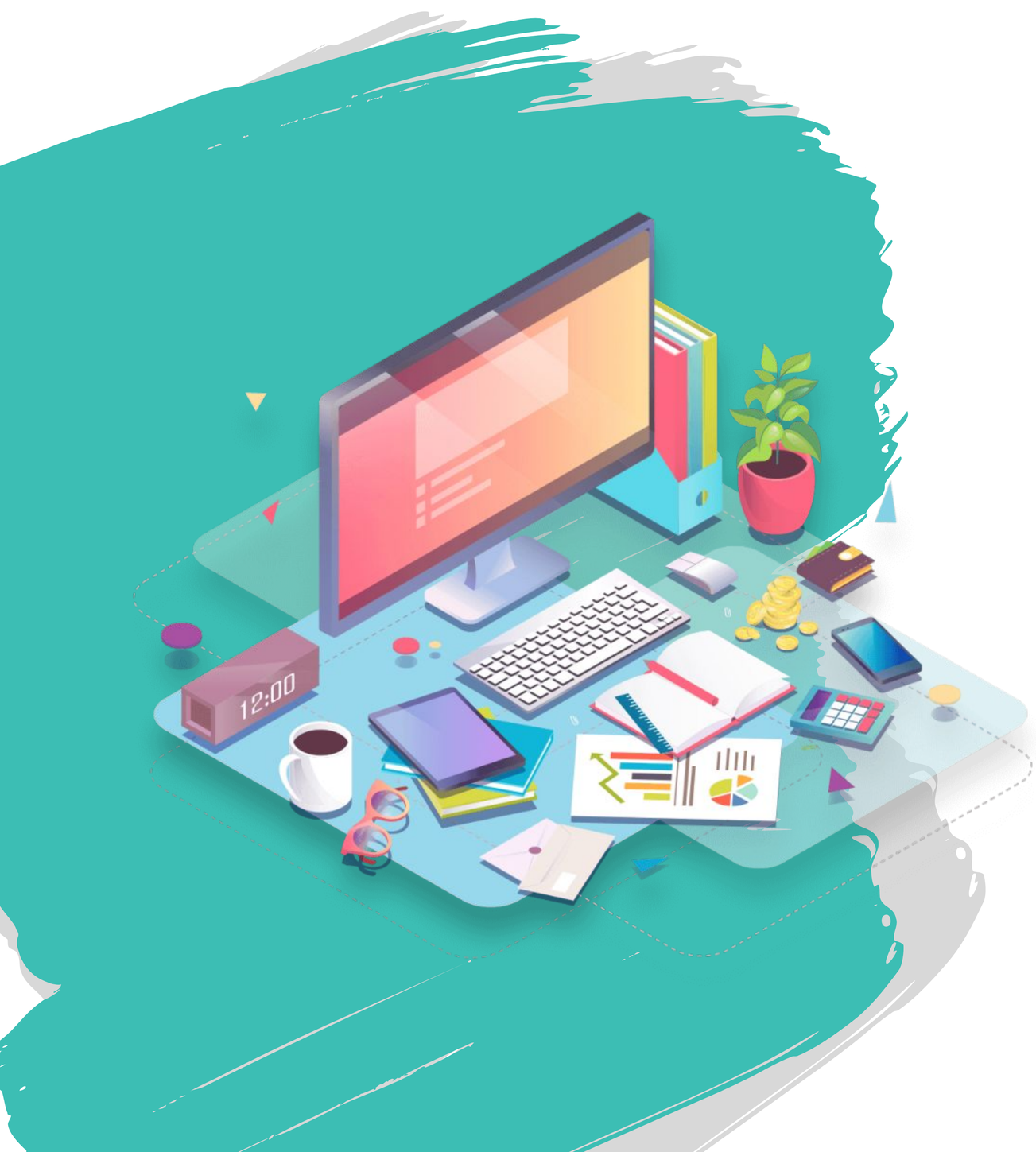


Collection-Based Looping vs Index-Based Looping

Comparison

```
# collection-based looping
for subject in subject_list:
    print(subject)
```

```
# index-based looping
for index in range(len(subject_list)):
    print(subject_list[index])
```

OUTPUT

```
Math
English
Chinese
Chemistry
Physics
```

EXPLANATION

These two blocks of code produces the same output!

So when do we use collection-based looping or index-based looping?

Comparison

```
# collection-based looping
for subject in subject_list:
    print(subject)
```

```
# index-based looping
for index in range(len(subject_list)):
    print(subject_list[index])
```

OUTPUT

Math
English
Chinese
Chemistry
Physics

EXPLANATION

As a rule of thumb, use collection-based looping. Only use index-based looping when you have to loop through multiple lists/tuples at the same time.

Let’s now understand how index-based looping works by simplifying the code!

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

for index in range(len(subject_list)):
    print(subject_list[index])
```

OUTPUT

Math
English
Chinese
Chemistry
Physics

EXPLANATION

Let's talk about how `range(len(subject_list))` works later. For now, let's replace it with the number of elements in the list!

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

for index in range(5):
    print(subject_list[index])
```

OUTPUT

Math
English
Chinese
Chemistry
Physics

EXPLANATION

Now let’s go through each iteration and see how index-based looping work!

But right before that, let’s show what happens when we just print index!

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

for index in range(5):
    print(index)
```

OUTPUT

```
0
1
2
3
4
```

EXPLANATION

If we simply print the index, the output will be what you see on the left!

Now let's go back to lesson!

Index-based looping (Iter #1)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
      0
for index in range(5):
    print(subject_list[index])
```

OUTPUT

EXPLANATION

In the first iteration, the variable *index* takes on the first number in the *range* iterable, i.e. 0.

Index-based looping (Iter #1)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
      0
for index in range(5):   0
    print(subject_list[index])
```

OUTPUT

EXPLANATION

Since the variable *index* is used in the next line, the variable *index* also takes on the number 0.

Let's replace index with 0.

Index-based looping (Iter #1)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

for index in range(5):
    print(subject_list[0])
```

0

OUTPUT

EXPLANATION

As we have learnt from list indexing, `subject_list[0]` will output `'Math'` as it is in the 0th index of `subject_list`.

Let's replace it with `'Math'`.

Index-based looping (Iter #1)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

for index in range(5):
    print('Math')
```

OUTPUT

Math

EXPLANATION

Finally, let's run the line of code.

Now, let's move to the next iteration! But just before that, let's refresh ourselves on the output of the range function

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

for index in range(5):
    print(index)
```

OUTPUT

```
0
1
2
3
4
```

EXPLANATION

In the second iteration, the number 1 will be printed.

Now, let's go back to the second iteration of index-based looping!

Index-based looping (Iter #2)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
      1
for index in range(5):   1
    print(subject_list[index])
```

OUTPUT

Math

EXPLANATION

This time, the index takes on the value 1 in the second iteration.

Index-based looping (Iter #2)

1

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

for index in range(5):
    print(subject_list[1])
```

OUTPUT

Math

EXPLANATION

This time, `subject_list[1]` will output `'English'`.

Again, let's replace it with `'English'`.

Index-based looping (Iter #2)

1

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

for index in range(5):
    print('English')
```

OUTPUT

Math
English

EXPLANATION

Finally, let's run the line of code.

Now, let's go through this quickly

Index-based looping (Iter #3)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
      2
for index in range(5):   2
    print(subject_list[index])
```

OUTPUT

Math
English

EXPLANATION

For the third iteration, the index takes on the value 2.

Index-based looping (Iter #3)

```
                                    2
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

for index in range(5):
    print(subject_list[2])
```

OUTPUT

```
Math
English
Chinese
```

EXPLANATION

This time, `subject_list[2]` will output `'Chinese'` and now let's print it out!

Index-based looping (Iter #4)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
      3
for index in range(5):   3
    print(subject_list[index])
```

OUTPUT

Math
English
Chinese

EXPLANATION

For the fourth iteration, the index takes on the value 3.

Index-based looping (Iter #4)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

for index in range(5):
    print(subject_list[3])
```

3

OUTPUT

```
Math
English
Chinese
Chemistry
```

EXPLANATION

This time, `subject_list[3]` will output `Chemistry` and now let's print it out!

Index-based looping (Iter #5)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
      4
for index in range(5):  4
    print(subject_list[index])
```

OUTPUT

```
Math
English
Chinese
Chemistry
```

EXPLANATION

For the third iteration, the index takes on the value 4.

Index-based looping (Iter #5)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

for index in range(5):
    print(subject_list[4])
```

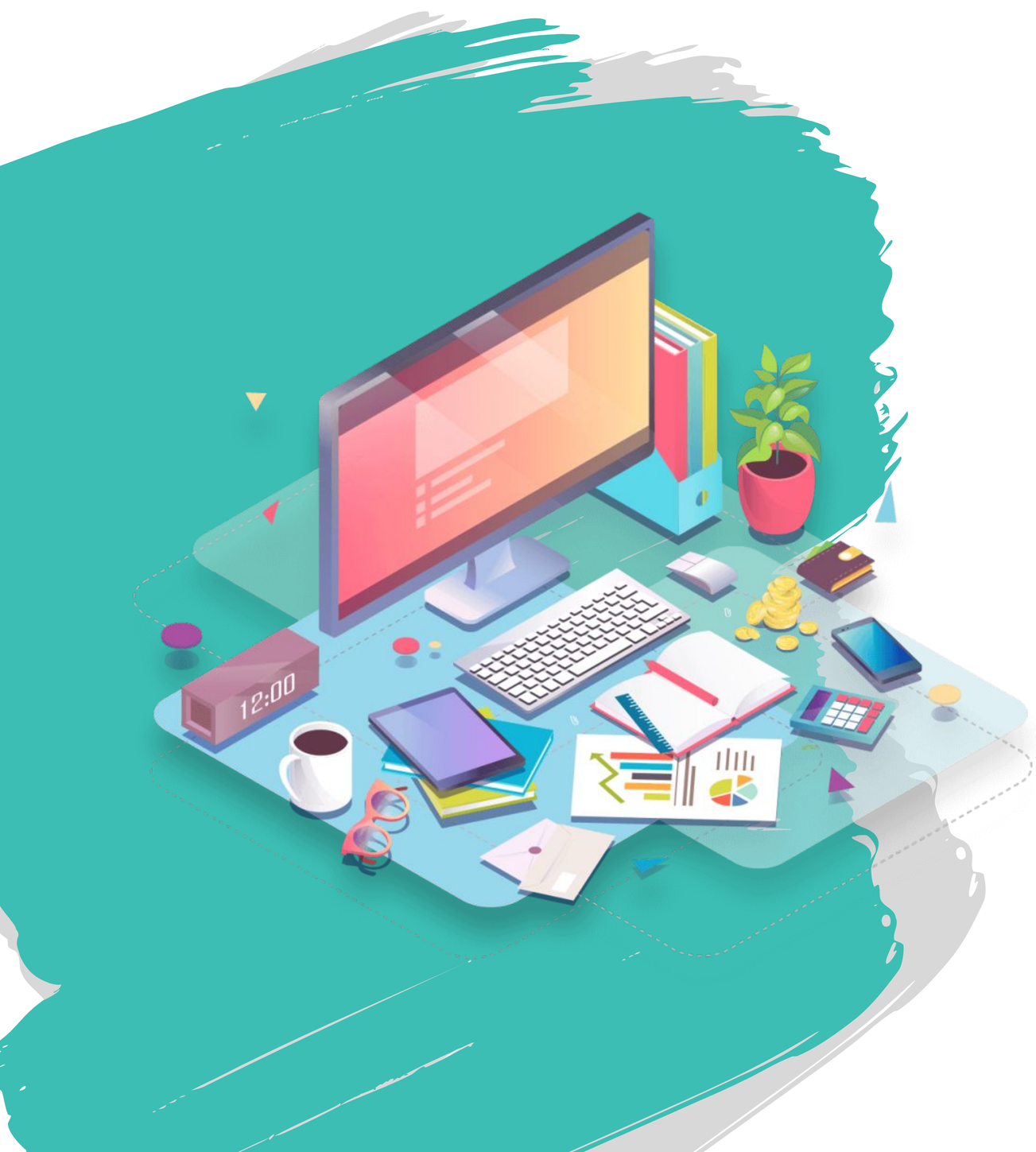
4

OUTPUT

```
Math
English
Chinese
Chemistry
Physics
```

EXPLANATION

This time, `subject_list[4]` will output `'Physics'` and now let's print it out!



Index-Based Looping on multiple lists

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
score_tuple = (97, 85, 89, 67, 75)

for index in range(5):
    print(subject_list[index], score_tuple[index])
```

OUTPUT

Math 97
English 85
Chinese 89
Chemistry 67
Physics 75

EXPLANATION

Now let’s put back our score_tuple and see how index-based looping works.

Let’s look at the first iteration!

Index-based looping (Iter #1)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
score_tuple = (97, 85, 89, 67, 75)
      0
for index in range(5):
    print(subject_list[0], score_tuple[0])
```

OUTPUT

Math 97

EXPLANATION

In the first iteration, the index takes on the first value in the range(5) iterator, i.e. 0.

Index-based looping (Iter #2)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
score_tuple = (97, 85, 89, 67, 75)
      1
for index in range(5):
    print(subject_list[1], score_tuple[1])
```

OUTPUT

Math 97
English 85

EXPLANATION

2nd iteration!

Index-based looping (Iter #3)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
score_tuple = (97, 85, 89, 67, 75)
      2
for index in range(5):
    print(subject_list[2], score_tuple[2])
```

OUTPUT

Math 97
English 85
Chinese 89

EXPLANATION

3rd iteration!

Index-based looping (Iter #4)

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
score_tuple = (97, 85, 89, 67, 75)
      3
for index in range(5):
    print(subject_list[3], score_tuple[3])
```

OUTPUT

Math 97
English 85
Chinese 89
Chemistry 67

EXPLANATION

4th iteration!

Made with love by Steven Chia

Index-based looping (Iter #5)

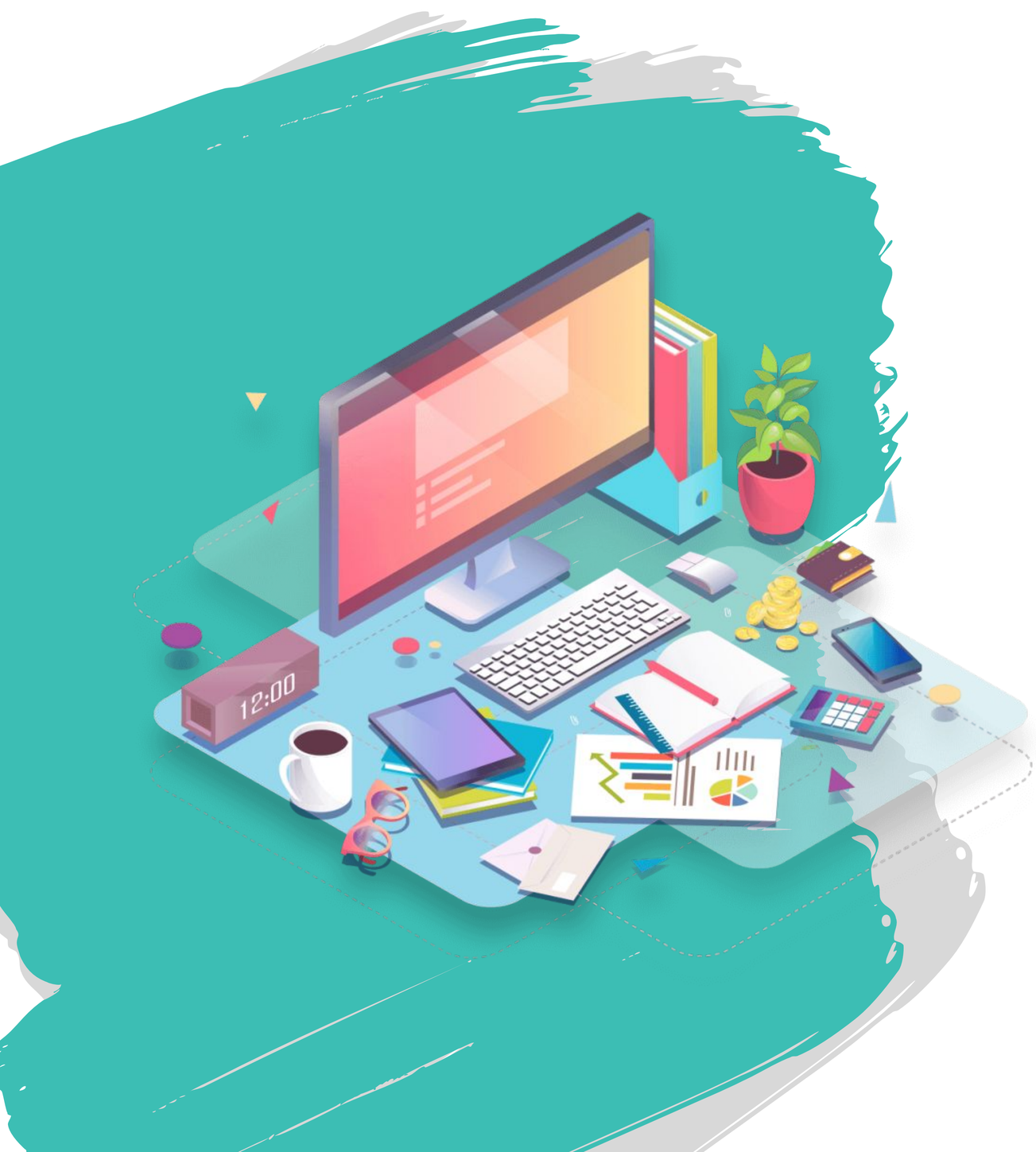
```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
score_tuple = (97, 85, 89, 67, 75)
      4
for index in range(5):
    print(subject_list[4], score_tuple[4])
```

OUTPUT

Math 97
English 85
Chinese 89
Chemistry 67
Physics 75

EXPLANATION

5th iteration!



In-Depth understanding of Index-Based Looping

Index-based looping

```
for index in range(len(subject_list)):
    print(subject_list[index], score_tuple[index])

for index in range(5):
    print(subject_list[index], score_tuple[index])
```

OUTPUT

Math 97
English 85
Chinese 89
Chemistry 67
Physics 75

EXPLANATION

Now, let's cover how `range(len(subject_list))` turned into `range(5)`

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

# range(len(subject_list))
```

OUTPUT

EXPLANATION

Let's focus on just the code block
`range(len(subject_list))`

We will keep the code block commented out at all times to keep it in view

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

# range(len(subject_list))

print(range(len(subject_list)))
```

OUTPUT

range(0, 5)

EXPLANATION

Let’s see what happens if we just print it out.

range(0,5) is equivalent to range(5) as we have seen earlier!

Let’s now further break this down!

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']

# range(len(subject_list))

print(len(subject_list))
```

OUTPUT

```
5
```

EXPLANATION

As can be seen, we know that there are 5 elements in `subject_list`. When we print out the length of the list, the output is 5!

Let's keep this in mind by putting a big 5 above the commented line

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
                                5
# range(len(subject_list))

print(len(subject_list))
```

OUTPUT

5

EXPLANATION

With this in place, Let's now look at what happens when we run a for loop over `range(len(subject_list))`

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
                        5
# range(len(subject_list))
                                    5
for index in range(len(subject_list)):
    print(index)
```

OUTPUT

```
0
1
2
3
4
```

EXPLANATION

The highlighted code are the same and can be replaced by 5!

So, when we print the index over a for loop over range(5), we see the output on the left!

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
                        5
# range(len(subject_list))

for index in range(5):
    print(index)
```

OUTPUT

```
0
1
2
3
4
```

EXPLANATION

This is an easier way to see it!

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry', 'Physics']
                        5
# range(len(subject_list))
                                    5
for index in range(len(subject_list)):
    print(subject_list[index])
```

OUTPUT

Math
English
Chinese
Chemistry
Physics

EXPLANATION

Let’s now look at the complete code again, but with the 5 still highlighted.

Now, let’s remove `'Physics'` from the list and see what happens!

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry']
                        4
# range(len(subject_list))
                                    4
for index in range(len(subject_list)):
    print(subject_list[index])
```

OUTPUT

Math
English
Chinese
Chemistry

EXPLANATION

The result of `len(subject_list)` is now 4!

To really anchor this concept, let's print the index together with the subjects.

Index-based looping

```
subject_list = ['Math', 'English', 'Chinese', 'Chemistry']

# range(len(subject_list))

for index in range(len(subject_list)):
    print(index, subject_list[index])
```

0 1 2 3 4 4

OUTPUT

```
0 Math
1 English
2 Chinese
3 Chemistry
```

EXPLANATION

We see that the number in front of the outputs correspond to the indexes of items in `subject_list`!

Comparison

```
# collection-based looping
for subject in subject_list:
    print(subject)
```

```
# index-based looping
for index in range(len(subject_list)):
    print(subject_list[index])
```

OUTPUT

Math
English
Chinese
Chemistry
Physics

EXPLANATION

As a rule of thumb, use collection-based looping. Only use index-based looping when you have to loop through multiple lists/tuples at the same time.

Let’s now understand how index-based looping works by simplifying the code!