Lecture 27 – Nested loops, passing mutable objects as parameters
At the end of this lecture, students should be able to:

- understand that the body of a loop can contain any types of statements including another loop
- show the output of code containing nested loops
- code trace functions which have mutable objects as parameters
def draw_shapes(a_canvas):
    number_of_rows = 5
    left_hand_side = 0
    y_down = 0
    size = 50
    first_is_circle = True

    for number_to_do in range(1, number_of_rows + 1):
        x_left = left_hand_side
        is_circle = first_is_circle
        for j in range(number_to_do):
            rect = (x_left + 3, y_down + 3, x_left + size - 3, y_down + size - 3)
            if is_circle:
                a_canvas.create_oval(rect, fill="blue")
            else:
                a_canvas.create_rectangle(rect)
        x_left = x_left + size * 2
        is_circle = not is_circle

        y_down = y_down + size
        first_is_circle = not first_is_circle

Recap from Lecture 26
Nested loops

The body of a `for ... in` loop can include any code structures (ifs, `if ... else`, `if ... elif`, assignment statements) and they can include other `for ... in` loops or `while` loops. These are called **nested loops**.

```python
for num1 in range(5):
    print("A")
    for num2 in range(3):
        print("B")
        print("C")
    print("D")
```

In total,
- how many times is "A" printed
- how many times is "B" printed
- how many times is "C" printed
- how many times is "D" printed
Nested loops – example 1

How many times is the word "hello" printed?

```python
1 def main():
2     for i in range(3):
3         for j in range(4):
4             print("hello")
5     main()
```
Nested loops – example 2

How many lines of output are printed?

def main():
    for i in range(3):
        for j in range(4):
            print("hello", end = " ")
        print()
main()
def main():
    number = 0
    for i in range(3):
        number = number + 1
        for j in range(4):
            print(number, end = " ")
    print()
def main():
    number = 0
    for num1 in range(3):
        print(number, end = " ")
        for num2 in range(4):
            number = number + 1
            print()
    print(number)

main()
def main():
    for i in range(2, 4):
        for j in range(3):
            print(i + j, end=" ")
    print()

main()
def main():
    list1 = [5, 4, 3, 2]
    list2 = [3, 4]
    list3 = []
    for num1 in list1:
        for num2 in list2:
            list3.append(num1 + num2)
    print(list3)

main()
Nested loops – exercise

The `get_list_of_vowel_count()` function returns a list of the number of vowels in each word of the parameter list.

```python
def get_list_of_vowel_count(word_list):
    vowels = "aeiouAEIOU"
```

def main():
    name_list = ["Mirabelle", "John", "Kelsey", ...]
    vowel_counts = get_list_of_vowel_count(name_list)
    print(vowel_counts)
```

main()
```

[4, 1, 2, 3, 4, 3, 4, 3, 1, 2, 3]
Nested loops – exercise

Give the output.

def main():
    for first in range(2, 5):
        for second in range(1, first):
            print(first + second, end=" ")
    print()

main()
def main():
    total = 0
    for first in range(1, 5):
        total = total + first

        for second in range(1, first):
            total = total + second

    print("Grand total: ", total)

main()
Nested loops - print_dotted_names()

The print_dotted_names() function prints the list of all the names in the parameter list after changing any of the letters of the name which are in the letters_to_dot parameter string to a dot.

def main():
    names_list = ["Kelsey", "Isobel", "Alistair", "Emmie", "Ophelia"]

    letters_to_dot = 'aeoutsAEOUTS'
    print(names_list)
    print_dotted_list(names_list, letters_to_dot)

def print_dotted_list(names_list, letters_to_dot):

main()

[['Kelsey', 'Isobel', 'Alistair', 'Emmie', 'Ophelia']
[['K.l..y', 'I..b.l', '.li...ir', '.mmi.', '.ph.li.']]
def main():
    a_list1 = [10, 9]
    a_list2 = [1, 3, 4]
    function_15(a_list1, a_list2)
    print("a_list2:", a_list2)

def function_15(list1, list2):
    list3 = list2
    list3.append(list1[1])
    list2.append(list1[0])
    print("  list3:", list3)

main()
def main():
    a_list1 = [10, 9]
    a_list2 = [1, 3, 4]
    a_list1 = function_16(a_list1, a_list2)
    print("a_list1:", a_list1)
    print("a_list2:", a_list2)

def function_16(list1, list2):
    list3 = []
    list3.append(list1[1])
    list3.append(list1[0])
    list2 = list3
    list2.append(list3[0])
    print("  list2:", list2)
    return list3

main()
def main():
    a_list1 = [4, 3]
    a_list2 = [1, 3, 4]
    function_17(a_list1, a_list2)

    print("a_list1:", a_list1)
    print("a_list2:", a_list2)

def function_17(list1, list2):
    list3 = list2
    for i in range(2):
        list3.append(list1[i])
        list2.append(list1[i])

    list1 = list3
    print("  list3:", list3)

main()
def main():
    a_list1 = [4, 3]
    a_list2 = [1, 3, 4, 5, 2]
    a_list2 = function_18(a_list1, a_list2)

    print("a_list1:", a_list1)
    print("a_list2:", a_list2)

def function_18(list1, list2):
    list3 = []
    for element in list2:
        if not element in list1:
            list1.append(element)
        else:
            list3.append(element)
    return list3

main()
The body of loops can contain any kind of statements including other loops.

Passing parameters which are mutable objects to functions means that the function code may change the object's data.
Python features used in this lecture

```python
def print_dots(dot_list):
    for num1 in dot_list:
        for num in range(num1):
            print(".", end = ")
        print()

for first in range(2, 5):
    for second in range(1, first):
        print("(", first, ", ", second, ")", sep = ", end = ")
    print()
```

20