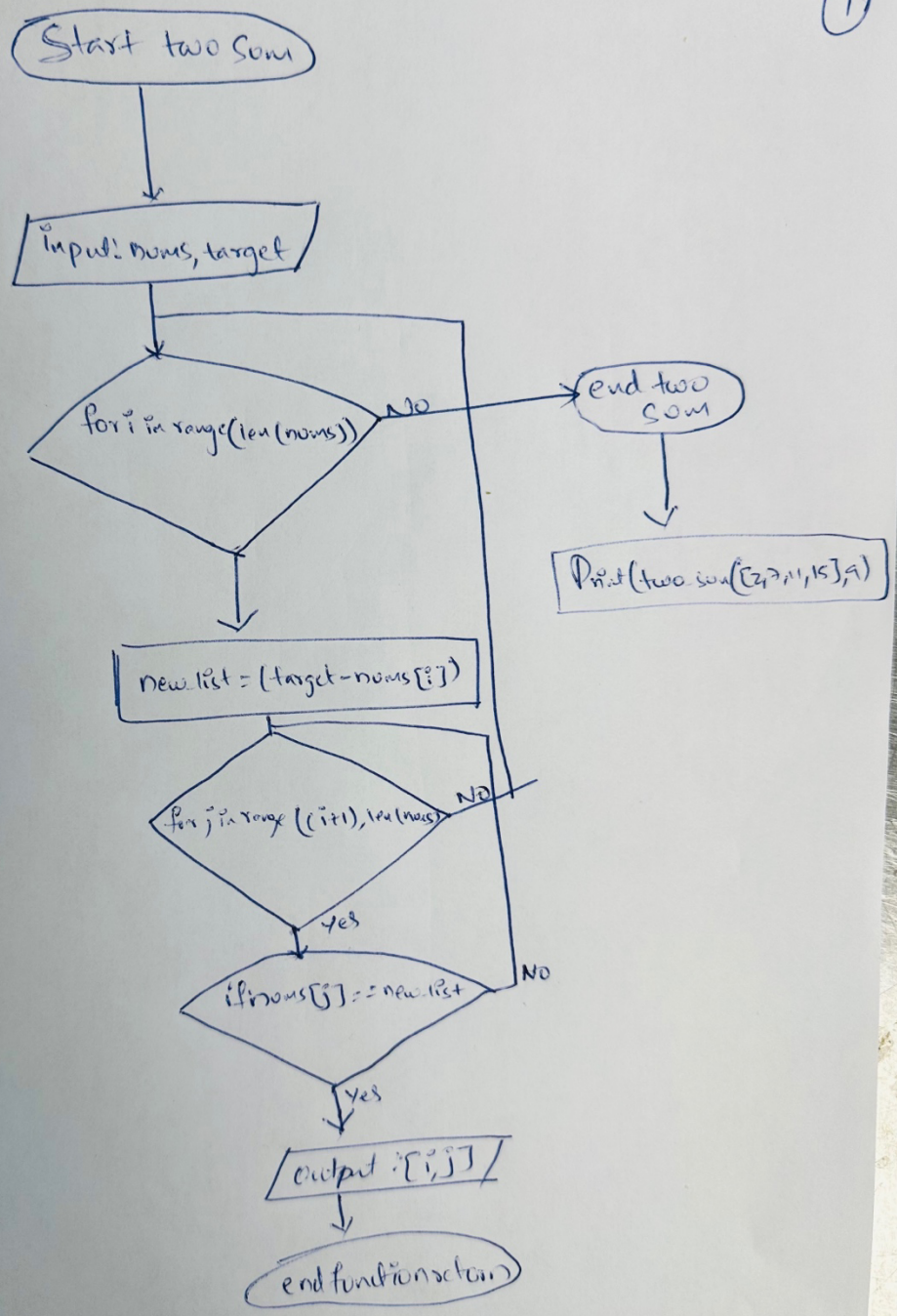
# Algorithms and structures programming

Rakesh Kasha
19695

## Python Program:

```
def two_sum(nums, target):

    for i in range(len(nums)):

        new_list = target - nums[i]


        for j in range(i + 1, len(nums)):

            if nums[j] == new_list:
                return [i, j]


print(two_sum([2,7,11,15], 9))
print(two_sum([2,3,4],6))
print(two_sum([3,3],6))
```

Flowchart:


Start two Sum
input: nums, target
for i in range(len(nums))
NO
end two sum
Print(two_sum([2,7,11,15],9)
new_list = (target-nums[i])
for j in range((i+1), len(nums))
NO
yes
if nums[j] == new_list
NO
yes
output: [i,j]
end function sum

Trace Table:

| Step | Nums | Target | I | J | New_list | Nums[j]==new_list | screen |
|---|---|---|---|---|---|---|---|
| 1 | [2,7,11,15] | 9 | | | | | |
| 2 | | | 0 | | | | |
| 3 | | | | | 9-2=7 | | |
| 4 | | | | 1 | | | |
| 5 | | | | | | True | |
| 6 | | | | | | | [0,1] |

Test Case:

Welcome | loop.py

Users > rakesh_kasha > Desktop > loop.py > two_sum

```python
def two_sum(nums, target):

    for i in range(len(nums)):

        new_list = target - nums[i]


        for j in range(i + 1, len(nums)):

            if nums[j] == new_list:
                return [i, j]


print(two_sum([2,7,11,15], 9))
# print(two_sum([2,3,4],6))
# print(two_sum([3,3],6))
```