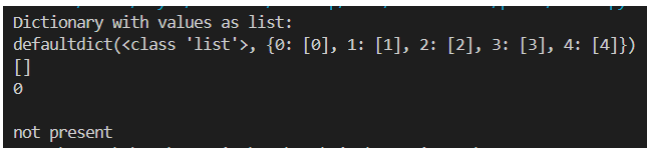
05-06

default dict use cases:

used to provide a default value for a nonexistent key in the dictionary, eliminating the need for checking if the key exists before using it (no KeyError is raised).

```
from collections import defaultdict

d1 = defaultdict(list)

for i in range(5):
    d1[i].append(i)

print("Dictionary with values as list:")
print(d1)
print(d1[6])

d2 = defaultdict(int)
print(d2[1])

d3 = defaultdict(str)
print(d3[2])

d4 = defaultdict(lambda : "not present")
print(d4['a'])
```



list operations:

list concat using f string:

```
name1='shyamani'
name2='jahnavi'
n3=f'{name1}{name2}'
print(n3)
```

reversing a list: (step is set to -1)

```
a = [1,2,3]
print(a[::-1])
```

walrus operator (:=)

It is an assignment operator (in python 3.8) which returns the value assigned as well.

Can be used inside expressions like if statements, while loops, and list comprehensions.

```
## without Walrus Operator
foods = list()
while True:
  food = input("What food do you like?: ")
  if food == "quit":
    break
  foods.append(food)

# with Walrus Operator
foods1 = list()
while (food := input("What food do you like? (type 'quit' to stop): ")) != "quit":
    foods1.append(food)
```

using type hints:

```
def multiply(num1: int, num2: int) -> int:
    return num1 * num2

def hello(name: str)-> str:
    return f'hello! {name}'

print(multiply(4, 5))
print(hello('Shyamani'))
```