MTA 98-381 LESSON 4 LOOPS

LOOPS IN PYTHON

- Loop structures allows us to repeat the same statements many times.
- Python supports 2 types of loops: `for` and `while`.
- `for` loop is typically used when we <u>know how many times </u>to repeat.
- `while` loop is typically used when we <u>do not know</u> how many times to repeat.

FOR LOOP

- `for` loop is typically used when we <u>know how many times</u> a to repeat.
- Syntax:

```
      for counter in range([start=0,] stop[, step=1]):
            statement(s)
```

  - `start` is an <u>optional</u> value. If omitted 0 will be used.
  - `stop` is a <u>compulsory</u> value. Note when the `stop` value is reached, the statements in loop <u>won't be executed</u>.
  - `step` is an <u>optional</u> value. If omitted 1 will be used.

FOR LOOP (CONT.)

- Example 1:

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

  - Output: `0 1 2 3 4`
  - Note that 5 is not printed.

- Example 2:

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

  - Output: `0 1 2 3 4`
  - `start` is 0 (default)
  - `step` is 1 (default)

F0R LOOP (CONT.)

- Example 3:

```
for i in range(2, 6):
    print (i)
```

  - Output: 2 3 4 5
  - step is 1 (default)

- Example 4:

```
for i in range(1, 9, 2):
    print (i)
```

  - Output: 1 3 5 7

FOR LOOP (CONT.)

- Example 5:

```
for i in range(6, 2):
    print (i)
```

  - No output because `stop` is smaller than `start` for a positive `step`.

- Example 6:

```
for i in range(1, 9, -1):
    print (i)
```

  - No output because `stop` is larger than `start` for a negative `step`.

EXERCISE 1: FOR LOOP

- 1: Write a for loop to print 0 1 2 3 4 5 6.
- 2: Write a for loop to print 0 2 4 6 8.
- 3: Write a for loop to print 1 4 7 10.
- 4: Write a for loop to print 11 8 5 2.
- 5: Write a for loop to print 2 1 0 -1 -2.
- 6: Write a for loop to print 0 -1 -2 -3.
- 7: Write a for loop to print 0 -4 -8 -12.

FOR LOOP (CONT.)

- Loop is widely used to perform summation.
- Example: Collect donations

```
total_donor = int(input('Enter total donor: '))
total_amount = 0
for i in range(total_donor):
    amount = float(input('Enter donation: RM '))
    total_amount += amount
print ('Total donor :', total_donor)
print ('Total amount: RM', total_amount)
```

EXERCISE 2: F0R LOOP

- Write a program that gets user inputs for exam marks, then calculate the following:
    1. Total number of marks
    2. Total number of passes and failures
    3. Highest mark and lowest mark
    4. Average mark

WHILE LOOP

- `while` loop is typically used when we <u>do not know</u> how many times to repeat.
- Syntax:

```
while condition:
    statement(s)
```

WHILE LOOP (CONT.)

- It is possible to use `while` loop when we know how many times to repeat, but `for` loop provides a more compact syntax.
- Example 1 (emulates `for` loop Example 1):

```
i = 0
while i < 5:
    print (i)
    i += 1
```

  - Output: `0 1 2 3 4`

WHILE LOOP (CONT.)

- Example 2 (emulates `for` loop Example 2):

```
i = 1
while i < 9:
    print (i)
    i += 2
```

  - Output: 1 3 5 7

WHILE LOOP (CONT.)

- `while` loop is best when we do not know how many times to iterate.
- Example 1 (guess an integer):

```
import random
random_int = random.randint(1, 10)
print (random_int)
guess = int(input('Guess a number (1 to 10): '))
while (guess != random_int):
  print ('Incorrect, try again.')
  guess = int(input('Guess a number (1 to 10): '))
print ('You got it!')
```

EXERCISE 3: WHILE LOOP

- Write a Python program that gets user input for an age, then output Minor, Adult, or Senior according to the age. Print an error message when negative age is entered. Repeat the user input until a non-negative age is entered.

EXERCISE 4: WHILE LOOP

- Write a Python program to collect donation from public. A person may donate any amount. We do not know how many person will donate. Zero should be entered when it is time to close the donation. Total donation should be output after a donation is made.

BREAKING A LOOP

- A `break` statement allows us to break/stop a loop when a condition is met.
- Example:

```
for i in range(10):
    print (i)
    if i == 4:
        break
```

  - Output: `0 1 2 3 4`
  - The loop stops when the value of i is 4.

EXERCISE 5: BREAK A LOOP

- Buy 5 items unless the total exceeds RM100.
- Sample output:
  - `Enter item 1 price: RM`<u>`50`</u>
  - `Enter item 2 price: RM`<u>`10`</u>
  - `Enter item 3 price: RM`<u>`20`</u>
  - `Enter item 4 price: RM`<u>`40`</u>
  - `4 items, total RM120.00 exceeds RM100.`

CONTINUE TO THE NEXT ITERATION

- A `continue` statement allows us to jump to the next iteration when a condition is met.
- Syntax:

```
for i in range(10):
    if i % 3 == 0:
        continue
    print (i)
```

  - Output: 1 2 4 5 7 8

EXERCISE 6: CONTINUE TO THE NEXT ITERATION

- Sum all integers between 3 to 10 that are not divisible by 4.
- 3 to 10 : 3+5+6+7+9+10 = 40
- 2 to 9 : 2+3+5+6+7+9 = 32

EXERCISE 7

- Write a loop that prints a multiplication table for 1 to n where n is an integer provided by user.
- If n is 3 then prints as follows:

```
      |    1    2    3
 -----------------
     1|    1    2    3
     2|    2    4    6
     3|    3    6    9
```