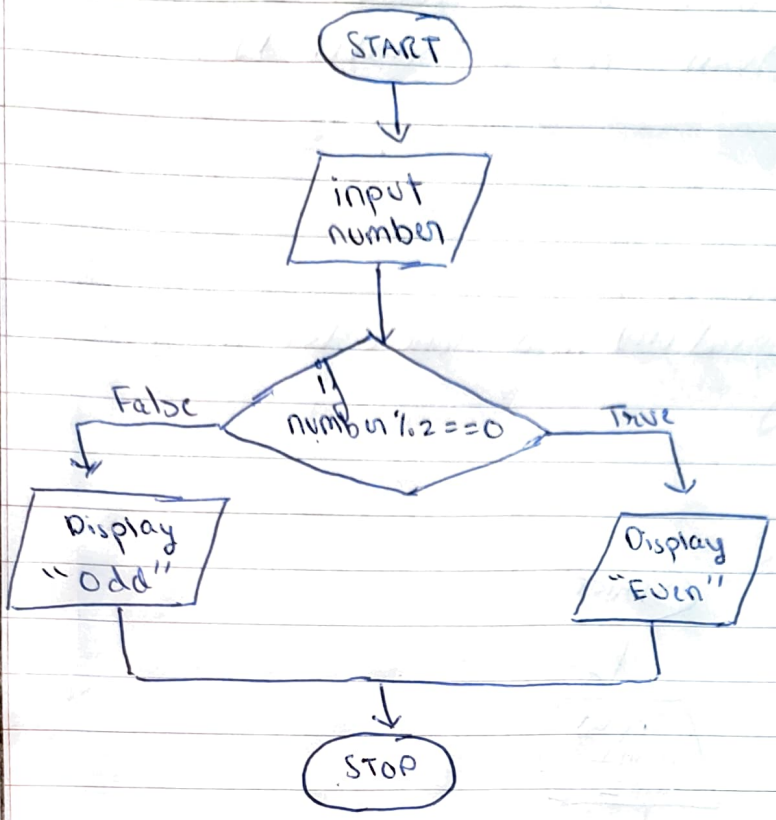
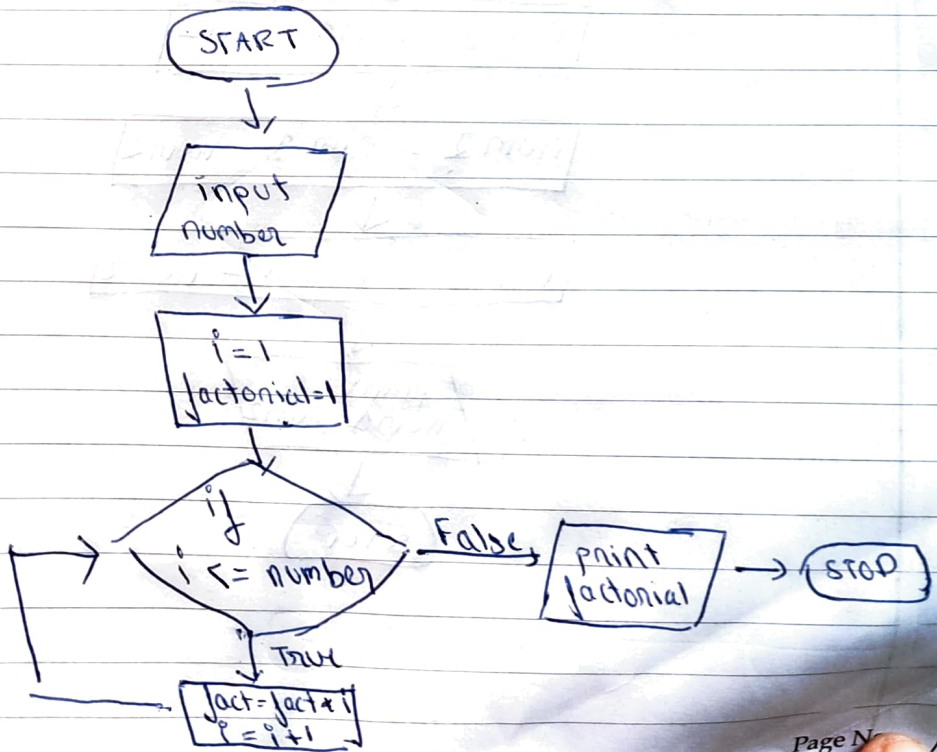
COAC : Assignment 1

1] If number is Even or ODD.



2] Factorial of number.



Q3) Factorial using recursion.

Algorithm:-

factorial(n)

1] Check if n==1, if True return 1.

2] Else, return n * factorial(n-1)

main()

1] Start

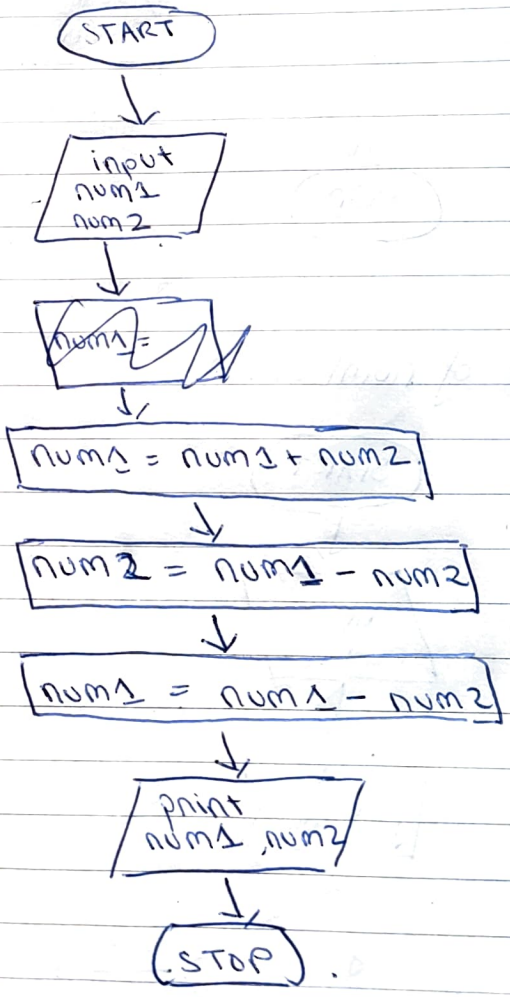
2] Input integer n

3] Call factorial(n) and store value in 'f'

4] Print 'f'

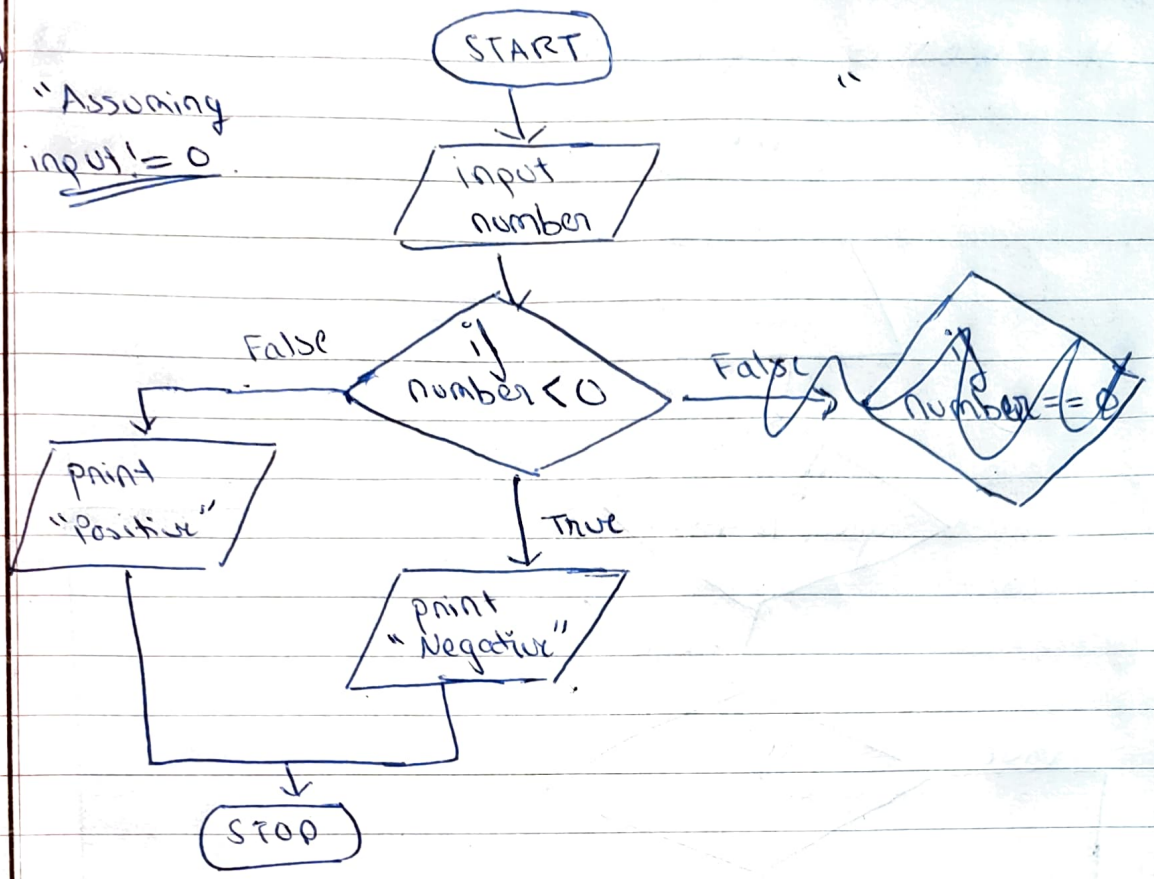
5] Stop.

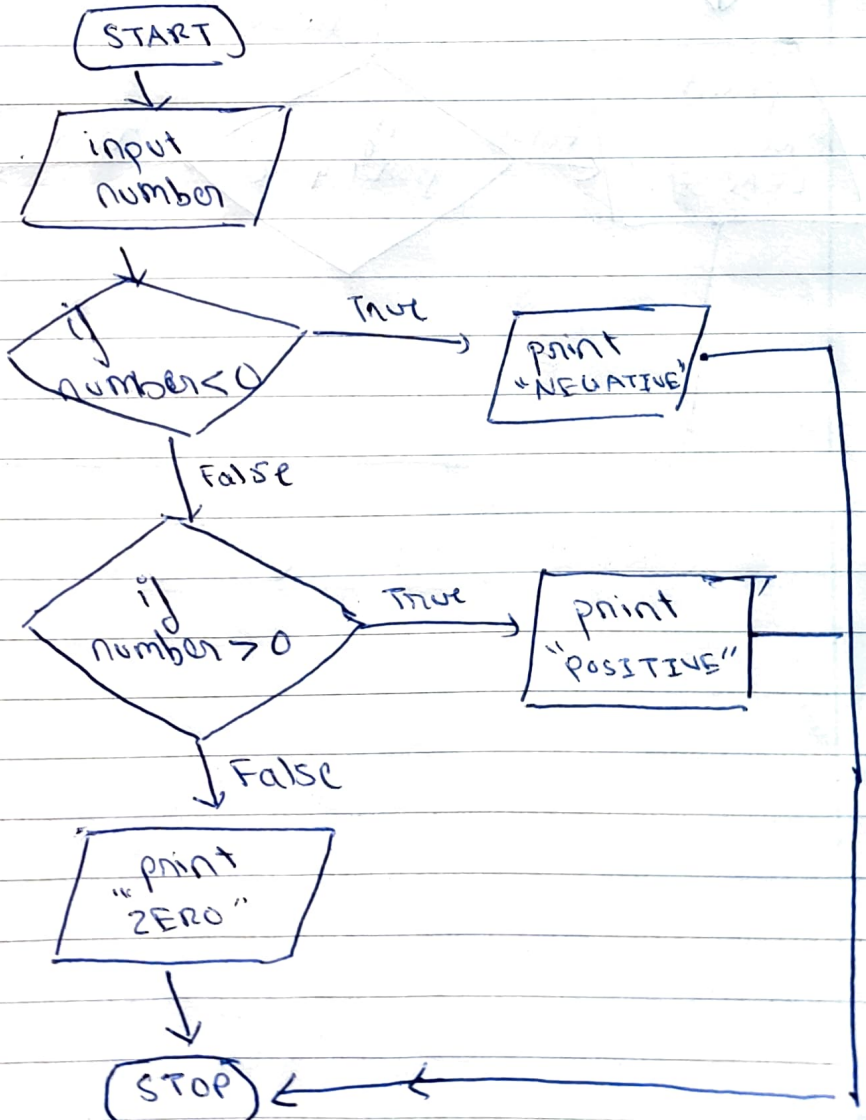
Q4]



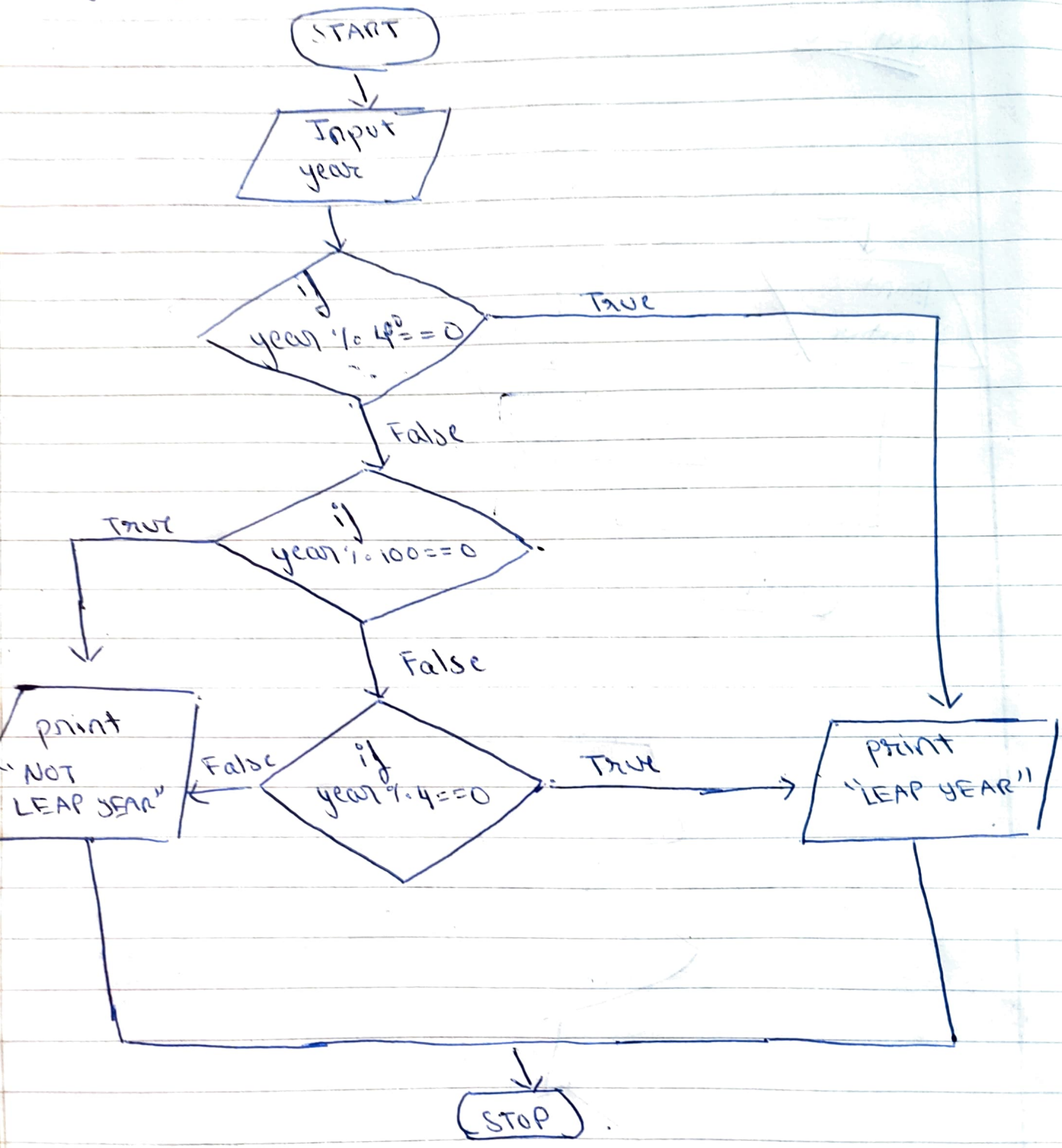
95)

"Assuming input != 0"

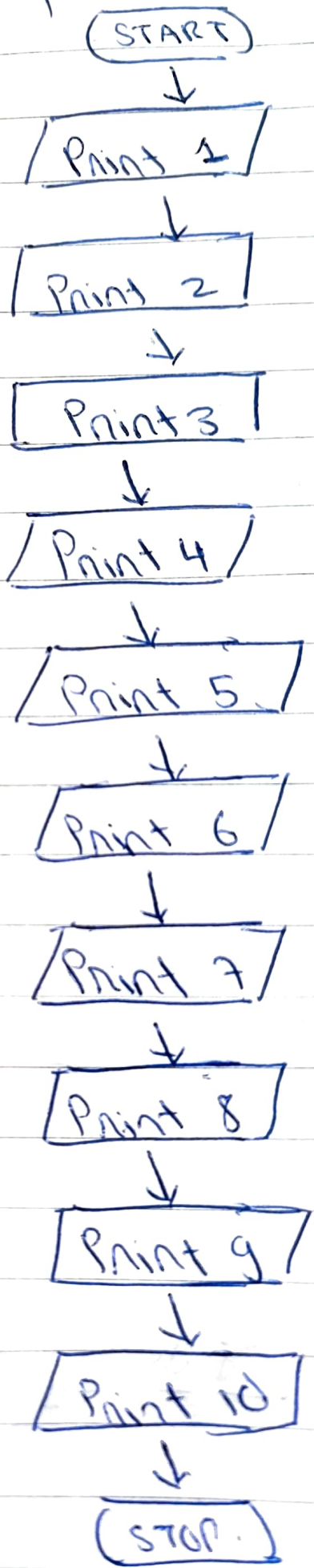




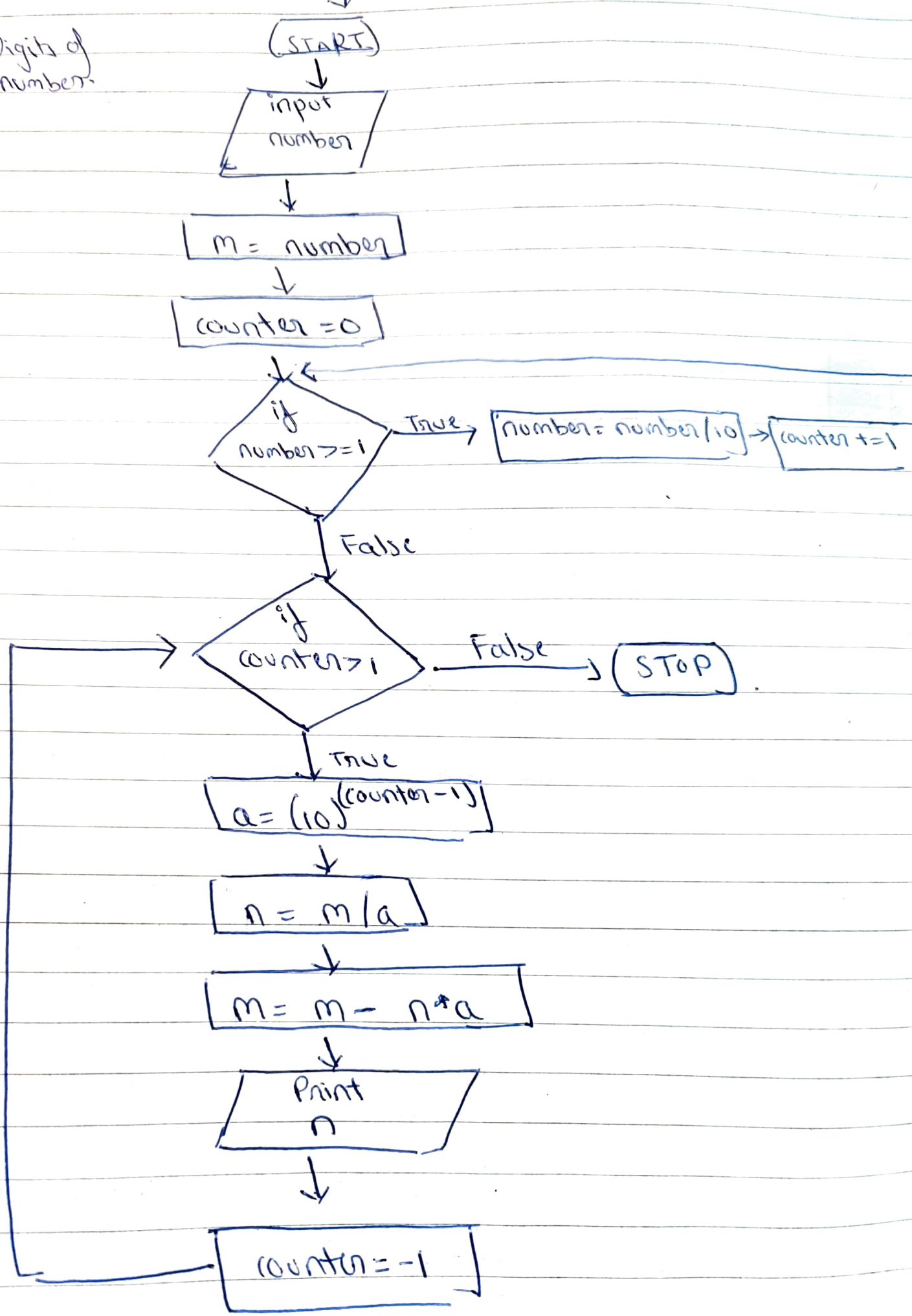
If a year is leap or not.



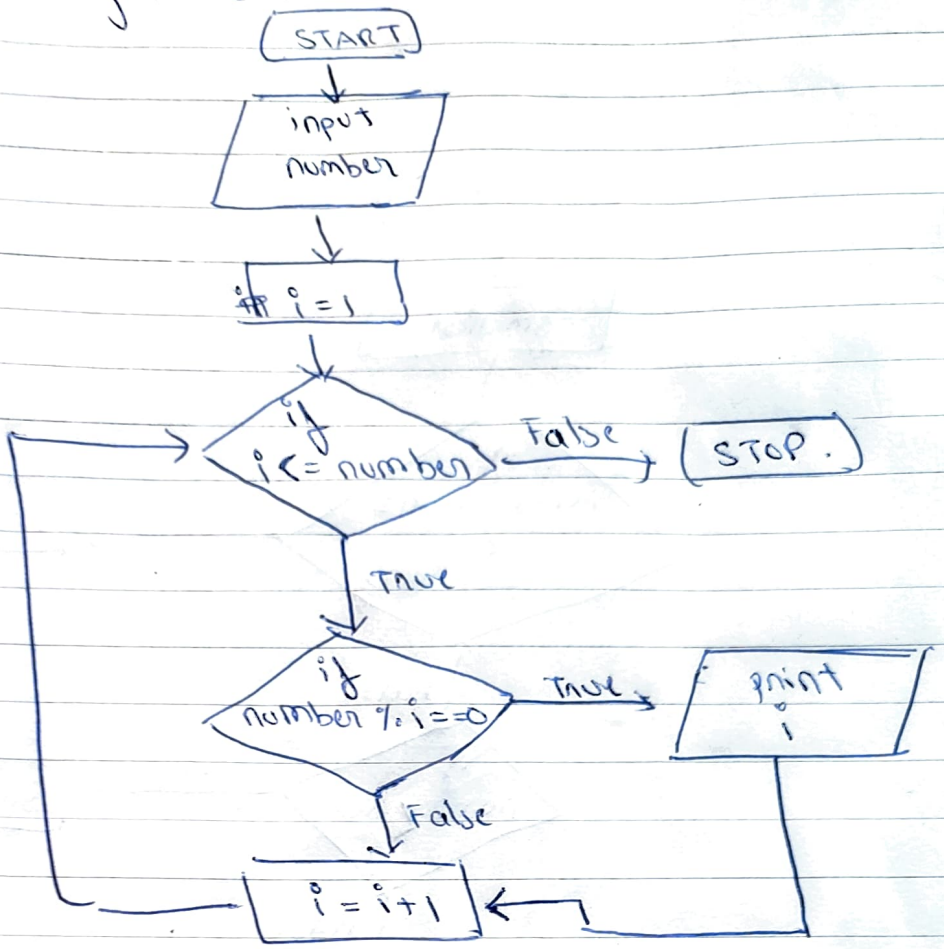
17) Program to prin 1 to 10 withouth loop.



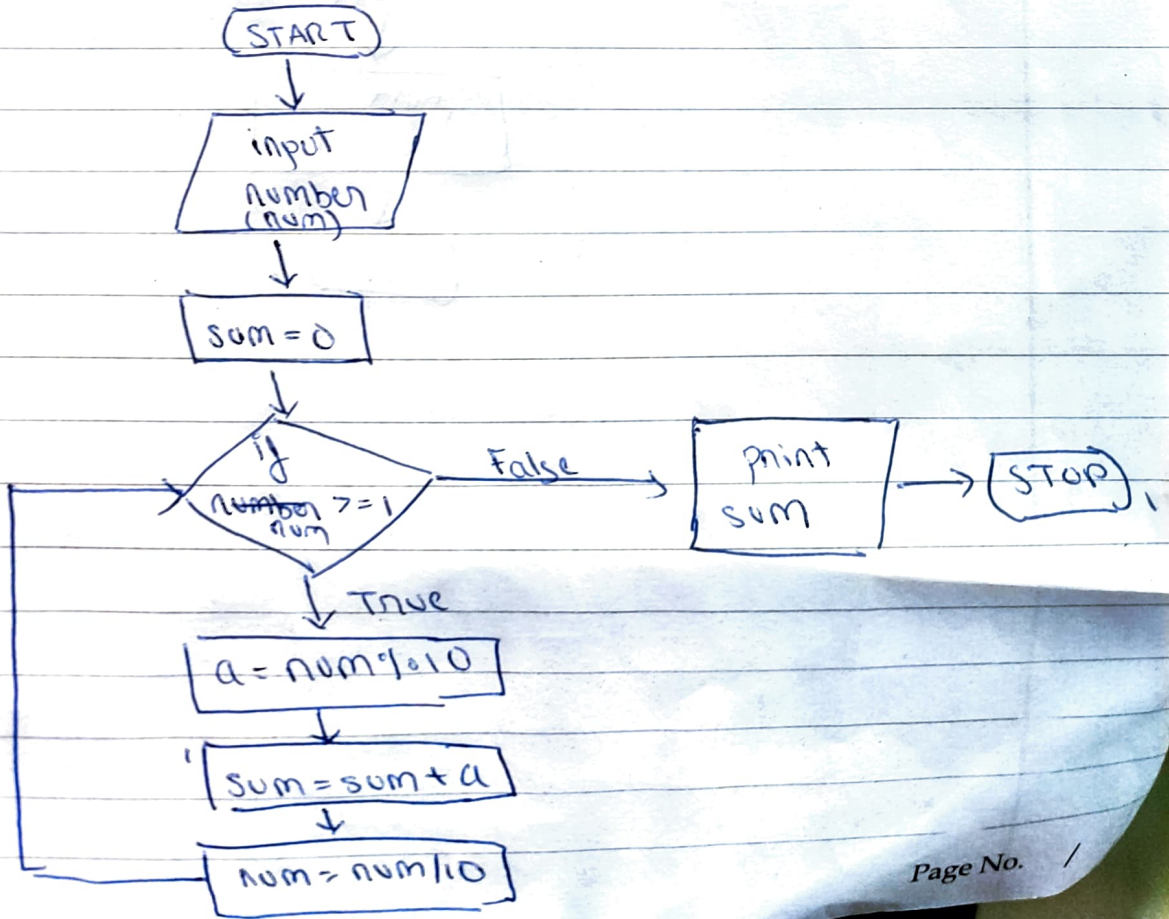
Q8] Digits of number.



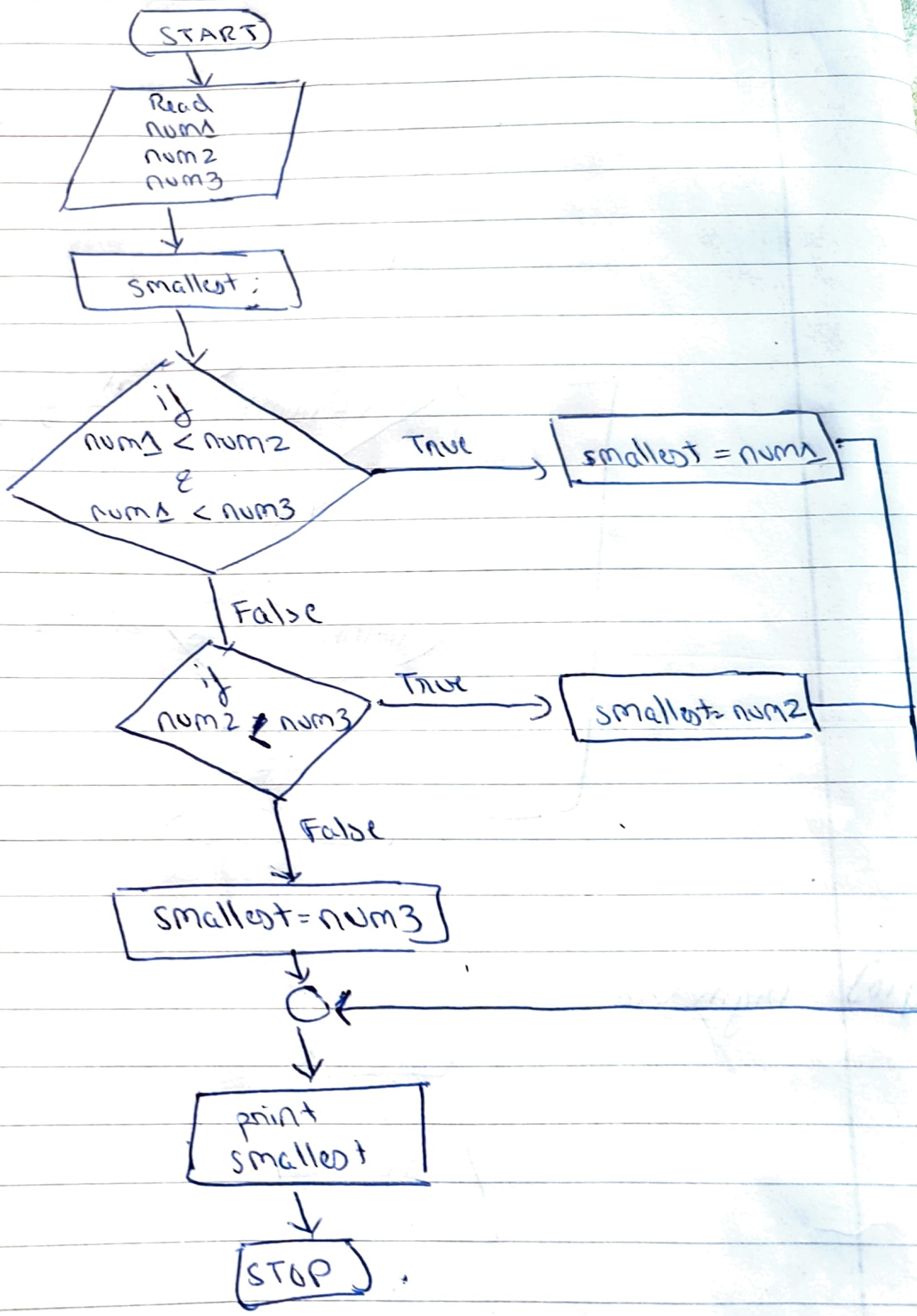
q9) Factors of number.



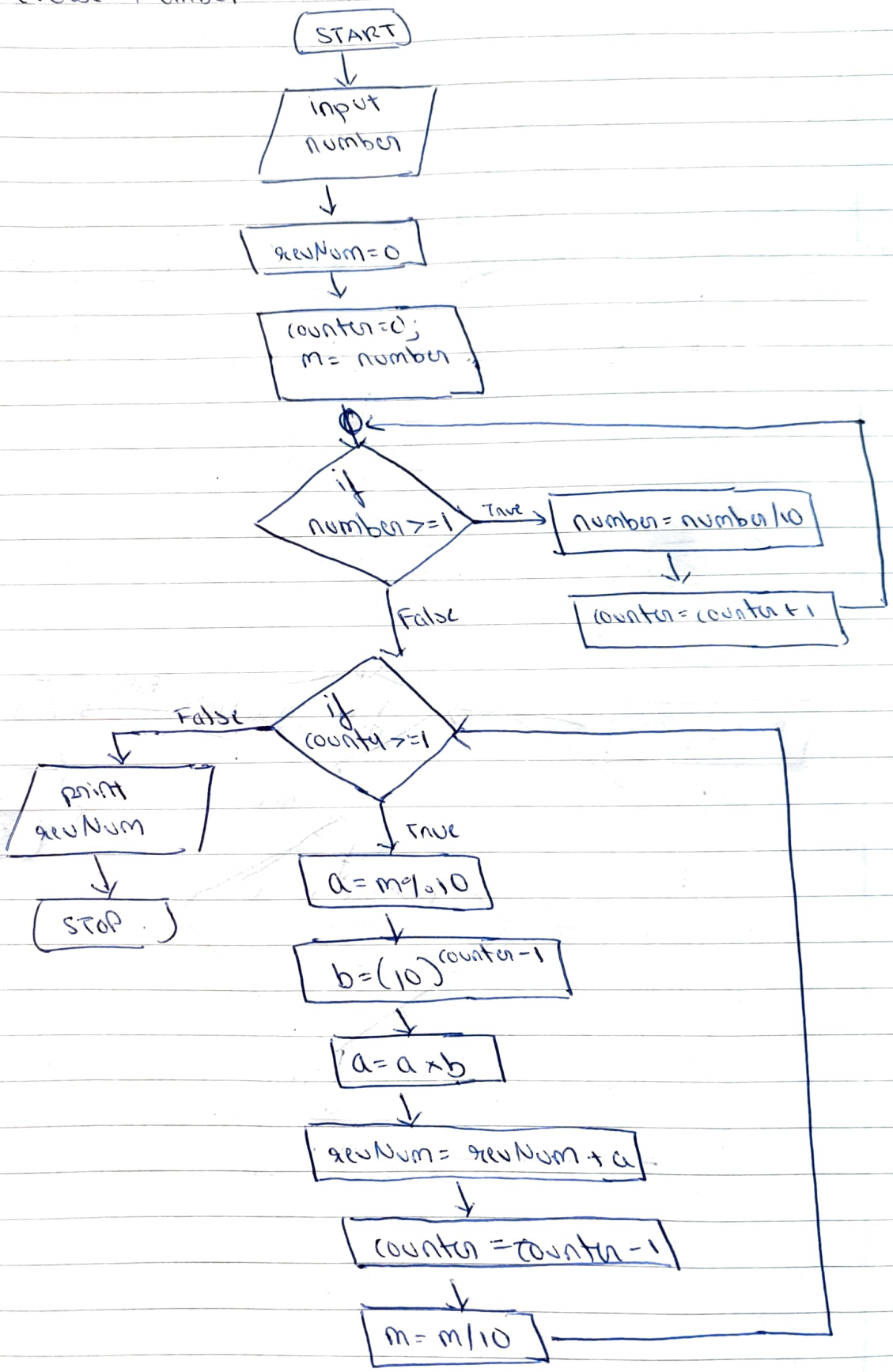
q10) Digits Sum:



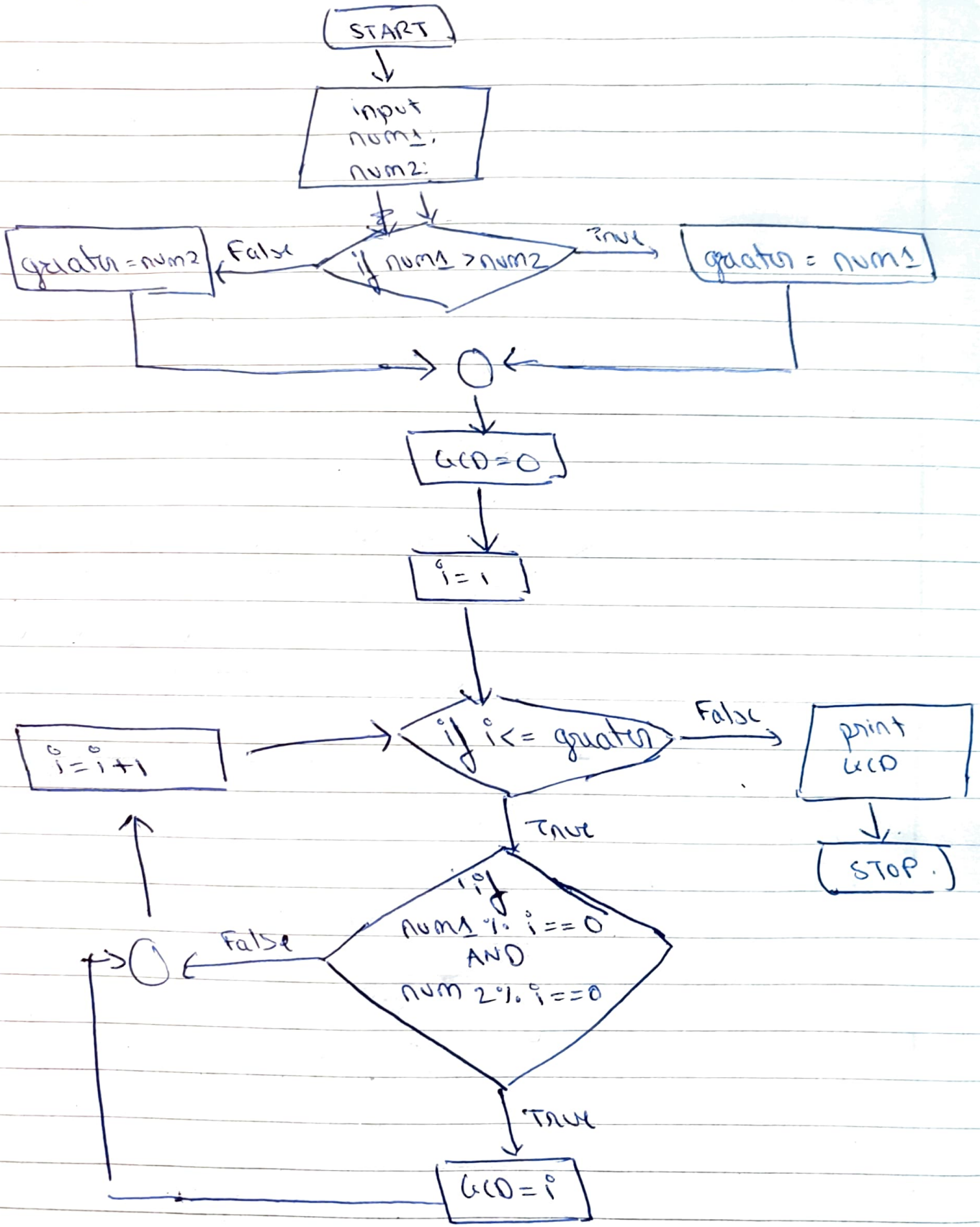
(11) Smallest


START
Read
num1
num2
num3
smallest ;
if
num1 < num2
&
num1 < num3
True
smallest = num1
False
if
num2 < num3
True
smallest = num2
False
smallest = num3
print
smallest
STOP

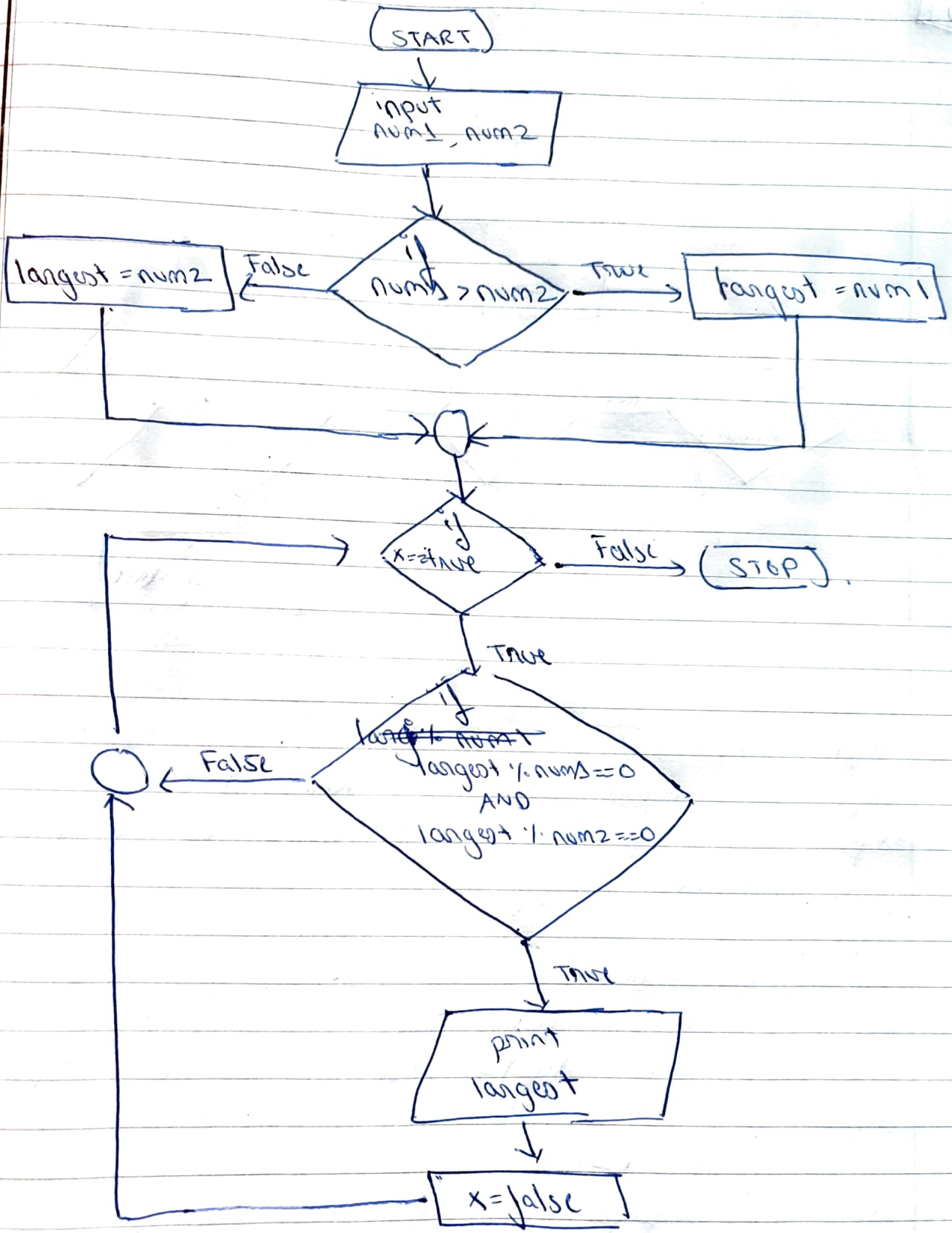
Q13] Reverse Number



9.4] GCD


START
input num1; num2;
if num1 > num2
True
greater = num1
False
greater = num2
GCD=0
i=1
if i<= greater
False
print GCD
STOP
True
if num1 % i == 0 AND num 2% i==0
False
True
GCD=i
i=i+1

Q15) LCM


START
input num1, num2
num1 > num2
True
False
largest = num1
largest = num2
x = true
False
STOP
True
largest % num1 == 0
AND
largest % num2 == 0
False
True
print largest
x = false

Step 1: START.

Step 2: Read num1 and num2 [Numbers].

Step 3: Store a copy of num1 and num2 in num1_copy and num2_copy.

Step 4: i = 2

Step 5: if i < num1 goto step 6, else goto step

Step 6: if num1 is prime add num1 to list num1fact; then goto step 13 else go to next step.

Step 7: if 'i' is prime, go to step 8, else go to step 12.

Step 8: if num1_copy % i == 0 go to step 9, else goto step

Step 9: add 'i' to num1fact.

Step 10: num1_copy = num1_copy / i

Step 11: Go to step 5

Step 12: i = i+1, Go to step 5.

Step 13: i = 2

Step 14: Repeat step 5 to step 12 for (num2) and variables num2_copy & num2fact.

Step 15: Print num1fact and num2fact

Step 16: Create variables greater, lesser, num1_state and LCMPrimefactor list

Step 17: if size of num1fact > num than size of num2fact then goto step 18 or go to step 19

Step 18: greater = size of num1fact
lesser = size of num2fact
num1_state = true, goto step 20

Step 19: greater = size of num2fact
lesser = size of num1fact
num1_state = false.

Step 20: j=0

Step 21: if j < greater, goto step 22, else goto step 29

Step 22: if j >= lesser, goto step 23, else goto step 26.

Step 23: num1_state == True, goto step 24, else goto step 25

Step 24: add the number at position 'j' of num1fact to LCMPrimefactor list. Go to step 28

Step 25: add number at position j of num2fact to LCM Prime factor list. Goto step 28

Step 26: if num1fact[j] == num2fact[j], then add num1fact[j] or num2fact[j] to LCMPrimefactor list. Go to step 28.

Step 27: if num1fact[j] ≠ num2fact[j], add both num1fact[j] and num2fact[j] to LCM Primefactor list.

Step 28: j=j+1, goto step 21

Step 29: Print LCM Prime factor list.

Step 30: LCM=1 ; k=0

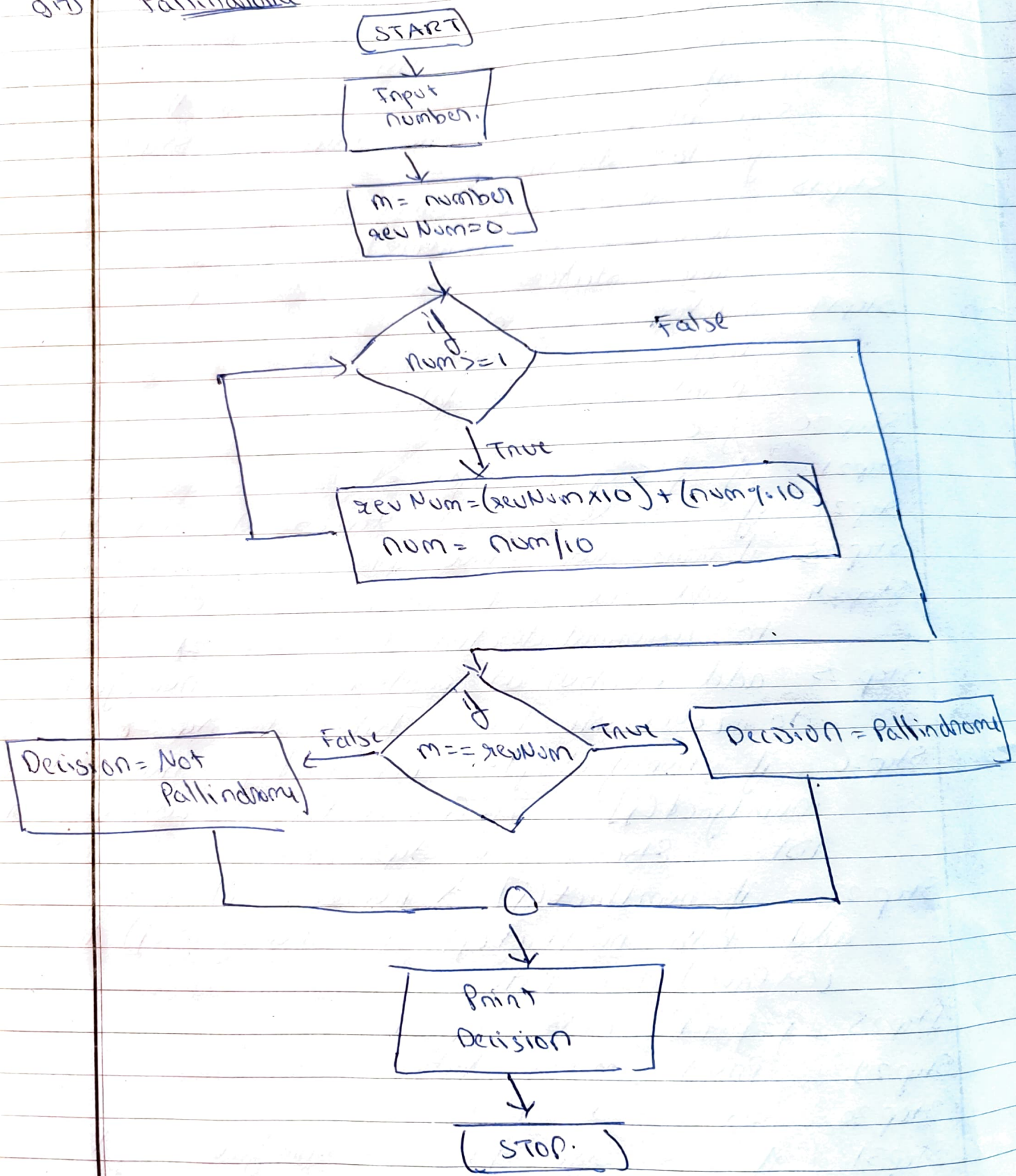
Step 31: if k < size of list LCM Primefactor, goto step 32, else goto step 34.

Step 32: LCM = LCM * [number at position 'k' of LCMPrimefactor list].

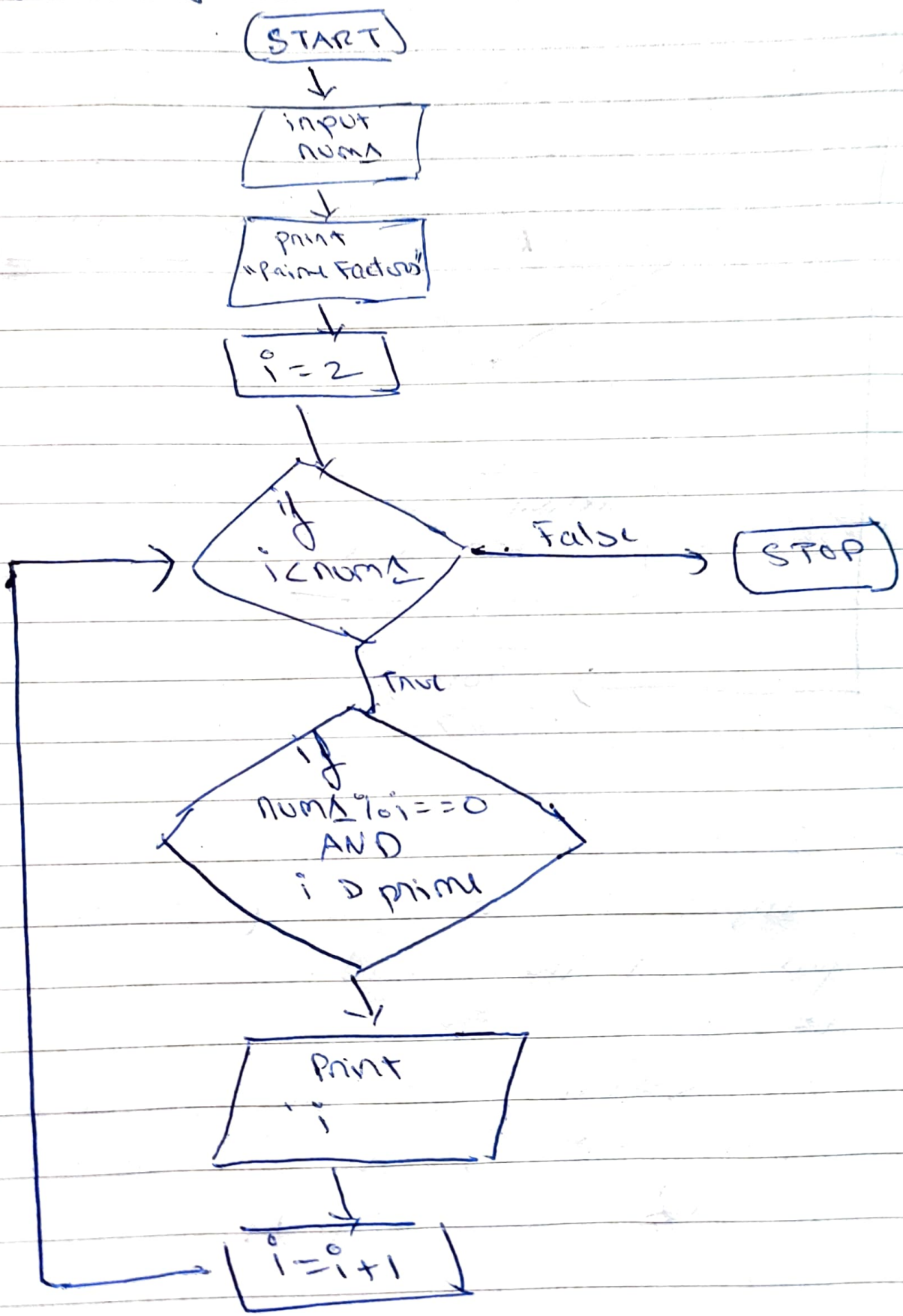
Step 33: k=k+1

Step 34: Print LCM.

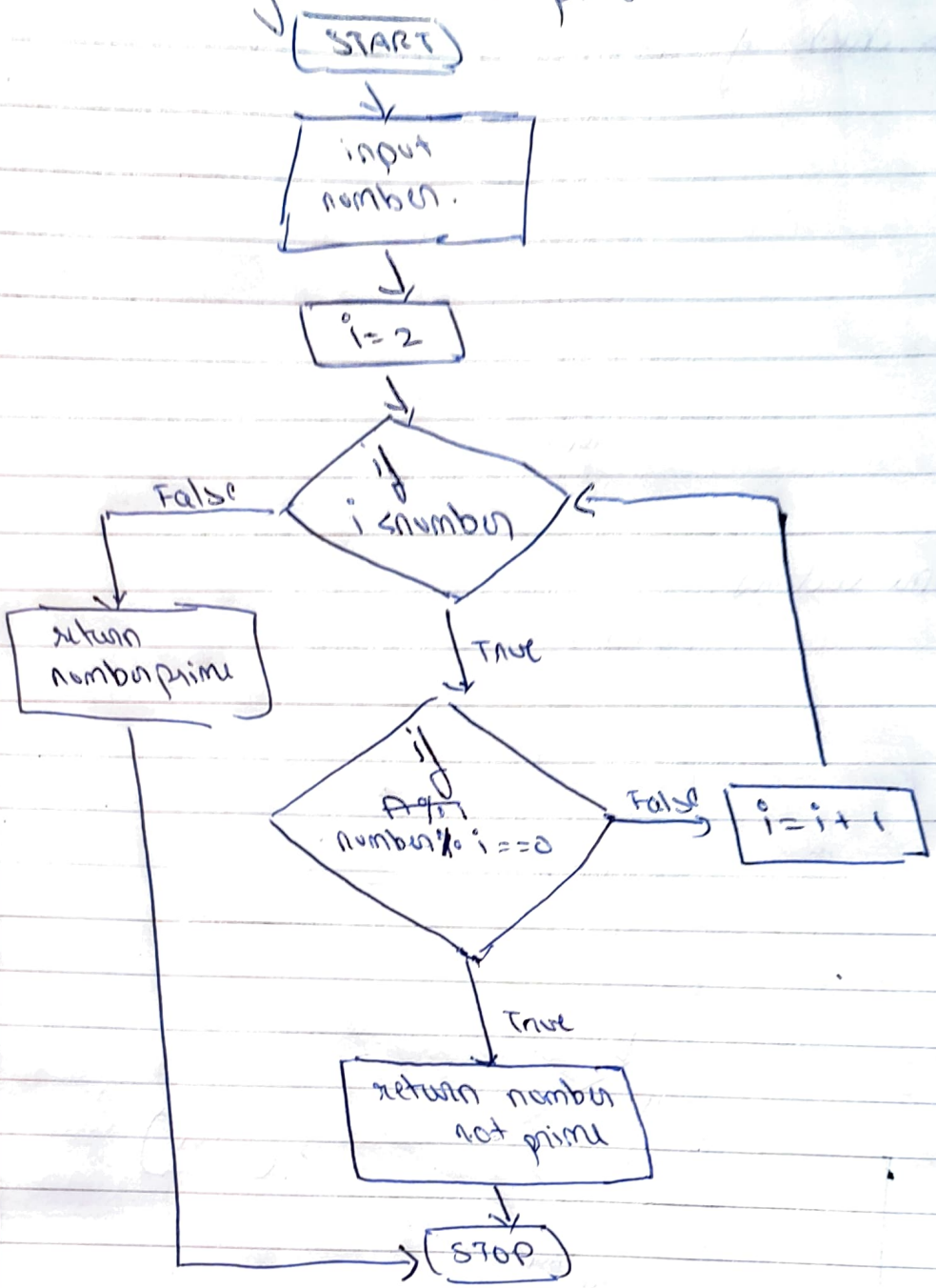
Q17) Pallindrome



Prime Factors

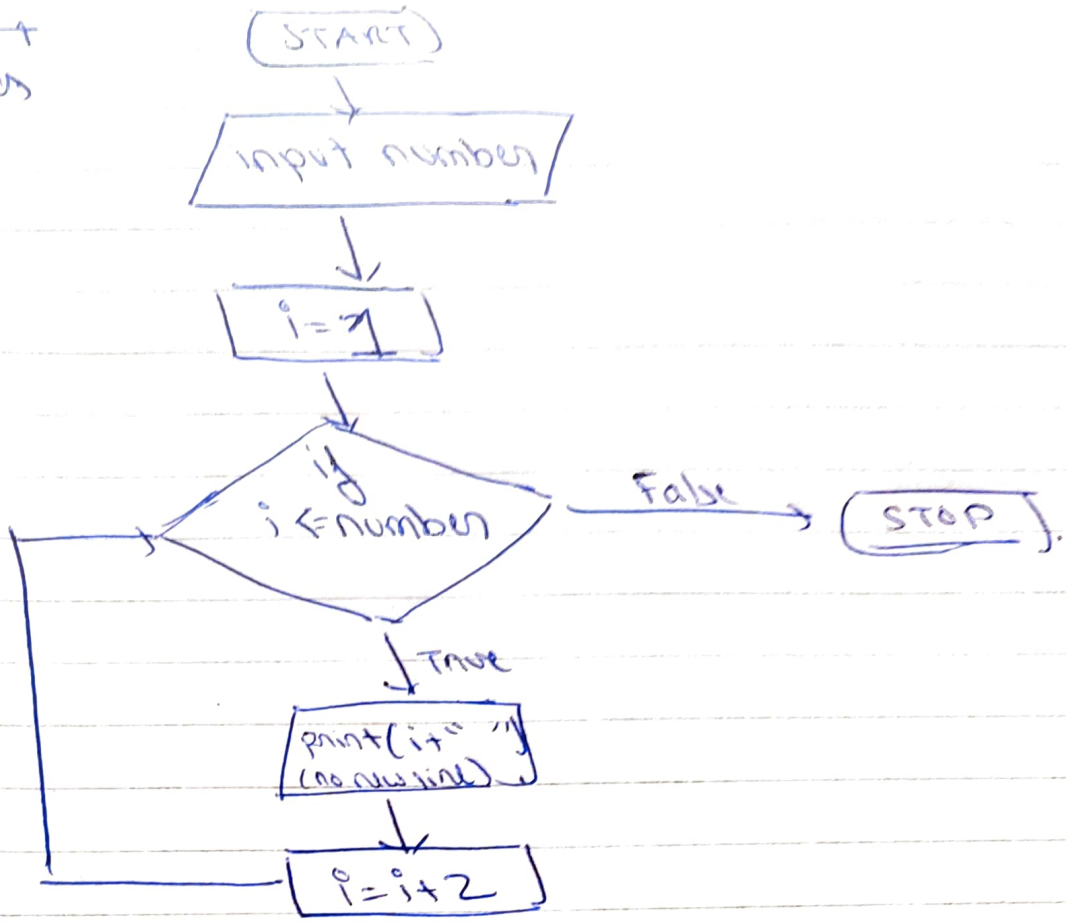


To chuck if number is prime.



Q.1] Step 1: Start
Step 2: Input number & num [till the ~~pes~~ number you want the series].
Step 3. i = 2.
Step 4 if i <= num, goto step 5, else goto step. 7
Step 5 print (i + " ") No new line.
Step 6 i = i + 2, goto step 4
Step 7 STOP

Q.1) Step 1
ODD series



Q.2) Addition w/o Arithmetic [Half Adder]