Loops

Loops are used to repeat a block of code until a certain condition is met.

loops helps you do repeat the same statement.

Save time, reduce errors & make the code more readable.

For loop

For loop is used for iterating over a sequence (either list, tuple, set, string or range.

For loop in sequence:

eg:
```
name = "theertha"
for i in name:
    print(i)
```
o/p
```
t
h
e
e
r
t
h
a
```

eg:
```
fruits = ['a', 'b', 'c']
for i in fruits:
    print(i)
```

Syntax →
```
for i in sequence:
    print statements
```

i → iterative variable.

enumerate() → fn which gives (index, value) pairs

eg:
```
name = "mannu"
for i in enumerate(name):
    print(i)
```
→
```
(0, 'm')
(1, 'a')
(2, 'n')
(3, 'n')
(4, 'u')
```

break

with the break statement we can stop the loop before it has looped through all the items.

```
fruits = ["apple", "banana", "cherry"]

for i in fruits:
    print(i)
    if i == "banana":
        break
```

o/p → apple
banana

1. i = apple → print(i) → apple.
   if (i == banana is false, continue to next (iteration)
2. i = banana → print(i) → banana
   if i == banana is true → then break executes & loop exits immediately.
3. Loop ends, "cherry" is never visited.

```
for i in fruits:
    if i == "banana":
        break
        print(i)
```

o/p → apple

↓

exits the loop when i is banana, but break comes before print

1. if i = apple → print() → appl
   i = apple
   1. if i == banana, false
      so print(i) print apple, & continue.
      i = banana
   2. i == banana, true
      → break executes immediately.
      so print(i) not executed.

else on for loop

else on for loop doesn't behaves like if-else.

else doesn't run before loop ends.

it runs only after the loop finishes completely. & only after break.

eg: fruits = ["apple", "banana", "cherry"]

```
for i in fruits:
    if (i == "banana"):
        print(i)

else:
    print("Hello")
```

o/p → banana
Hello.

Continue

continue in for loop is used to skip the current iteration & immediately move to the next one.

- the loop jumps to the next iteration
- it doesn't stop the loop. & doesn't exit the loop.

eg: fruits = ["apple", "banana", "cherry"]

```
for x in fruits:
    if (x == "banana"):
        continue
    print(x)
```

→ continue: → if we write like this, nothing prints.
print(i) anything after continue is ignored.

for loop with ranges

eg: for i in range(1,10,1): o/p → 1 2 3 ⋮ 9
print(i)

range(5) → 0 1 2 3 4

• range(1,10,2) → 1 3 5 7 9

• range(1,10,3) → 1 4 7

• range(2,10,2) → 2 4 6 8

→ to loop through a set of code a specified no. of times, we can use the ranges fn.

break, continue, pass → flow control statements.

break – stop the loop immediately

continue – skip current iteration & move next

pass – does nothing (place holder)

```
for i in range(1,10,1):
    if i==5:
        break
    print(i)
```

or

```
for i in range(1,10,1):
    if i==3
        continue
    print(i)
```

While Loops

→ can execute a set of statements as long as condition is true.

→ repeats while a condition holds.

initialization
while (condition):
    statement
    increment / decrement.

eg:
```
i=1
while (i<6):
    print(i)
    i=i+1
```

eg:
```
i=1
while (i<6):
    print(i)
    if (i==3):
        break
    i=i+1
```
o/p 1 2 3

eg:
```
i=1
while (i<6):
    i=i+1
    if (i==3):
        continue
    print(i)
```
→ 1 2 4 5 6.

eg: even or odd

```
i=1
while (i<=10):
    if (i%2==0):
        print(f"{i} is not prime")
    else:
        print("prime")
    i=i+1
```

for

```
Name = "minnu"
for i in Name:
    print(i)
```

```
fruits = ["apple", "banana", "cherry"]
for i in fruits:
    if i == "banana":
        break
    print(i)
```

o/p → apple

OR

```
if i == "banana":
    print(i)
    break
```

o/p → banana

OR

```
print(i)
if i == banana":
    break
```

continue

```
if v == "banana":
    continue
print(i)
```

while

```
Name = 'minnu'
i = 0
while (i < len(Name)):
    print(Name[i])
    i = i+1
```

```
i = 0
while (i <= 6):
    if (i == 3):
        print(i)
        break
    i = i+1
```

OR

```
while (i <= 6):
    print(i)
    if (i == 3):
        break
    i = i+1
```

~~while (i <= 6):~~
~~if i = i+1~~
~~if (i == 3)~~
~~if~~

```
while(i<=6):
    i = i+1
    if(i==3)
        continue
    print(i)
```

```
for i in range(1,10,1):
    print(i)
```

```
sum = 0
for i in range(11):
    sum = sum+i
    print(sum)
print(sum)
```

```
for i in range (1,10):
    if (i%2==0):
        print(f"{i} is even")
    else:
        print("odd")
```

```
fact=1
n=int(input())
for i in range(1,n+1):
    fact= fact*i
print(fact)
```

```
n=int(input())
for i in range(n+1):
    mul= n*i
    print(mul)
```

```
i=10
while (i<=10):
    print(i)
    i=i+1
```

```
sum=0
i=11
while (i<11):
    sum= sum+i
    print(sum)
```

```
i=1
while (i<=10):
    if (i%2==0):
        printf("even")
    else:
        print("odd")
    i=i+1
```

```
n=int(input())
fact=1
i=1
while (i<=n):
    fact= fact*i
    i=i+1
print(fact)
```

```
n= int(input())
i=1
while (i<=n):
```

Nested Loop

Loop unsude

for i in

for

print()

print(

oute

inn