**Day-01**

**19-03-2025**

**================**

**While loop:**

**-------------**

Q-1: When we need for loop?

---------------------------------

Ans:

----

When we can estimate the number of iterations, we need to use "for loop".

Ex: 7 is prime or not

7 ==> 2 to 6

perfect number

6 ==> perfect or not

6 ==> 1,2,3,6

1 + 2 + 3 ==> 6

Q-2: When we need while loop?

-----------------------------------

Ans:

----

When we cannot estimate the number of iterations, we need to use "while loop".

Ex: 987654321 ==> 9-digit

12034056098710101 ==> how many number of digits

While Vs for loop:

--------------------

1) when we can estimate the number of iterations we can use "for loop"

where as when we cannot estimate the number of iterations we can use "while loop".

2) for always allow to define with range()

but while is not required to define with range()

3) both are loop entry control statements.

for i in range(1,11):

print("hi")

Syntax of while loop:

------------------------

three things:

1) initialization --> describe that which value the loop can get start

2) condition --> until which value the loop can continue to execute

3) update --> describe the difference from current value to the next value.

Syntax:

initialization

while condition:

statement-1

statement-2

update

**# Write a program to print the numbers from 1 to 10**

start = 1

while start <= 10:

print(start,end = "\t")

start = start + 1

**Day-02**

**20-03-2025**

**===================**

**# write a python program to count the number of digits of the given number.**

**# Solution-1: using str()**

number = int(input("Enter a value:"))

dummy = str(number) # number can convert into the string

lengthOfNumber = len(dummy)

print("The number of digits of a number",number,"is = ",lengthOfNumber)

len():

-----

-> can give you the length of the reference datatype or string

-> to count the number of characters in a string, we can use "len()".

syntax:

len(str-data-variable)

note: len() is not possible with numbers and boolean

**# solution-2: using while loop**

number = int(input("Enter a value:")) # 9876

n =number # initialization 9876

count = 0

while n != 0:

n = n // 10 # 987 98 9 0

count = count + 1 # 1 2 3 4

print("The number of digits in a number",number,"is = ",count)

**# Write a python program to reverse the given number.**

# n = number // 10

# number % 10

# reverse \* 10 + remainder

number = int(input("Enter a value:")) # 9876

reverse = 0

n = number # 9876

while n != 0:

remainder = n % 10 # 6 7 8 9

reverse = reverse \* 10 + remainder # 6 67 678 6789

n = n // 10 # 987 98 9 0

print("The number after the reverse = ",reverse)

**# Write a python program to find the sum of individual digits of a given number**

"""

n = 9876

individual digits n % 10

9 + 8 + 7 + 6 => 30

"""

number = int(input("Enter a value:")) # 9876

sumOfDigits = 0

n = number # 9876

while n != 0:

individualDigits = n % 10 # 6 7 8 9

sumOfDigits = sumOfDigits + individualDigits # 6 13 21 30

n = n // 10 # 987 98 9 0

print("The sum of all digits of number",number,"is = ",sumOfDigits)

**Assignment:**

**========**

Write a python program to check whether the given number is palindrome number or not.

Hint:

-----

reverse of the number == number:

palindrome number

otherwise:

not a palindrome number

Ex: 1221

rev = 1221

1221 == 1221 ==> Palindrome number

9876

rev = 6789

6789 != 9876

not a palindrome number

**Day-03**

**21-03-2025**

**=======================**

**# Write a python program to accept a number as an input**

**# And check whether the given number is Armstrong number or not.**

151

==> three digit number

The sum of cubes (3 power) of individual digits of a number is equals to given number then it is called as "Armstrong number" (For only three digit number)

151: 1^3 + 5^3 + 1^3

1 + 125 + 1 ==> 127

-> Not an Armstrong number

153: 1^3 + 5^3 + 3^3

1 + 125 + 27

153

Armstrong number

=======================

Assume, if we have a four-digit number

then:

the sum of 4th power of individual digits of a number is equals to given number then, it is called as "Armstrong number"

Ex: abcd

a^4 + b^4 + c^4 + d^4 == abcd

Common definition:

------------------

The sum of nth powers of individual digits is equals to given number, then it is called as an "Armstrong number".

here: n is the length of the number

To do this we need to perform the below steps:

-----------------------------------------------

1) find the length of the given number.

2) Calculate the individual digits from the number using '%' operator

3) find powers of each individual digit of a number

4) find the sum of powers of each digit.

5) verify the sum == given number or not.

number = int(input("Enter a value:"))

length = len(str(number))

n = number # initialization of while loop 153

sumOfPowers = 0

while n > 0:

digits = n % 10 # 3 5 1

powers = digits \*\* length # 27 125 1

sumOfPowers = sumOfPowers + powers # 27 152 153

n = n // 10 # 15 1 0

if sumOfPowers == number:

print("The given number",number,"is an Armstrong number.")

else:

print("The given number",number,"is not an Armstrong number.")

**Write a python program to accept a range of numbers.**

**And print all palindrome numbers from the given range of numbers.**

121

rev = 121

rev == number ==> palindrome number

r1 = int(input("Enter the start value for the range:"))

r2 = int(input("Enter the last value for the range:"))

for n in range(r1,r2+1): # 100 to 1000

m = n # 100

reverse = 0

while m != 0:

digits = m % 10

reverse = reverse \* 10 + digits

m = m // 10

if reverse == n:

print(n,end = "\t")

**Write a python program to accept range of numbers as an input.**

**And print all Armstrong numbers from the given range.**

r1 = int(input("Enter the start value for the range:"))

r2 = int(input("Enter the stop value for the range:"))

for i in range(r1,r2+1):# 1000 to 10000

length = len(str(i))

sumOfPowers = 0

n = i

while n != 0:

digits = n % 10

powers = digits \*\* length

sumOfPowers += powers

n = n // 10

if sumOfPowers == i:

print(i,end = "\t")

Assignment:

-----------

Write a python program to print all 6-digit Armstrong numbers.

(range ==> 100000 to 999999)