Day 5: *Control Structures: for loops and if statements*

After spending some time in Munich, Dot moved on to their next destination: London, the capital of the United Kingdom. When they arrived, the first thing they did was settle in at a tea room over a steaming cup of English Breakfast. Then, as they let the soothing beverage warm them and the caffeine perk them up, Dot started to think about what they wanted to visit in London. The clock tower with Big Ben, London Bridge, the Natural History Museum...the possibilities were endless and a bit overwhelming.

Dot overheard some other tourists talking about their day at the table next to them. “I’m telling you, we must have waited a good 45 minutes just to be able to see the structure!” a man exclaimed to his partner. Wait times are something Dot didn’t think about — Dot hated waiting, even to see something magnificent and historical. Actually, waiting in line was Dot’s number one fear, coincidentally. If they waited in line for more than 15 minutes, they might have an emotional meltdown and become psychologically disconnected from the world around them, which wouldn’t bode well for their invigorating, joyful vacation. So let’s help Dot sidestep their #1 fear by using data to guide them towards only landmarks with a waiting time of fewer than fifteen minutes.

Tutorial

In this challenge, you'll use **conditional logic** and a **for loop** to solve the problem.

For Loops

In programming, a **loop** is a block of code that's executed repeatedly, either for a set number of times, or until a certain condition is met.

A **for loop** is a type of loop that is executed for a set number of times, iterated over the number of items in a sequence. To learn more about for loops, read this article.

```
lst = [1,2,3,4,5]


for i in range(len(lst)):
    print(i)
    '''insert code'''
```

In the above code, the range() function creates a sequence of numbers, and the len() function outputs the length of a list.

Conditional Logic

Conditional logic refers to the execution of different actions based on whether a certain condition is met. In programming, these conditions are expressed by a set of symbols called **Boolean Operators**.

Boolean Comparator	Example	Meaning
>	x > y	x is greater than y

Boolean Comparator	Example	Meaning
>=	x >= y	x is greater than or equal to y
<	x < y	x is less than y
<=	x <= y	x is less than or equal to y
!=	x != y	x is not equal to y
==	x == y	x is equal to y

Conditional logic is essential to **control flow statements**. Control flows are expressed in **if...else** statements. To read more on if...else statements, read this article before proceeding.

```
#If Statement
if condition is True:
    '''
    execute body of code
    '''

#If Else Statement
if var > 10: #if var is greater than 10
    '''
    execute body of code
    '''
else: #if none of the above conditions are true, execute this code
    '''
    execute body of code
    '''

#Multiple If...else statements
if var > 10: #if var is greater than 10
    '''
    execute body of code
    '''

elif var > 5 and <= 10: #if var is greater than 5 but less than or equal to 10
    '''
    execute body of code
    '''
else: #if none of the above statement is true, execute code
    '''
    execute body of code
    '''
```

Note: *If...else* statements can be executed within a *for loop*. However, be aware of the indentation.

Challenge

Dot wants to visit all the London landmarks where the wait time is less than 15 minutes. We have a dictionary called `landmarks`, within which landmark names are keys and wait times are their respective values.

Using Python, develop a list of the landmarks where the wait time is less than 15 minutes. What will be the length of the list?

In [2]:
```
# dictionary where
landmarks = {
    "Big Ben": 12,
    "Tower Bridge": 25,
    "Buckingham Palace": 15,
    "Madame Tussauds": 25,
    "London Eye": 40,
    "Tower of London": 25,
    "Emirates Air Line cable car": 16,
    "London Transport Museum": 7,
    "Wembley Stadium": 8,
    "Hyde Park": 0,
    "The View from The Shard": 14
}
```

In [3]:
```
# SOLUTION

keys = list(landmarks.keys())
values = list(landmarks.values())


res = []
for i in range(len(keys)):
    if values[i] < 15:
        res.append(keys[i])
```

In [4]:
```
res
```

Out[4]:
```
['Big Ben',
 'London Transport Museum',
 'Wembley Stadium',
 'Hyde Park',
 'The View from The Shard']
```

In [5]:
```
len(res)
```

Out[5]:
```
5
```