The Reaction Timer

Contents

The Reaction Timer

Using some simple Python we can build a reaction timer. Then we’ll use it to see who has the fastest reactions in the class!

Learning objectives

Students learn:

- To plan a program by analysing a problem and using pseudo code,
- Commenting code is useful as a development aid,
- To use the `random` library.

Resources

- 8 copies of the Program cards,
- 15 copies of the cheat sheet,
- Pens and paper.

What do we need for a reaction timer? : 20 minutes

Discuss with the class how the software will work. We'll need to print something on the screen and then time how long it takes for the subject to react to it.

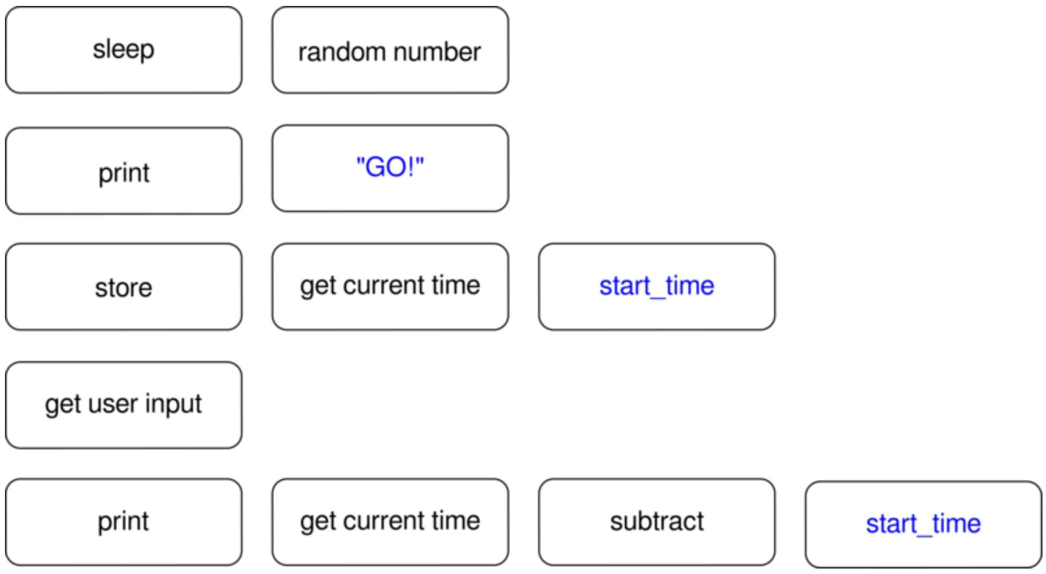
Pseudo code

Tell the students to get into groups of 4 and use the program cards to lay out a program sequence (from top to bottom). They'll need to use the blank ones to add their own variable names or extra codes.

After a few minutes ask 2 members of each team to walk around and get ideas from the other teams. They can ask questions, and the remaining members answer them.

Example code

My variable names are written in blue.



If you're feeling shaky with Python, then you can ask the whole class to agree to follow one team's program structure. This will make it easier to debug code later.

Convert pseudo code to Python comments

Ask students to get back in pairs, open a new program with Idle and save it as `reaction.py`.

Ask students to copy the code structure into their program, starting each line with a # so it becomes a comment. The lines will change colour to remind them it's a comment.

It should end up something like this:

```
#pick a random number and store in sleep_time

#sleep for sleep_time seconds

#print "GO!"

#store the current time in start_time

#wait for subject to press keyboard
```

```
#work out how long they took by subtracting the start_time now from the time

#print out how long they took
```

Build reaction timer software : 25 minutes

Students work in pairs to convert the comments into Python code. They can reference any of the programs they've saved so far. They should write the Python below the comment, and leave the comment in for reference.

Ask them to start at the top and to test each line of code at a time by saving and running the code.

Reference code

```
#import the libraries
import time
import random

#pick a random number and store in sleep_time
sleep_time = random.randint(3,10)

#sleep for sleep_time seconds
time.sleep(sleep_time)

#print "GO!"
print("GO!")

#store the current time in start_time
start_time = time.time()

#wait for the subject to press the 'enter key'
raw_input()

#work out how long they took by subtracting the start_time now from the time
reaction_time = time.time() - start_time
```

```

#print out how long they took
print(reaction_time)

```

Extension activity

We make the program better by printing better instructions and nicely formatting the output. Perhaps even adding a conditional to print out a congratulation or insult.

We don't get a great representation of reaction time without making multiple measurements (just like in science). Find a way to repeat the test 3 times and make an average before printing a final score.

Who's the fastest? : 10 minutes

Now it's time to run the competition! To save time, we'll play a 'sudden death' tournament. If you lose then you get knocked out.

Ask all pairs to test each other. The winners then play against each other. A class of 30 should only need 5 tests to find the overall winner.

You can ask at the end how to make the competition fairer, as the ways that the students come up with will probably be fairly straight forward to add to our program. For example taking 3 scores and making an average gets rid of the bad luck element.

Homework

todo

Outcome

All students:

- Have helped to write a program using the program cards,

- Have helped to start converting the program cards into Python.

Most students:

- Understand that analysis of the problem first makes it easier to turn into code later.
- Have finished the reaction timer.

Some students:

- Have added averaging to the reaction times.