Times Tables Trainer

This week's project introduces three new ideas. Firstly, it requires the user to type something in, in response to a question. Secondly, it uses a *list* to group a set of numbers together. Thirdly, it uses a *timer* to measure how long the user took to respond. We will re-use some ideas we've seen before; in particular, *random numbers* and some simple *loops*.

Here's the problem. One of my friends is a bit slow with his times-tables, and I want to write a program that will help him improve. I want this program to do four things:

- I want to be able to control which tables it tests him on
- I want it to show him which answers are right and which are wrong
- I want to encourage him to get faster, by timing him

As usual, we will build up slowly to our final program. Let's get started.

As usual, start by opening the python shell (IDLE) and use `file->save as`. Navigate to the code club drive and set the name of your program. Mine is called neal_tables. Use your own name, so that everyone's program has a different name.

Type in this piece of code.

```
# Times table trainer
#

from random import choice

first_set = [1,2,3,4,5,6,7,8,9,10,11,12]
second_set = [1,2,3,4,5,6,7,8,9,10,11,12]

first = choice(first_set)
second = choice(second_set)

answer = input("What is " + str(first) + " x " + str(second) + " = ? ")
if int(answer) == first * second:
   print ("Correct")
else:
   print ("Wrong")
```

Now click `run->run module` (or press F5). When the program runs it should print out a question (for example: What is 12 x 10 = ?) and wait for you to type in the answer and press ENTER. If you get an error when you try to run it, compare the code above carefully with what you typed. Correct any mistakes and try again.

The program uses two lists, named `first_set` and `second_set`. Each of them is a list of numbers between 1 and 12. When the program asks a question like "What is 5 x 12" it is going to pick a number from the first list (in this case, it picks 5) and a number from the second list (in this case, it picks 12). Each part of the list is called an element and the elements are separated by commas and enclosed in square brackets.

The program needs to pick the two number for the question at random (otherwise it

would ask the same question over and over again). That is what `choice()` does: it picks one element at random from the list. So, `first` ends up with one element chosen at random from `first_set` and `second` ends up with one element chosen at random from `second_set`.

Next, the program prints the question. Previously we've used `print` to display information. This time we use `input`. This allows the program to print a message, but also waits for the user to type in a response. That response goes into the variable `answer`.

The final fiddly thing is that we have to tell the computer which things are numbers and which things are strings. Remember in an earlier project, where we learned that 4 + 9 is 13 (adding numbers), but that "4" + "9" is "49" (adding strings)? The program uses `str()` when it needs to turn a number into a string and `int()` when it needs to turn a string into a number.

The final fragment of code is *comparing* the value typed in by the user with the value that the program calculates to be the correct answer. You will see a strange thing there: two equal signs (==).

```
myanswer = 6 * 2      # change variable myanswer to the value 12
myanswer == 6 * 2     # compare variable myanswer to the value 12 - does not
                      # change myanswer. The result of a comparison is either
                      # true (they are the same) or false (they are not)
```

If you are unclear about any of this, ask for help.

So far, our program only asks one question. Not much of a test. To ask a series of questions, we just have to use a loop. The only things we need inside the loop are the things that change for each question. Therefore, we need `first` and `second` *inside* the loop (so that we pick a random number for each question) but we can leave first_set and second_set *outside* the loop, because they stay fixed for all the questions.

Modify your program to look like the one below. You don't need to type it all in; just the parts that have changed. The new code is shown in red. You also have to change the indentation (use the TAB key to the left of the Q key) for the parts shown in green.

When you type the code in, your code will not be coloured like this!

```
# Times table trainer
#

from random import choice

first_set = [1,2,3,4,5,6,7,8,9,10,11,12]
second_set = [1,2,3,4,5,6,7,8,9,10,11,12]
questions = 10

for q in range(0, questions):
    first = choice(first_set)
    second = choice(second_set)

    answer = input("What is " + str(first) + " x " + str(second) + " = ? ")
    if int(answer) == first * second:
       print ("Correct")
    else:
       print ("Wrong")
```

As before, click `run->run module` (or press F5). When the program runs it should do just what it did before, but do it 10 times. As before, if you get an error when you try to run it, compare the code above carefully with what you typed. Correct any mistakes and try again.

The change for this second version of the program is very simple. The program has a new variable, `questions`, to control how many questions to ask. Each time the `for` loop runs, it makes the value of variable `q` bigger by 1. The `for` loop stops when `q` tries to go to 10 (the value of `questions`). That means that q has the values 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 in turn (but not 10! The loop has stopped by then).

Tip: while you are writing and testing the program, you might want to change "`questions = 10`" to "`questions = 3`".

Now that the program is asking more questions, we want it to keep track of how many the user gets right and how many the user gets wrong. The program only needs to count one of these two things (either the right answers or the wrong answers).

Modify your program to look like the one below. You don't need to type it all in; just the parts that have changed. The new code is shown in red.

When you type the code in, your code will not be coloured like this!

```
# Times table trainer
#

from random import choice

first_set = [1,2,3,4,5,6,7,8,9,10,11,12]
second_set = [1,2,3,4,5,6,7,8,9,10,11,12]
questions = 10
correct = 0

for q in range(0, questions):
    first = choice(first_set)
    second = choice(second_set)

    answer = input("What is " + str(first) + " x " + str(second) + " = ? ")
    if int(answer) == first * second:
       print ("Correct")
       correct = correct + 1
    else:
       print ("Wrong")

print("")
print("Thanks for playing. You scored " + str(correct) + " out of " +
str(questions))
if correct/questions > 0.9:
   print("Well done")
else:
   print("You need to practise a bit more!")
```

As before, click `run->run module` (or press F5). When the program runs it should do just what it did before, but at the end it should tell you how well you did. As before, if you get an error when you try to run it, compare the code above carefully with what you typed. Correct any mistakes and try again.

The change for this third version of the program is to add a new variable, `correct`, to count the correct answers. At the end the program prints out the user's score. Then, the program uses another *comparison*. This time, the comparison is looking at the *fraction* or *proportion* of correct answers, and deciding how well the user has performed.

If you are unclear about any of this, ask for help.

The final change that we are going to make to the program is to add a timer so that we can see how long the user takes to answer the questions. This acts as a great way for the user to know that they are improving.

Modify your program to look like the one below. You don't need to type it all in; just the parts that have changed. The new code is shown in red.

When you type the code in, your code will not be coloured like this!

```
# Times table trainer
#

from random import choice
import time

first_set = [1,2,3,4,5,6,7,8,9,10,11,12]
second_set = [1,2,3,4,5,6,7,8,9,10,11,12]
questions = 10
correct = 0
start_time = time.time()

for q in range(0, questions):
    first = choice(first_set)
    second = choice(second_set)

    answer = input("What is " + str(first) + " x " + str(second) + " = ? ")
    if int(answer) == first * second:
       print ("Correct")
       correct = correct + 1
    else:
       print ("Wrong")

total_time = time.time() - start_time
print("")
print("Thanks for playing. You scored " + str(correct) + " out of " +
str(questions))
print("You took an average of " +  str(total_time/questions) +  " second to
answer a question")
if correct/questions > 0.9:
   print("Well done")
else:
   print("You need to practise a bit more!")
```

As before, click `run->run module` (or press F5). When the program runs it should do just what it did before, but at the end it should tell you how well you did. As before, if you get an error when you try to run it, compare the code above carefully with what you typed. Correct any mistakes and try again.

In this fourth version of the program we add two new variables, `start_time` and `total_time`. The code "`time.time()`" tells us the current time, and we use it twice: once to find out the current time at the moment when the program starts and once to find out the current time at the moment when the program is about to stop. By subtracting one from the other we can see how long the program ran for. If the program asked 3 questions, we can divide the time by 3 to get the average time that it took the user to answer a question. Of course, this is a bit of a cheat, because we are timing the whole of the program and ignoring the time that the computer took to do things. We are working on the assumption that the computer is much, much faster than the user.

If you are unclear about any of this, ask for help. Otherwise, read on for some extras...

Extras

The program we have set up tests all the tables, from 1 to 12. Suppose we only wanted to test the 7-times table, and suppose we didn't want to include any easy stuff like 1-times or 10-times table. Thanks to our lists, this change is really simple; just change what is in `first_set` and `second_set`:

```
first_set = [7]
second_set = [2,3,4,5,6,7,8,9,11,12]
```

You can see from this that it's fine for a list to have just one element in it (in this case, first_set only has a single element: the value 7). When you have made that modification and tried it out, look at this change:

```
first_set = [2,3,4,5,6,7,8,9,11,12]
second_set = [7]
```

Before you try it out, think about how it is different and what will happen when you try it. Now try it out ans see if you worked out the difference correctly.

Finally, my friend is going on holiday and will not have a computer with him, but he wants to keep up his times-table practise. Make a version of the program that simply prints out a set of questions on the screen. The idea is that I can print these out for him. To make my life easier I want the correct answer printed over on the right-hand side (I will tear that edge off the paper before I give him the printout, and use it to help me mark his work more quickly). Try and write the program without looking at my version (save it with a different name. I called mine `neal_holiday_tables`)

```
# Times table trainer
#

from random import choice

first_set = [7]
second_set = [2,3,4,5,6,7,8,9,11,12]
questions = 20

for q in range(0, questions):
    first = choice(first_set)
    second = choice(second_set)

    print ("What is " + str(first) + " x " + str(second) + " = ?
" + str(first*second))
```

