By Sai Kumar

PYTHON – TRANSFER STATEMENTS

Transfer Statements

1) **break:**

We can use break statement inside loops to break loop execution based on some condition.

Eg 1)

```
for i in range(10):
  if i==7:
     print("processing is enough..plz break")
     break
  print(i)
```

Eg 2)

```
cart=[10,20,600,60,70]
for item in cart:
  if item>500:
    print("To place this order insurance must be required")
    break
 print(item)
```

2) **continue:**

We can use continue statement to skip current iteration and continue next iteration.

Eg 1: To print odd numbers in the range 0 to 9

```
for i in range(10):
  if i%2==0:
    continue
  print(i)
```

Eg 2:

1)
```
cart=[10,20,500,700,50,60]
for item in cart:
  if item>=500:
print("We cannot process this item :",item)
continue
print(item)
```

Eg 3:

```
numbers=[10,20,0,5,0,30]
for n in numbers:
if n==0:
print("Hey how we can divide with zero..just skipping")
continue
print("100/{} = {}".format(n,100/n))
```

Output

100/10 = 10.0

100/20 = 5.0

Hey how we can divide with zero..just skipping

100/5 = 20.0

Hey how we can divide with zero..just skipping

100/30 = 3.3333333333333335

Loops with else Block:

Inside loop execution, if break statement not executed, then only else part will be executed.
else means loop without break.

```
cart=[10,20,30,40,50]
for item in cart:
if item>=500:
print("We cannot process this order")
break
print(item)
else:
print("Congrats ...all items processed successfully")
```

Congrats ...all items processed successfully

Eg:

```
cart=[10,20,600,30,40,50]
for item in cart:
if item>=500:
print("We cannot process this order")
break
print(item)
else:
print("Congrats ...all items processed successfully")
```

3) pass statement:

pass is a keyword in Python.

In our programming syntactically if block is required which won't do anything then we can define that empty block with pass keyword.

pass

|- It is an empty statement

|- It is null statement

|- It won't do anything

Eg: if True:

SyntaxError: unexpected EOF while parsing

if True: pass ☐ valid

def m1():

SyntaxError: unexpected EOF while parsing

def m1(): pass

Use Case of pass:

Sometimes in the parent class we have to declare a function with empty body and child class responsible to provide proper implementation. Such type of empty body we can define by using pass keyword. (It is something like abstract method in Java)

Eg: def m1(): pass

for i in range(100):

if i%9==0:

print(i)

else:pass

del Statement:

del is a keyword in Python.

After using a variable, it is highly recommended to delete that variable if it is no longer required, so that the corresponding object is eligible for Garbage Collection.

We can delete variable by using del keyword.

x = 10

print(x)

del x

After deleting a variable we cannot access that variable otherwise we will get NameError.

x = 10

del x

print(x)

NameError: name 'x' is not defined.

Note: We can delete variables which are pointing to immutable objects.But we cannot delete the elements present inside immutable object.

s = "Sai"

print(s)

del s → valid

del s[0] → TypeError: 'str' object doesn't support item deletion

Difference between del and None:

In the case del, the variable will be removed and we cannot access that variable(unbind operation)

s = "Sai"

del s

print(s) □ NameError: name 's' is not defined.

But in the case of None assignment the variable won't be removed but the corresponding object is eligible for Garbage Collection (re bind operation). Hence after assigning with None value, we can access that variable.

s = "Sai"

s = None

print(s) → None

***** Happy Learning*****