Algorithms and Repetition

An example needing repetition

Try the Exchange Table example: src/exchange_table.py

```
uk_amount = float(input("\nEnter a starting amount in UK sterling: "))
print("\n\tPOUNDS\t\tEUROS")
print("\t{:.2f}\t\t{:.2f}".format(uk_amount,
                                  uk_amount * EXCHANGE_RATE))
uk_amount = uk_amount + 10
print("\t{:.2f}\t\t{:.2f}".format(uk_amount,
                                  uk_amount * EXCHANGE_RATE))
uk_amount = uk_amount + 10
print("\t{:.2f}\t\t{:.2f}".format(uk_amount,
                                  uk_amount * EXCHANGE_RATE))
```

The `while` loop

- The general form of a `while` loop is:

```
while boolean expression:
   python statement(s)
   ....
```

- Python executes the loop as follows:

  1. It checks if *boolean expression* is `True` or `False`.
  2. If it is `False` it finishes with the while loop.
  3. If it is `True` it executes the *Python statement(s)*, then goes back to step 1.

- It is the programmer's responsibility to make sure that there is something in the Python statement(s) that will eventually make the boolean expression false.

Example with a `while` loop

Try this version of the Exchange Table program which uses a `while` loop: src/exchange_table_with_while.py

```
uk_amount = float(input("\nEnter a starting amount in
                         UK sterling: "))
number_of_lines = int(input("\nEnter the number of
                             lines for the table: "))

print("\n\tPOUNDS\t\tEUROS")

line_counter = 1

while line_counter <= number_of_lines:
    print("\t{:.2f}\t\t{:.2f}".format(uk_amount,
                                     uk_amount * EXCHANGE_RATE))
    uk_amount = uk_amount + 10
    line_counter += 1
```

Example with while and for loops

Try this version of the Exchange Table program which uses both a `while` loop and a `for` loop:
src/exchange_table_with_while_and_for.py

```
EXCHANGE_RATE = 1.564

again = 'y'

while again == 'y':
    uk_amount = float(input("\nEnter a starting amount in
                             UK sterling: "))
    number_of_lines = int(input("\nEnter the number of
                                 lines for the table: "))

    print("\n\tPOUNDS\t\tEUROS")

    for line_counter in range(number_of_lines):
        print("\t{:.2f}\t\t{:.2f}".format(uk_amount,
                                         uk_amount * EXCHANGE_RATE))
        uk_amount = uk_amount + 10
```

Checking for correct input

A `while` loop can be used to insist on sensible input from the user. Look at retire_with_input_check.py.

Try entering:

```
231
-8
42
```

Checking for correct input (cont.)

```
users_age = int(input("\nEnter your age: "))

while users_age < 0 or users_age > MAX_AGE:
    print("\tDon't be stupid! Enter your real age: ")
    users_age = int(input("\nEnter your age: "))

if users_age < RETIREMENT_AGE:
    print("You will retire in ", end="")
    print(RETIREMENT_AGE - users_age," years.\n")
elif users_age == RETIREMENT_AGE:
    print("You will retire this year.")
elif users_age > RETIREMENT_AGE:
    print("You retired ", end="")
    if users_age - RETIREMENT_AGE == 1:
        print(users_age - RETIREMENT_AGE," year ago.\n")
    else:
        print(users_age - RETIREMENT_AGE," years ago.\n")
```