Program:

```
def productDigits(n):
    a=n
    temp=[]
    list1=[]
    list2=[]
    rem=0
    while a!=0:
        rem=a%10
        temp.append(rem)
        a=a//10
    for i in range(len(temp)):
        if(i+1)%2==0:
            list1.append(temp[i])
        else:
            list2.append(temp[i])
    pro=1
    sum=0
    for i in list1:
        sum+=i
    for i in list2:
        pro*=i
```

```
if pro%sum==0:
    return True
else:
    return False
```

	Test	Expected	Got	
✔	print(productDigits(1256))	True	True	✔
✔	print(productDigits(1595))	False	False	✔

Ex. No. : 9.3 **Date: 01.06.24**

Register No.: 231901035 **Name: Nitheesh K K**

Abundant Number

An abundant number is a number for which the sum of its proper divisors is greater than the number itself. Proper divisors of the number are those that are strictly lesser than the number.

Input Format:
Take input an integer from stdin
Output Format:
Return Yes if given number is Abundant. Otherwise, print No
Example input:
12
Output:
Yes
Explanation
The proper divisors of 12 are: 1, 2, 3, 4, 6, whose sum is 1 + 2 + 3 + 4 + 6 = 16. Since sum of proper divisors is greater than the given number, 12 is an abundant number.
Example input:
13
Output:
No
Explanation
The proper divisors of 13 is: 1, whose sum is 1. Since sum of proper divisors is not greater than the given number, 13 is not an abundant number.

For example:

Test	Result
print(abundant(12))	Yes
print(abundant(13))	No

Program:

```
def abundant(number):

    d_s=sum([divisor for divisor in range(1,number) if number % divisor == 0])

    if d_s>number:
```

```
    return"Yes"
else:
    return "No"
```

	Test	Expected	Got	
✔	`print(abundant(12))`	Yes	Yes	✔
✔	`print(abundant(13))`	No	No	✔

Ex. No. : 9.4 **Date: 01.06.24**

Register No.: 231901035 **Name Nitheesh K K**

Ugly number

A number is considered to be ugly if its only prime factors are 2, 3 or 5.
[1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, …] is the sequence of ugly numbers.
Task:
complete the function which takes a number n as input and checks if it's an ugly number. return ugly if it is ugly, else return not ugly
Hint:
An ugly number U can be expressed as: U = 2^a * 3^b * 5^c, where a, b and c are nonnegative integers.
For example:

Test	Result
print(checkUgly(6))	ugly
print(checkUgly(21))	not ugly

Program:

```
def checkUgly(n):
    if n <= 0:
        return "not ugly”
    while n % 2 == 0:
        n //= 2
    while n % 3 == 0:
        n //= 3
```

```
while n % 5 == 0:
    n //=5
return "ugly" if n == 1 else "not ugly"
```

	Test	Expected	Got	
✔	print(checkUgly(6))	ugly	ugly	✔
✔	print(checkUgly(21))	not ugly	not ugly	✔

Ex. No. : 9.5 **Date: 01.06.24**

Register No.: 231901035 **Name Nitheesh K K**

Automorphic number or not

An automorphic number is a number whose square ends with the number itself. For example, 5 is an automorphic number because 5*5 =25. The last digit is 5 which same as the given number.

If the number is not valid, it should display “Invalid input”.
If it is an automorphic number display “Automorphic” else display “Not Automorphic”.

Input Format:
Take a Integer from Stdin
Output Format:
Print Automorphic if given number is Automorphic number, otherwise Not Automorphic
Example input: 5 Output: Automorphic Example input: 25 Output: Automorphic Example input: 7 Output: Not Automorphic

For example:

Test	Result
print(automorphic(5))	Automorphic

Program:

```
def automorphic(n):

    if(n<0):

        return "Invalid input"

    square = n  * n

    n_s=str(n)

    s_s=str(square)

    if s_s.endswith(n_s):
```

```
    return "Automorphic"
else:
    return "Not Automorphic"
```

	Test	Expected	Got	
✔	`print(automorphic(5))`	`Automorphic`	`Automorphic`	✔
✔	`print(automorphic(7))`	`Not Automorphic`	`Not Automorphic`	✔

10 - Searching & Sorting

Ex. No. : 10.1 **Date: 01.06.24**

Register No.: 231901035 **Name Nitheesh K K**

Bubble Sort

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order. You read an list of numbers. You need to arrange the elements in ascending order and print the result. The sorting should be done using bubble sort.

Input Format: The first line reads the number of elements in the array. The second line reads the array elements one by one.

Output Format: The output should be a sorted list.

For example:

Input	Result
6 3 4 8 7 1 2	1 2 3 4 7 8
5 4 5 2 3 1	1 2 3 4 5

Program:

```
n=int(input())
k=[int(x) for x in input().split()]
k.sort()
for i in k:
    print(i,end=' ')
```