PYTHON - CONTROL FLOW

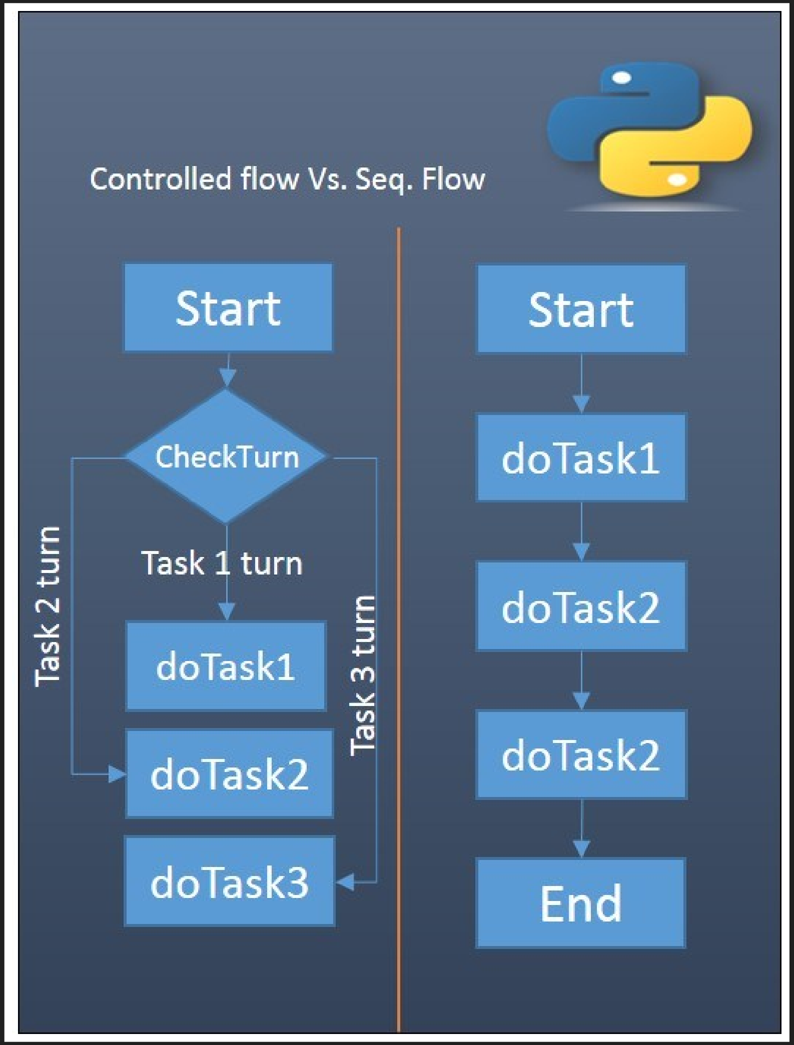
Bendik Egenes Dyrli

UiA University of Agder

OVERVIEW

1. Why control Flow..
2. Syntax
3. Things I forgot to mention in the last lecture
4. ...code?

WHY CONTROL FLOW..



SYNTAX

WHILE

```
while (expression is true)
    do something indefinitely
```

```
while True:
    print("Hi, i'm mr meeseeks")
```

FOR-LOOP

```
for target in iterable:
    do something over the length/size of the iterable
```

```
name = "Rick"
for letter in name:
    print(letter)
```

RANGE

```python
for number in range(10):
    print(number)
```

BREAK

like in C, it will break out of the innermost enclosing for or while loop

```
count = 0
while True:
    count += 1
    if count is 6:
        break
    else:
        print("Hi, i'm Mr Meeseeks")
```

CONTINUE

Also borrowed from C, continues with the next iteration of the for loop.

```
for num in range(2, 10):
    if num % 2 == 0:
        print("Found an even number", num)
        continue
    print("Found a number", num)
```

PASS

```
StrangeList = []

def futureFunction():
    pass

for k in StrangeList:
    pass

futureFunction()
```

PASS

```
while True:
    pass # Busy-wait for keyboard interrupt (CTRL + C )
```

FUNCTIONS

```
def Hello(name):

print (f"Hello {name}")
```

FUNCTIONS

```python
def Hello(name):
    print (f"Hello {name}")
```

```python
# Functions get called like this.
print (Hello("Riiiiiick"))

# Functions can be assigned to variables
say_my_name = Hello


say_my_name("Heisenberg")
```

DEFAULT ARGUMENTS VALUES

```
def where_is_rick(position_rick="Secret Lab")
    return (position_rick)

print(where_is_rick())
```

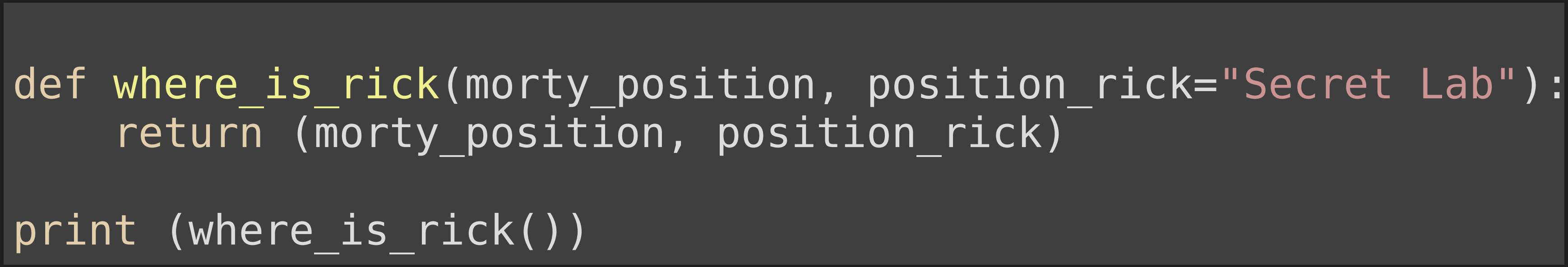
DEFAULT ARGUMENTS VALUES

```
def where_is_rick(position_rick="Secret Lab")
    return (position_rick)

print(where_is_rick("HOME"))
```

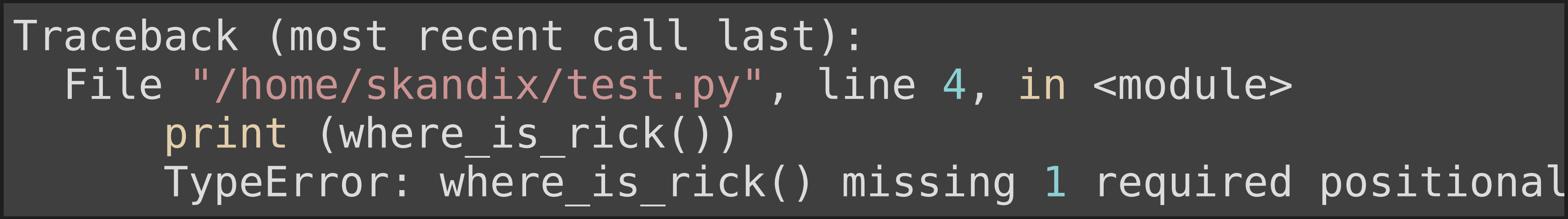
POSITIONAL ARGUMENTS

```
def where_is_rick(morty_position, position_rick="Secret Lab"):
    return (morty_position, position_rick)

print (where_is_rick())
```

POSITIONAL ARGUMENTS



```python
def where_is_rick(morty_position, position_rick="Secret Lab"):
    return (morty_position, position_rick)

print (where_is_rick())
```



```
Traceback (most recent call last):
  File "/home/skandix/test.py", line 4, in <module>
      print (where_is_rick())
      TypeError: where_is_rick() missing 1 required positional
```

POSITIONAL ARGUMENTS

```python
def where_is_rick(morty_position, position_rick="Secret Lab"):
    return (morty_position, position_rick)

print (where_is_rick("AT SCHOOL"))
```

ARBITRARY ARGUMENT LISTS

```
def smith_family(*args, seperator="/"):
    return (seperator.join(args))

    print (smith_family("rick", "morty", "summer"))
```

FUNCTIONS WITH TYPE HINTING

This is the closest you will get to type checking in python

```
def ransom_note(text: str) -> str:
    output = ""
    for letter_index in range(len(text)):
        if letter_index % 2 is 0:
            output += text[letter_index].lower()

        elif letter_index % 2 is not 0:
            output += text[letter_index].upper()
    print(output)

ransom_note("give me back my portal gun
and no one will get hurt!")
```

FUNCTIONS WITH TYPE HINTING

- List[int] – a list of integers
- List[str] - a list of strings
- Tuple[bool, float] – a tuple with two items: a boolean and a float
- Dict[str, int] – a dictionary accessed using a string key and holding an integer
- Dict[str, List[int]] – a dictionary with a string key holding a list of integers

```python
from typing import List

def IReturnAListWithStrings(text:str) -> List[str]:
    print(list(text))

IReturnAListWithStrings("HELLLOOO ")
```

LAMBDA EXPRESSIONS

Lambdas are small anonymous functions which can be created with the lambda keyword

```
Hello = lambda text: print(f"Hello {name}")
```

```
def Incrementing(n):
    return lambda x: x + n
```

CAN YOU USE LAMBDAS FOR EVERYTHING?

If you are crazy enough... yes !

```
from math import sin, cos, log, exp, pi
nr = lambda x,f,tol: print(f(x)) if abs((f(x)-x)/f(x))<=tol el
nr(2.5,lambda x: (2*x**3+3)/(3*x**2),9e-2)
nr(1500.,lambda x: (-x*log(x)+10001*x)/(5*x+1),1e-6)
nr(.5,lambda x: (x**2-sin(x)+x*cos(x)+2)/(2*x+cos(x)),.5e-8)
nr(-1.,lambda x: (x**2-sin(x)+x*cos(x)+2)/(2*x+cos(x)),5e-8)
nr(3.,lambda x: (x**2+10)/(2*x),.5e-8)
nr(-.5,lambda x: x-(log(x**2+1.)-exp(.4*x)*cos(pi*x))/((2*x)/(
```

DOCUMENT STRINGS

```
def recipie_concentrated_dark_matter():
    """
    Galactic Federation AIN'T GETTING SHIT YOOO... *BUUURP*
    """
    pass

print (recipie_concentrated_dark_matter.__doc__)
```

```
    Galactic Federation AIN'T GETTING SHIT YOOO... *BUUURP*
```

THINGS I FORGOT TO MENTION IN THE LAST LECTURE

TUPLE

Tuple is Immutable !

```
cords = (x,y)
for x in range(10):
    for y in range(10):
    print (x,y)
```

SLICES

```
[start:stop:step]
```

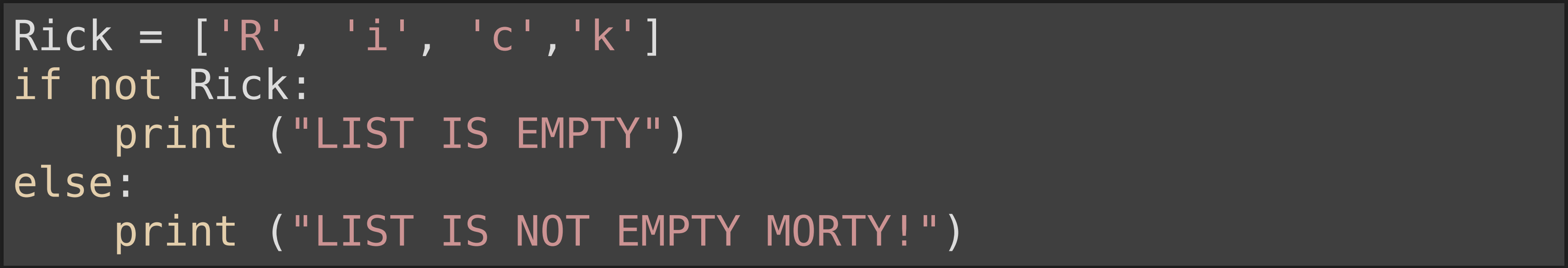
```
showMeWhatYouGot = "GET SCHWIFTY"
print (showMeWhatYouGot[1])
print (showMeWhatYouGot[:8])
print (showMeWhatYouGot[:8:2])
print (showMeWhatYouGot[-1])
```

SLICES

```
showMeWhatYouGot = "GET SCHWIFTY"
print (showMeWhatYouGot[1])
print (showMeWhatYouGot[:8])
print (showMeWhatYouGot[:8:2])
print (showMeWhatYouGot[-1])
```

G	E	T		S	C	H	W	I	F	T	Y
0	1	2	3	4	5	6	7	8	9	10	11

HOW TO CHECK IF LIST IS EMPTY



```
Rick = ['R', 'i', 'c','k']
if not Rick:
    print ("LIST IS EMPTY")
else:
    print ("LIST IS NOT EMPTY MORTY!")
```

RESOURCES

https://docs.python.org/3/tutorial/controlflow.html