for loops with range (class slides)

CSc 110 - for loops with range

`while` vs. `for` loops

In addition to `while`, we can use `for` to create loops

```
values = [70, 20, 30]
index = 0
while index < len(values):
  print(values[index])
  index += 1
```

```
70
20
30
```

```
values = [70, 20, 30]
for index in range(len(values)):
  print(values[index])
```

```
70
20
30
```

Write a function

1. Its name is `make_all_even`
2. It takes one argument, a list of `integers`
3. It iterates over the list, changing odd numbers to even number (even up)

4. Use a `for` loop

```
test_integers = [1, 2, 3, 4]
assert make_all_even(test_integers) == [2, 2, 4, 4]
```

Write a function – solution

```
def make_all_even(integers):
  # for each index in list
  for index in range(len(integers)):
    integers[index] += integers[index] % 2 # add zero if even, one if odd
  return integers

def main():
  test_integers = [1, 2, 3, 4]
  assert make_all_even(test_integers) == [2, 2, 4, 4]
  print(test_integers) # we print the list we created before function call

main()
```

```
[2, 2, 4, 4]
```

Write a function

1. Its name is `indices_of_vowels`
2. It takes a single `string` as its parameter.
3. It returns a list of integers that represent the indices of the vowels in the original list
4. Use a `for` loop

Test cases:

```
assert indices_of_vowels("hello") == [1, 4]
assert indices_of_vowels("") == []
assert indices_of_vowels("aeiou") == [0, 1, 2, 3, 4]
```

Write a function – solution

```
def indices_of_vowels(string):
  result = [] # initialize empty list to hold indices
  # for every index in list
  for index in range(len(string)):
    if string[index] in "aeiou": # check if character is vowel
      result.append(index) # append index to result
  return result

def main():
  assert indices_of_vowels("hello") == [1, 4]
  assert indices_of_vowels("") == []
  assert indices_of_vowels("aeiou") == [0, 1, 2, 3, 4]

main()
```

Submit code for attendance

Submit your `indices_of_vowels` functions to Gradescope for attendance.

Name your file `indices_of_vowels.py`

More on how to use `range()`

Syntax: `range(start, stop, step)`

- `start` Optional. An integer number specifying at which position to start. Default is 0
- `stop` Required. An integer number specifying at which position to stop (not included).
- `step` Optional. An integer number specifying the incrementation. Default is 1

More on how to use `range()`

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

```
0
1
2
3
4
```

```python
for n in range(0, 5, 1):
  print(n)
```

```
0
1
2
3
4
```

```python
for n in range(0, 5, 2):
  print(n)
```

```
0
2
4
```

Write a function

1. Its name is `every_two_together`
2. It takes one argument, a list of `characters`
3. It creates a new string with items at even indices in `characters` concatenated together
4. It returns the string created
5. Use a `for` loop

Test cases:

```python
characters = ["a", "e", "p", "o", "p", "w", "l", "i", "e", "f"]
assert every_two_together(characters) == "apple"
```

Write a function – solution

What other test cases should we run?

```
def every_two_together(chars):
  new_string = ""
  for i in range(0, len(chars), 2):
    new_string += chars[i]
  return new_string

def main():
  characters = ["a", "e", "p", "o", "p", "w", "l", "i", "e", "f"]
  assert every_two_together(characters) == "apple"

main()
```

Write a function

Test cases:

```
my_numbers = [0, 1, 0, 3, 0]
assert add_every_two(my_numbers) == 0
```

Write a function – solution

```
def add_every_two(numbers):
  result = 0
  for i in range(0, len(numbers), 2):
    result += numbers[i]
  return result

def main():
  my_numbers = [0, 1, 0, 3, 0]
  assert add_every_two(my_numbers) == 0

main()
```