CS1117 – Introduction to Programming

Dr. Jason Quinlan,

School of Computer Science and Information Technology

A TRADITION OF INDEPENDENT THINKING

UCC

University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Announcements

Continuous Assessment 1

Wednesday – 23rd October 3-4pm in room 107

Multiple Choice Questions

We will cover some sample questions

over the next few lectures

Announcements

CS1117 day off

I’m off campus next Monday, 21st October, so:

No morning class, no coding class, no office drop-in, no catch-up class and no evening class.

We will have labs on Tuesday/Wednesday and the Multiple Choice Quiz on Wednesday

University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Announcements

CS1117 day off

Monday, 28th October, is a bank holiday, and as such:

No morning class, no coding class, no office drop-in, no catch-up class and no evening class.

We will have labs on Tuesday/Wednesday and class on Wednesday as per normal

Announcements

Continuous Assessment 1

This Multiple Choice Quiz covers the first 5 weeks

This is a good chance for you to see if you understand what we have covered in the first 5 weeks

Announcements

Continuous Assessment 1

Available only on Canvas

So you will need a laptop, tablet, etc, to take the quiz.

If you do not have one of these, please let me know by email and I will arrange alternative access for the quiz.

If you have not received an email from me allocating you a space with alternative access, you must be in this room to access the MCQ.

Announcements

Continuous Assessment 1

Available only on Canvas

You will need access to Eduroam WiFi
so make sure you have signed up

IP filtering will be used for access to the quiz

University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Announcements

Continuous Assessment 1

Available only on Canvas

A code will be needed to access the quiz

This will be given out at the beginning of the class

Announcements

Continuous Assessment 1

Mobile phones will be turned off and placed on the desk in front of you.

You should not access online website for answers during the quiz

If you are seen surfing these sites, you will get a zero grade for this quiz.

Announcements

Continuous Assessment 1

This work must be your own,
so no asking your neighbour for answers

Do not take answers from others machines

There is no guarantee they are correct :)

You can bring pen and paper for rough work

Rough work does not need to be handed up.

Announcements

Sign-In

To practise new IP and access code filtering,
you will sign-in to a quiz today ☺

So, make sure you are on Eduroam

If you do not have access to Eduroam or can not sign-in,
let me know now or after class.

Announcements

Sign-In

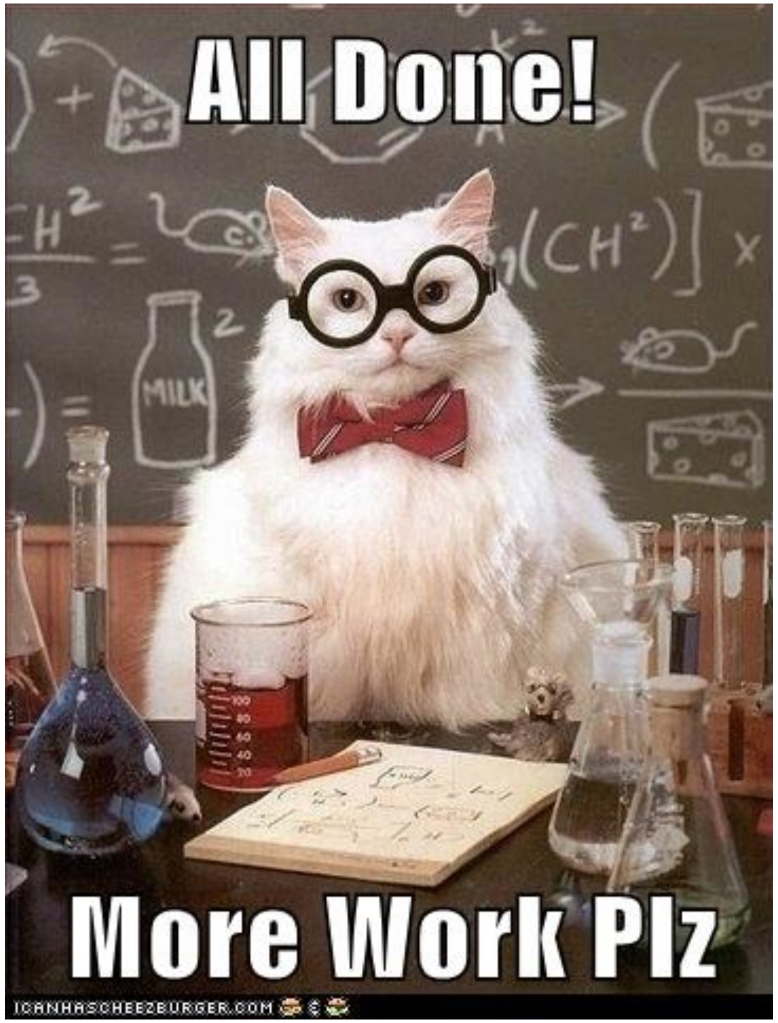
Let’s look at what sign-in looks now

Canvas / Student view

Canvas Student App

Let's Sign into this lecture now

Access Code:
12349

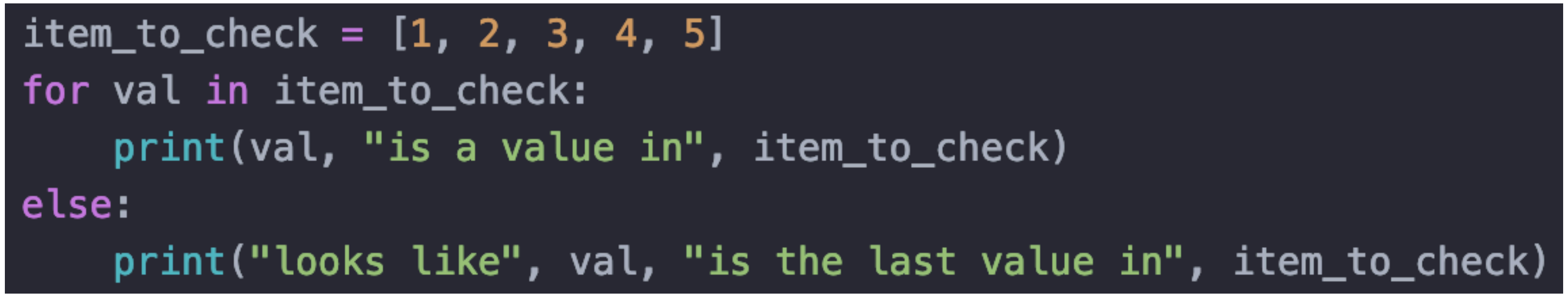


Recap

We now have 2 ways of getting information from an indexable structure, e.g., list, string, tuple,…

Indexing example 1

We now have 2 ways of getting information from an indexable structure, e.g., list, string, tuple,…

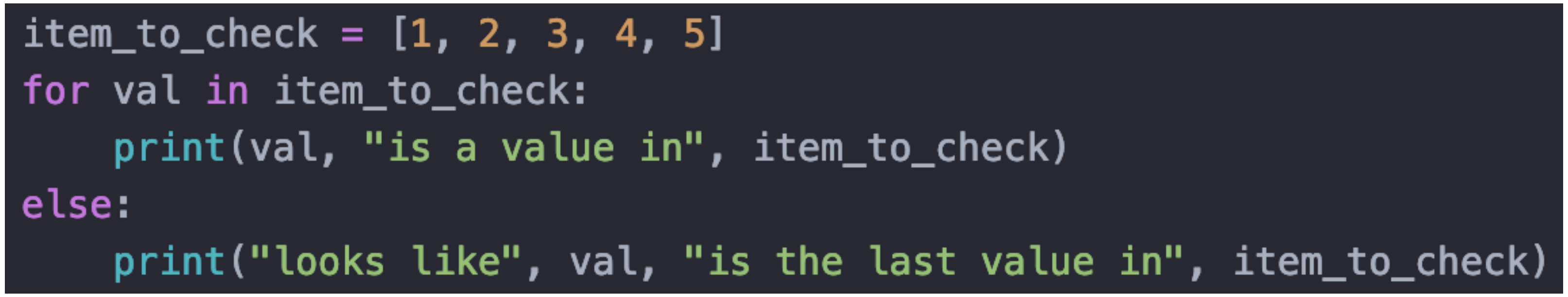


```python
item_to_check = [1, 2, 3, 4, 5]
for val in item_to_check:
    print(val, "is a value in", item_to_check)
else:
    print("looks like", val, "is the last value in", item_to_check)
```

Output:

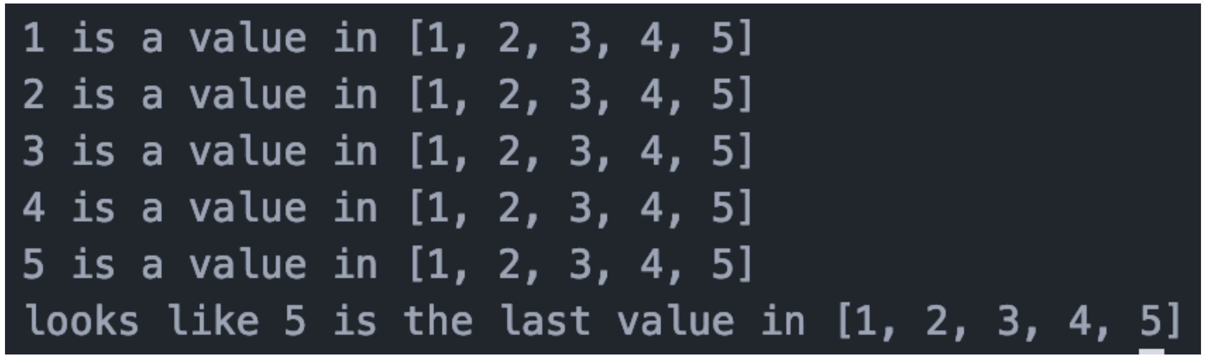
Indexing example 1

We now have 2 ways of getting information from an indexable structure, e.g., list, string, tuple,…



```python
item_to_check = [1, 2, 3, 4, 5]
for val in item_to_check:
    print(val, "is a value in", item_to_check)
else:
    print("looks like", val, "is the last value in", item_to_check)
```

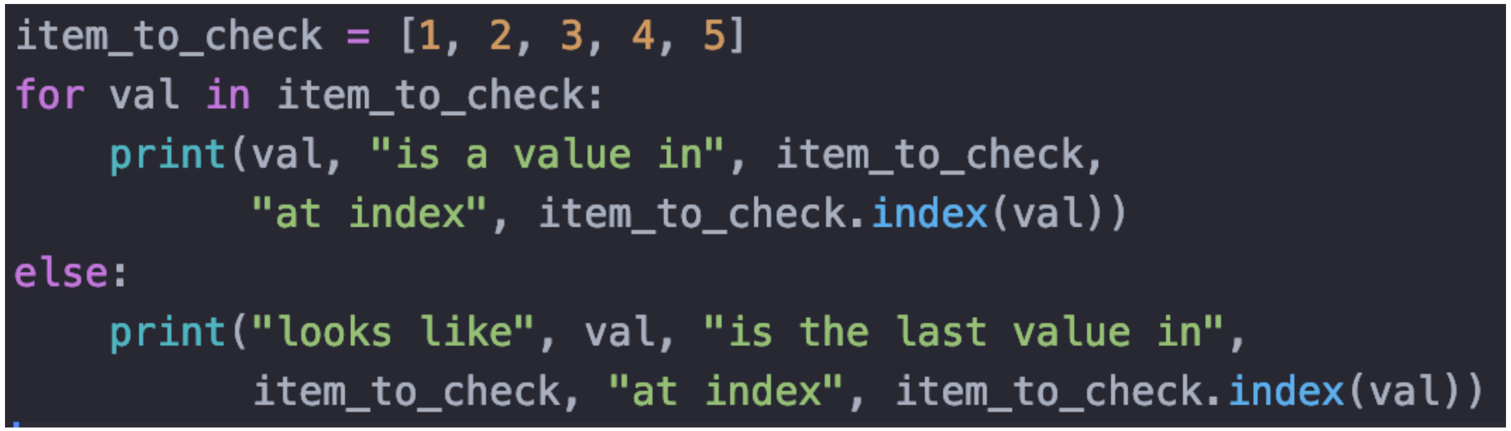
Output:



```
1 is a value in [1, 2, 3, 4, 5]
2 is a value in [1, 2, 3, 4, 5]
3 is a value in [1, 2, 3, 4, 5]
4 is a value in [1, 2, 3, 4, 5]
5 is a value in [1, 2, 3, 4, 5]
looks like 5 is the last value in [1, 2, 3, 4, 5]
```

Indexing example 1 (rewrite)

We now have 2 ways of getting information from an indexable structure, e.g., list, string, tuple,…

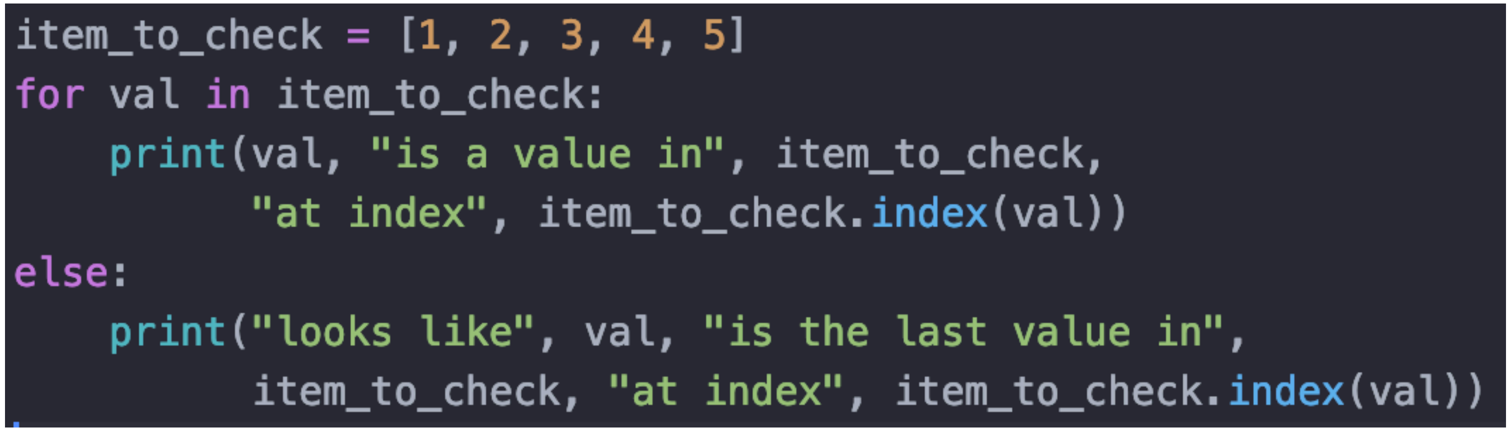


```python
item_to_check = [1, 2, 3, 4, 5]
for val in item_to_check:
    print(val, "is a value in", item_to_check,
          "at index", item_to_check.index(val))
else:
    print("looks like", val, "is the last value in",
          item_to_check, "at index", item_to_check.index(val))
```

Output:

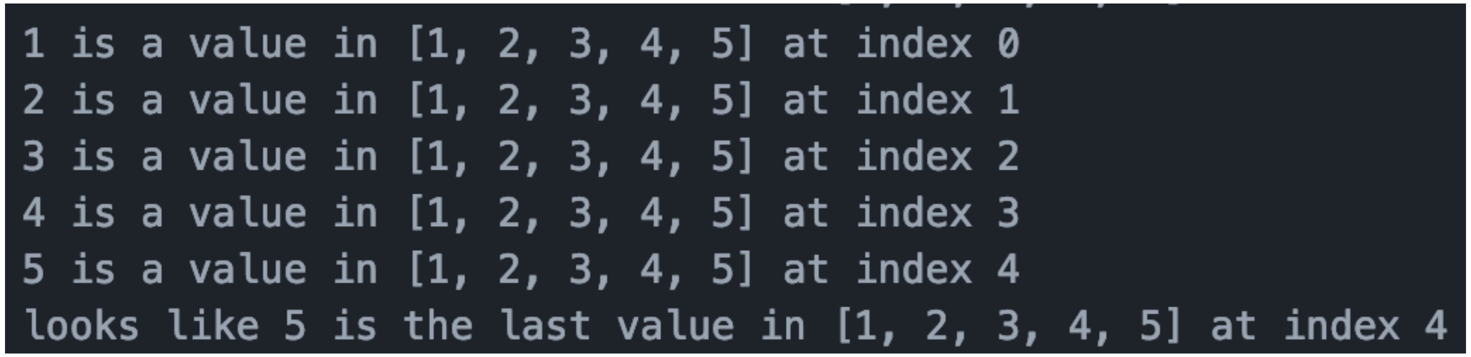
Indexing example 1 (rewrite)

We now have 2 ways of getting information from an indexable structure, e.g., list, string, tuple,…



```python
item_to_check = [1, 2, 3, 4, 5]
for val in item_to_check:
    print(val, "is a value in", item_to_check,
          "at index", item_to_check.index(val))
else:
    print("looks like", val, "is the last value in",
          item_to_check, "at index", item_to_check.index(val))
```

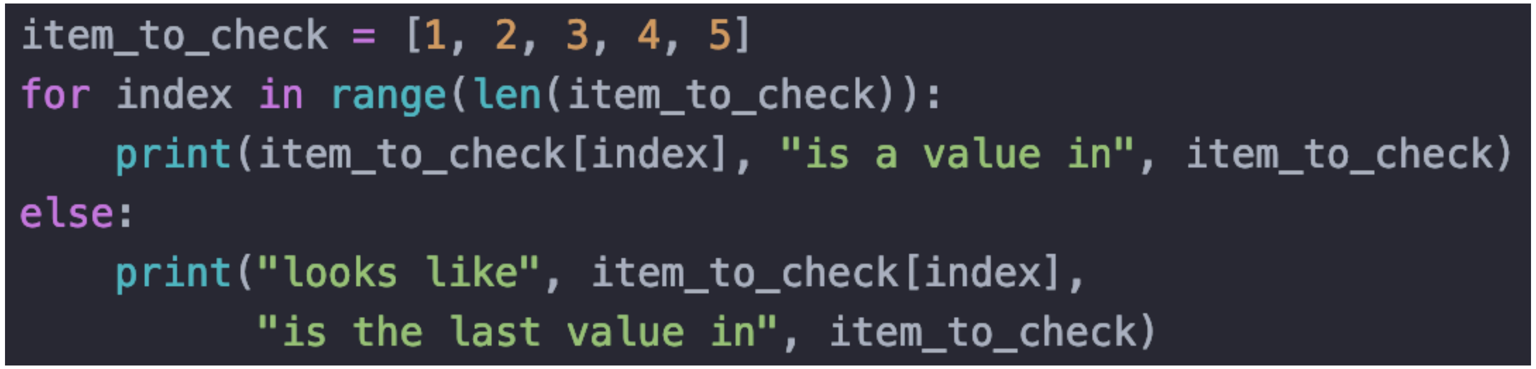
Output:



```
1 is a value in [1, 2, 3, 4, 5] at index 0
2 is a value in [1, 2, 3, 4, 5] at index 1
3 is a value in [1, 2, 3, 4, 5] at index 2
4 is a value in [1, 2, 3, 4, 5] at index 3
5 is a value in [1, 2, 3, 4, 5] at index 4
looks like 5 is the last value in [1, 2, 3, 4, 5] at index 4
```

Indexing example 2

We now have 2 ways of getting information from an indexable structure, e.g., list, string, tuple,…

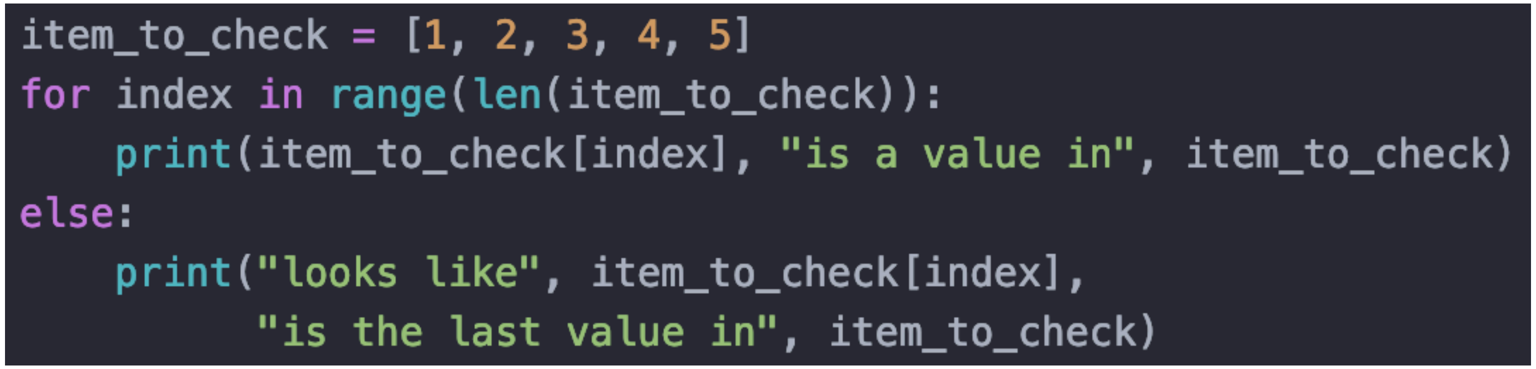


```
item_to_check = [1, 2, 3, 4, 5]
for index in range(len(item_to_check)):
    print(item_to_check[index], "is a value in", item_to_check)
else:
    print("looks like", item_to_check[index],
          "is the last value in", item_to_check)
```

Output:

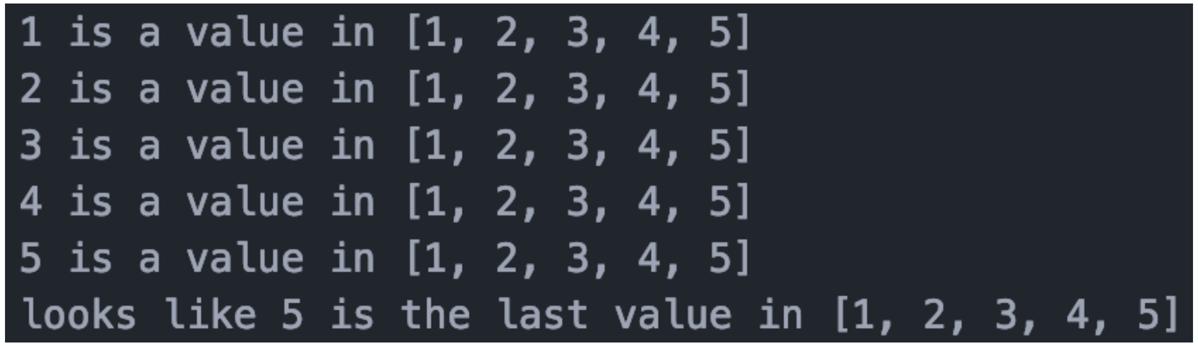
Indexing example 2

We now have 2 ways of getting information from an indexable structure, e.g., list, string, tuple,…



```python
item_to_check = [1, 2, 3, 4, 5]
for index in range(len(item_to_check)):
    print(item_to_check[index], "is a value in", item_to_check)
else:
    print("looks like", item_to_check[index],
          "is the last value in", item_to_check)
```

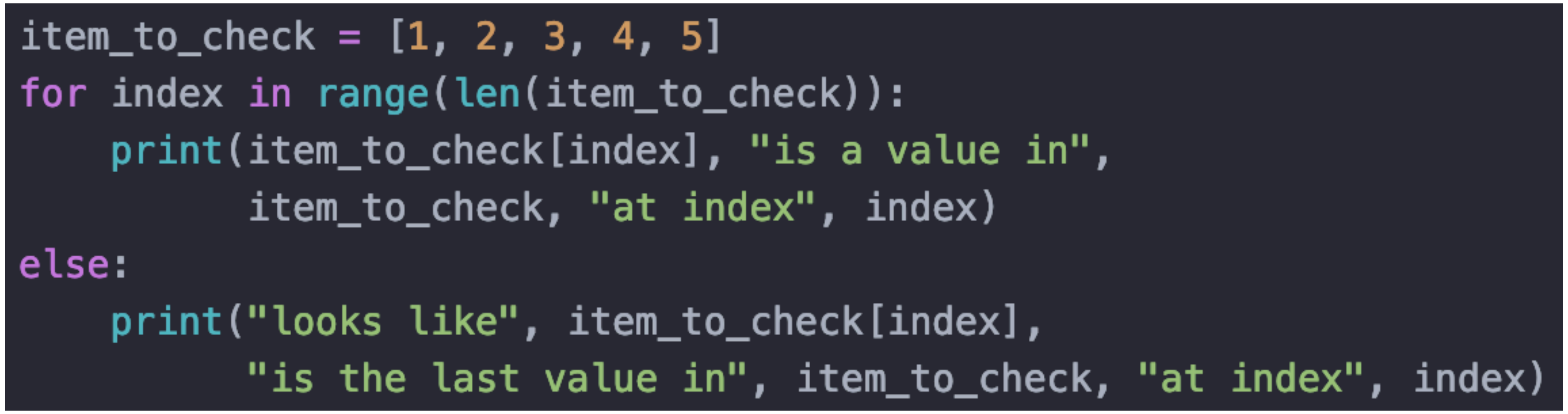
Output:



```
1 is a value in [1, 2, 3, 4, 5]
2 is a value in [1, 2, 3, 4, 5]
3 is a value in [1, 2, 3, 4, 5]
4 is a value in [1, 2, 3, 4, 5]
5 is a value in [1, 2, 3, 4, 5]
looks like 5 is the last value in [1, 2, 3, 4, 5]
```

Indexing example 2 (rewrite)

We now have 2 ways of getting information from an indexable structure, e.g., list, string, tuple,…



```python
item_to_check = [1, 2, 3, 4, 5]
for index in range(len(item_to_check)):
    print(item_to_check[index], "is a value in",
          item_to_check, "at index", index)
else:
    print("looks like", item_to_check[index],
          "is the last value in", item_to_check, "at index", index)
```

Output:

Indexing example 2 (rewrite)

We now have 2 ways of getting information from an indexable structure, e.g., list, string, tuple,…

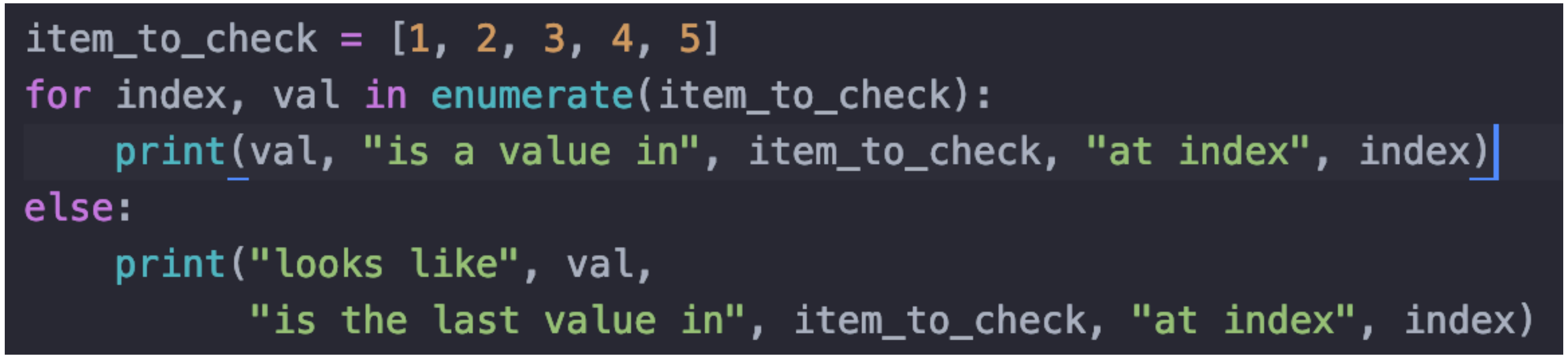
```python
item_to_check = [1, 2, 3, 4, 5]
for index in range(len(item_to_check)):
    print(item_to_check[index], "is a value in",
          item_to_check, "at index", index)
else:
    print("looks like", item_to_check[index],
          "is the last value in", item_to_check, "at index", index)
```

Output:

```
1 is a value in [1, 2, 3, 4, 5] at index 0
2 is a value in [1, 2, 3, 4, 5] at index 1
3 is a value in [1, 2, 3, 4, 5] at index 2
4 is a value in [1, 2, 3, 4, 5] at index 3
5 is a value in [1, 2, 3, 4, 5] at index 4
looks like 5 is the last value in [1, 2, 3, 4, 5] at index 4
```

Recap

Let's learn a third way:



```python
item_to_check = [1, 2, 3, 4, 5]
for index, val in enumerate(item_to_check):
    print(val, "is a value in", item_to_check, "at index", index)
else:
    print("looks like", val,
          "is the last value in", item_to_check, "at index", index)
```

enumerate – returns the index and value at each value

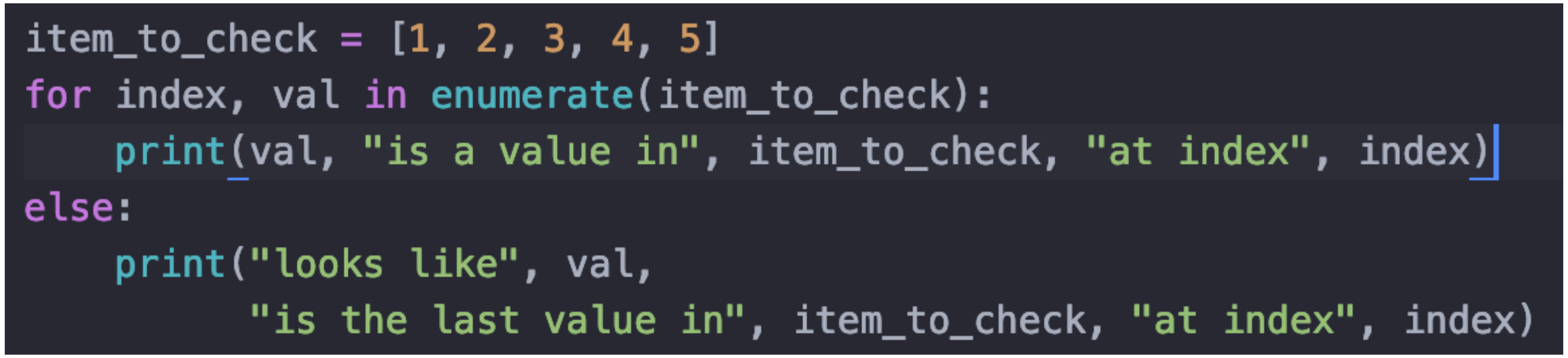
from an indexable structure

Reduces the need to use <input>.index() or <input>[index] to get

the index/value from <input>

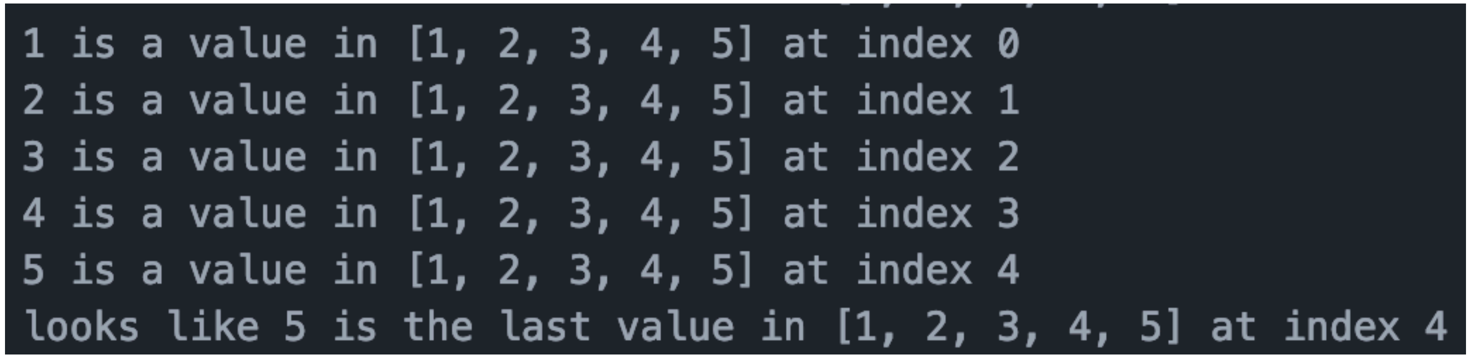
Recap

Let’s learn a third way:



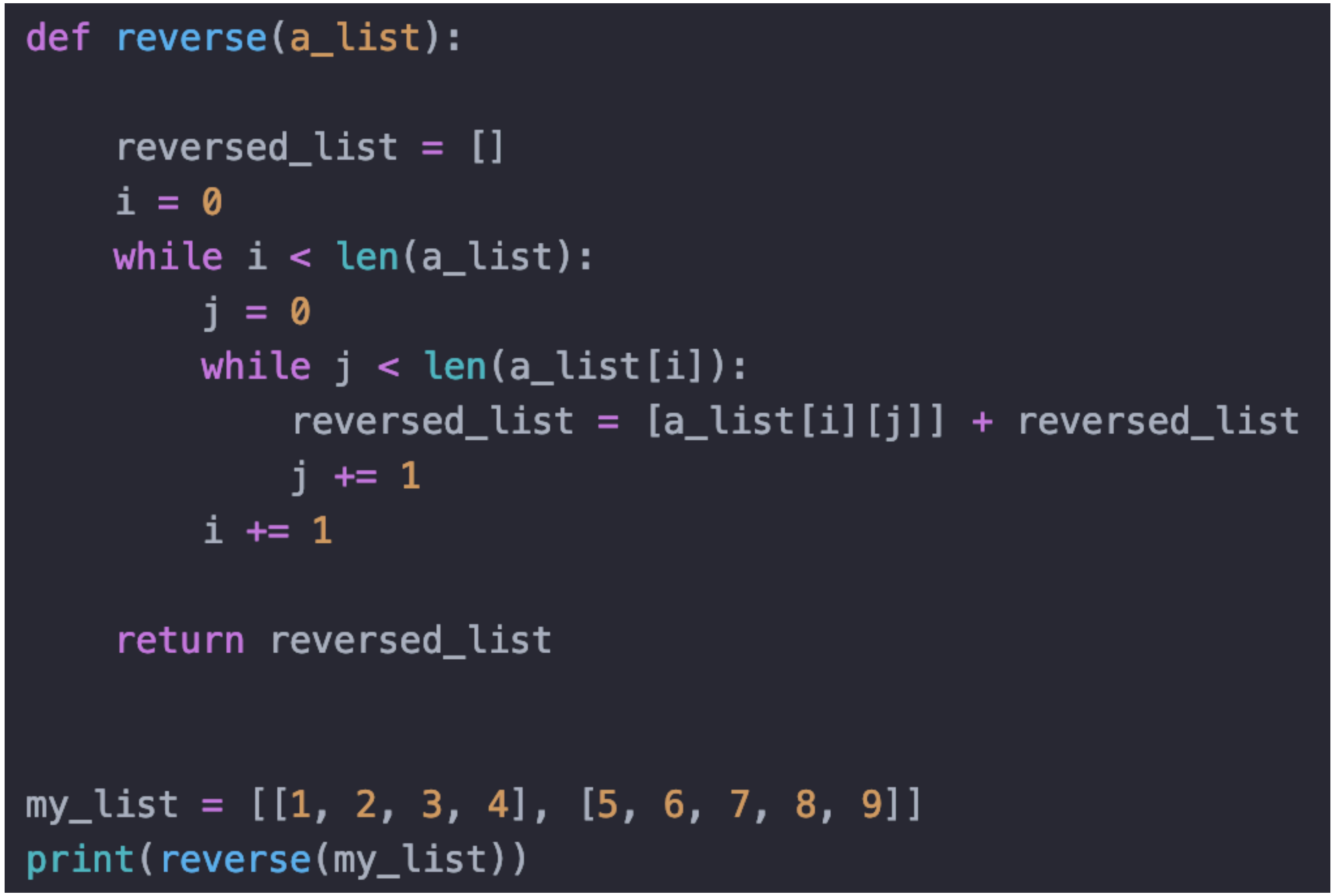
```python
item_to_check = [1, 2, 3, 4, 5]
for index, val in enumerate(item_to_check):
    print(val, "is a value in", item_to_check, "at index", index)
else:
    print("looks like", val,
          "is the last value in", item_to_check, "at index", index)
```

Output:



```
1 is a value in [1, 2, 3, 4, 5] at index 0
2 is a value in [1, 2, 3, 4, 5] at index 1
3 is a value in [1, 2, 3, 4, 5] at index 2
4 is a value in [1, 2, 3, 4, 5] at index 3
5 is a value in [1, 2, 3, 4, 5] at index 4
looks like 5 is the last value in [1, 2, 3, 4, 5] at index 4
```

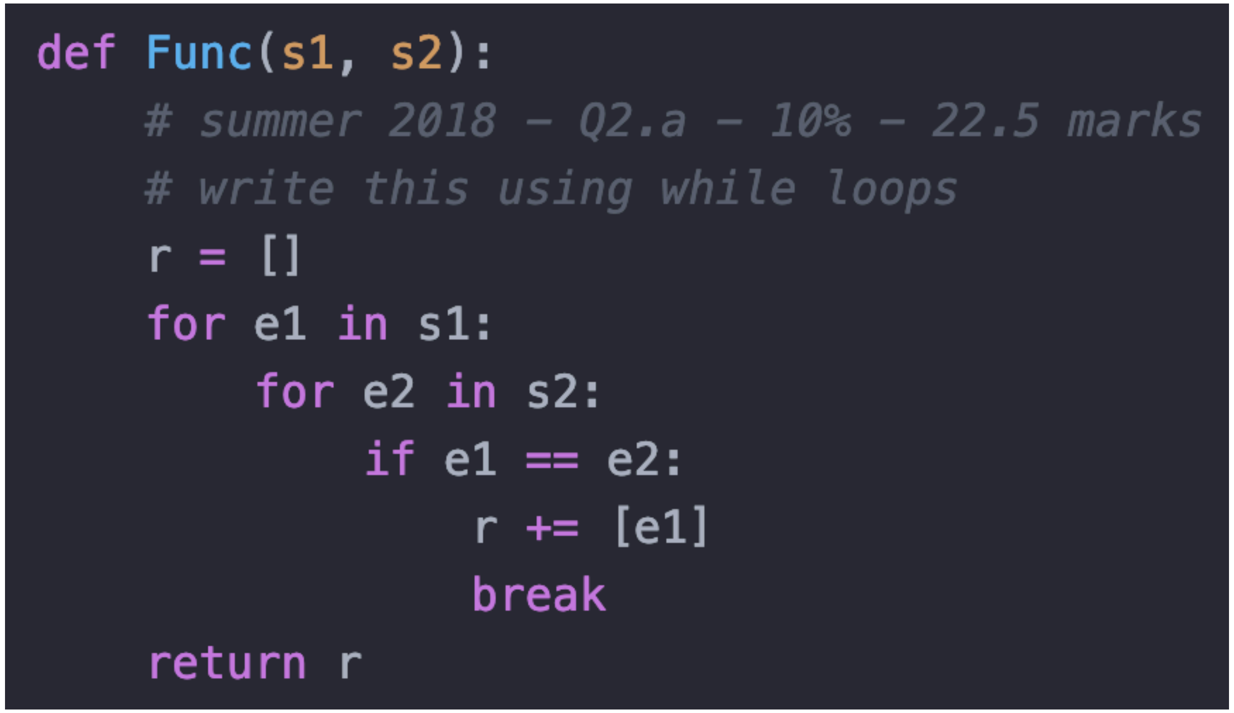
while from week 5



```python
def reverse(a_list):

    reversed_list = []
    i = 0
    while i < len(a_list):
        j = 0
        while j < len(a_list[i]):
            reversed_list = [a_list[i][j]] + reversed_list
            j += 1
        i += 1

    return reversed_list


my_list = [[1, 2, 3, 4], [5, 6, 7, 8, 9]]
print(reverse(my_list))
```

We have seen nested while loops

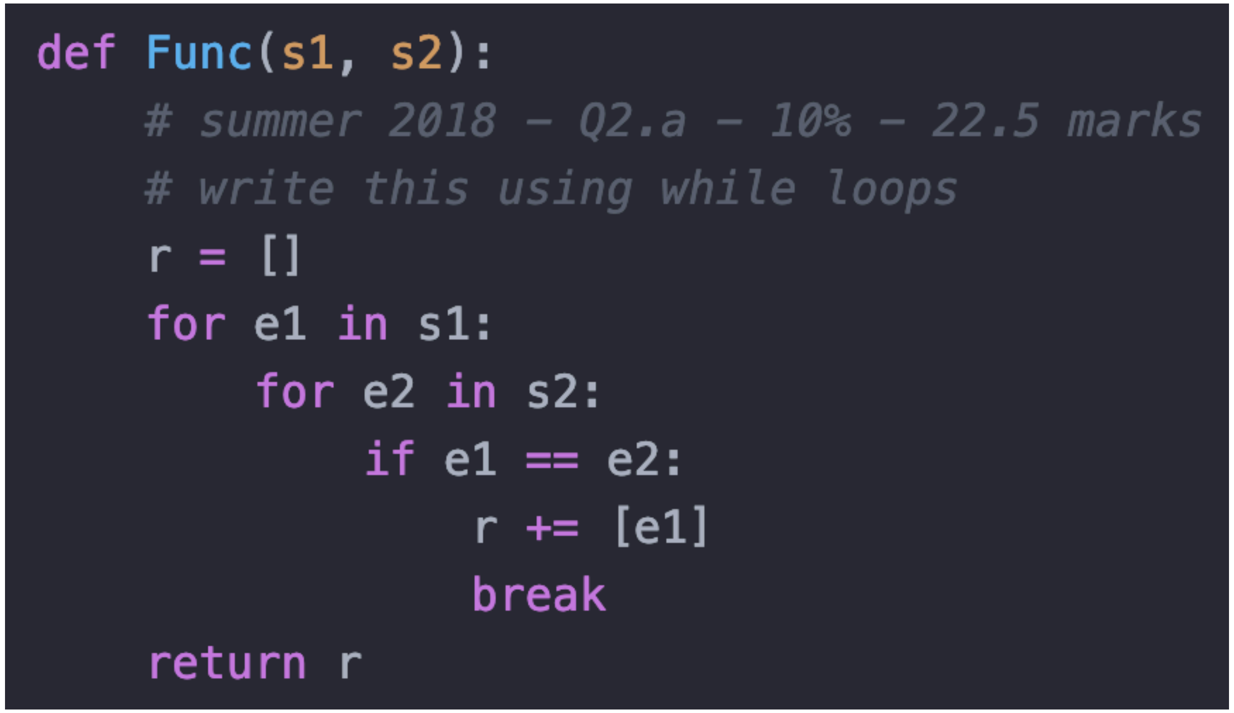
Summer 2019 exam question



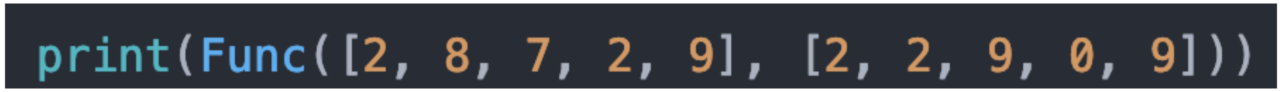
```python
def Func(s1, s2):
    # summer 2018 - Q2.a - 10% - 22.5 marks
    # write this using while loops
    r = []
    for e1 in s1:
        for e2 in s2:
            if e1 == e2:
                r += [e1]
                break
    return r
```

To explain nested for loops
I’m going to use an exam question

Summer 2019 exam question

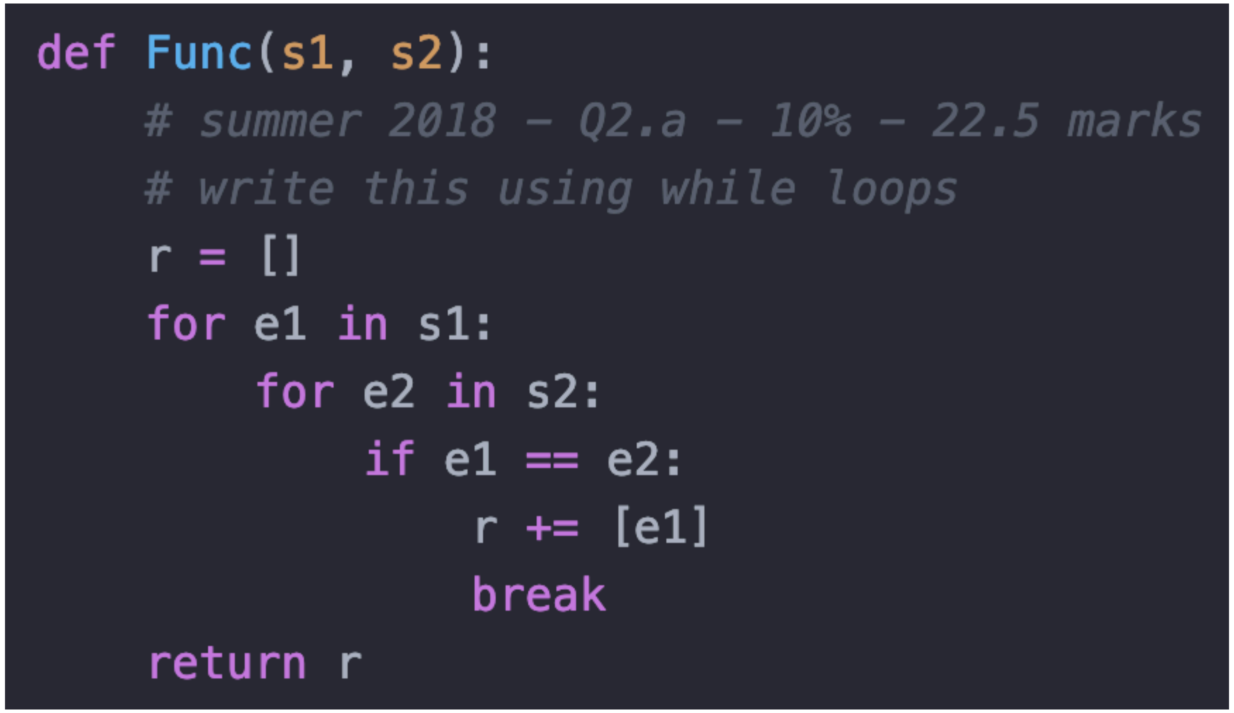


```python
def Func(s1, s2):
    # summer 2018 - Q2.a - 10% - 22.5 marks
    # write this using while loops
    r = []
    for e1 in s1:
        for e2 in s2:
            if e1 == e2:
                r += [e1]
                break
    return r
```

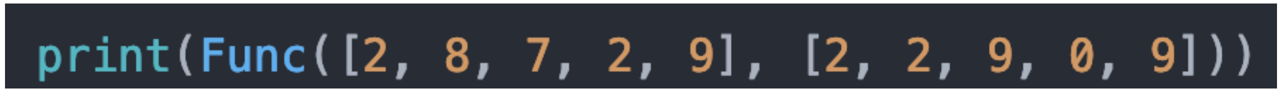


```python
print(Func([2, 8, 7, 2, 9], [2, 2, 9, 0, 9]))
```

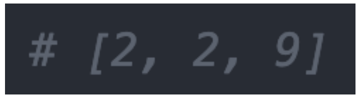
Summer 2019 exam question



```python
def Func(s1, s2):
    # summer 2018 - Q2.a - 10% - 22.5 marks
    # write this using while loops
    r = []
    for e1 in s1:
        for e2 in s2:
            if e1 == e2:
                r += [e1]
                break
    return r
```

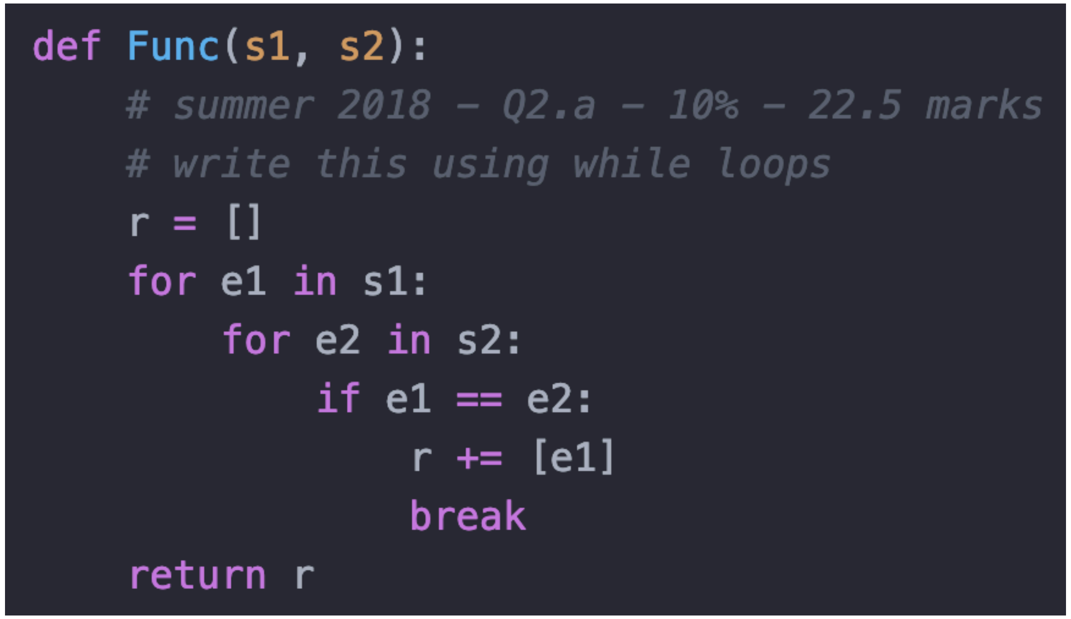


```python
print(Func([2, 8, 7, 2, 9], [2, 2, 9, 0, 9]))
```



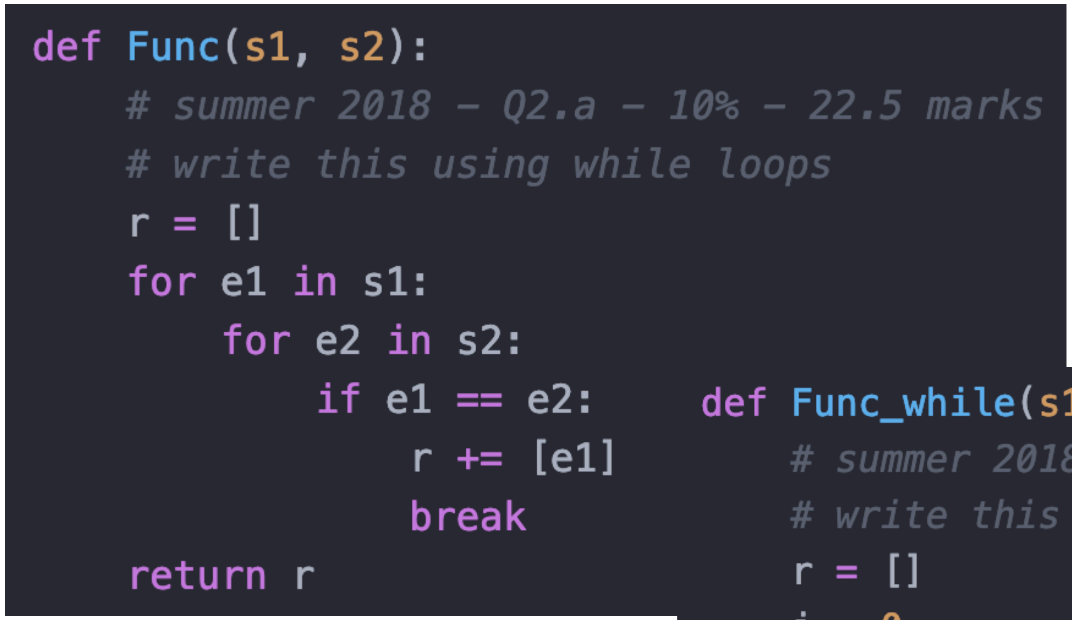
```python
# [2, 2, 9]
```

Summer 2019 exam question

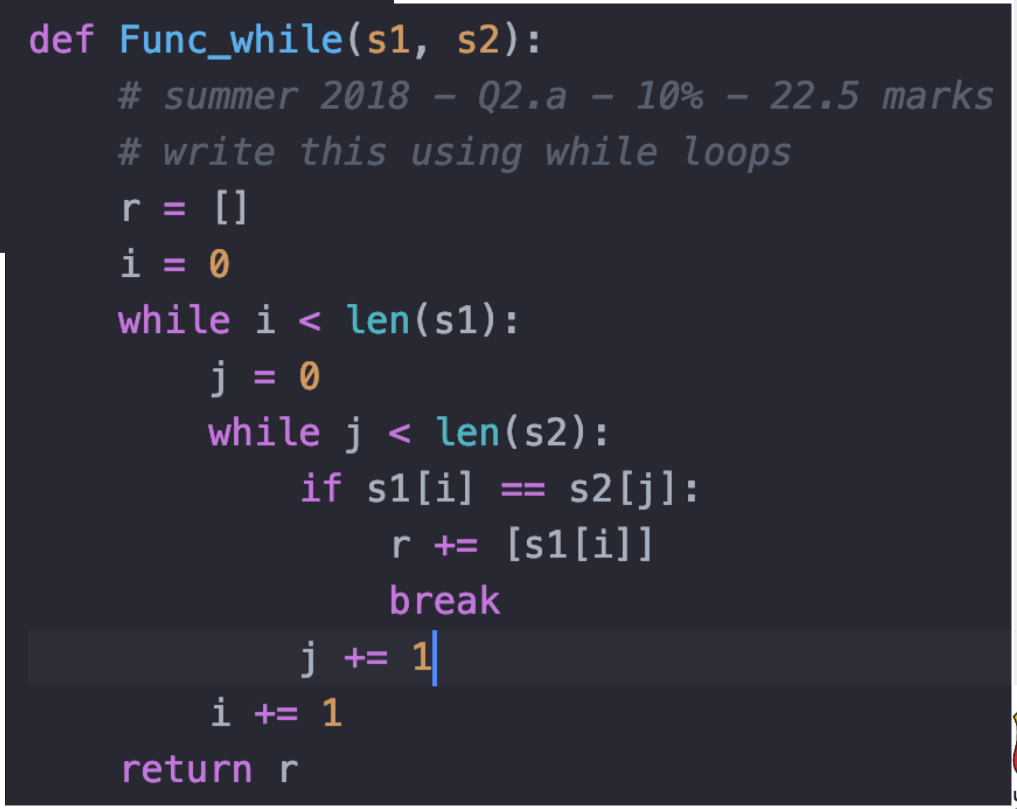


```python
def Func(s1, s2):
    # summer 2018 - Q2.a - 10% - 22.5 marks
    # write this using while loops
    r = []
    for e1 in s1:
        for e2 in s2:
            if e1 == e2:
                r += [e1]
                break
    return r
```

Summer 2019 exam question



```python
def Func(s1, s2):
    # summer 2018 - Q2.a - 10% - 22.5 marks
    # write this using while loops
    r = []
    for e1 in s1:
        for e2 in s2:
            if e1 == e2:
                r += [e1]
                break
    return r
```



```python
def Func_while(s1, s2):
    # summer 2018 - Q2.a - 10% - 22.5 marks
    # write this using while loops
    r = []
    i = 0
    while i < len(s1):
        j = 0
        while j < len(s2):
            if s1[i] == s2[j]:
                r += [s1[i]]
                break
            j += 1
        i += 1
    return r
```

University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Live Coding Time…

WUBBA LUBBA DUB DUB!