COMPUTER SCIENCE CONTROLLED ASSESSMENT

□ MARK CLARKSON

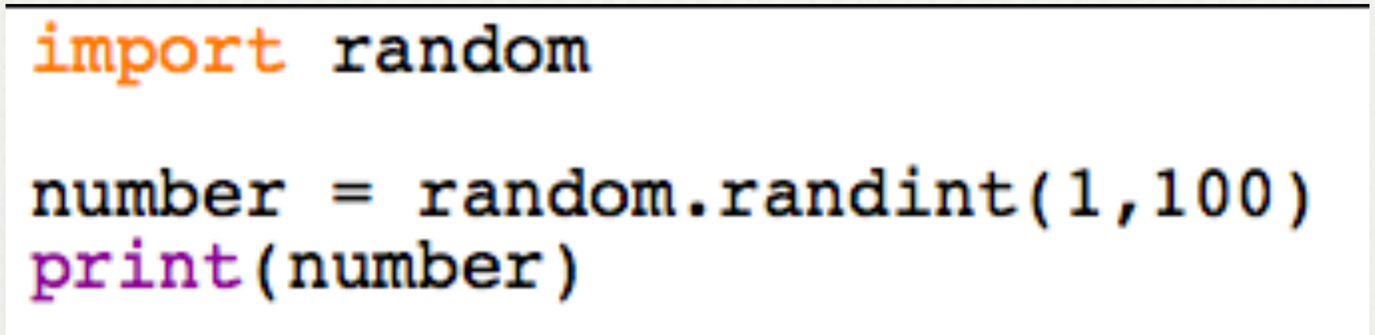
□ PI.MWCLARKSON.CO.UK

□ @MWCLARKSON

□ MWCLARKSON@GMAIL.COM

STARTER ACTIVITY

- □ WRITE AND TEST THE FOLLOWING CODE:



```
import random

number = random.randint(1,100)
print(number)
```

- □ ALTER THE PROGRAM TO:
  - □ PRINT OUT A NUMBER BETWEEN 1 AND 6
  - □ IF THE USER SCORES 5 OR 6 PRINT "YOU WIN"
  - □ OTHERWISE PRINT "YOU LOSE"

A QUICK COMPARISON

OCR	AQA	Edexcel	WJEC
1 Programming 1 Investigation	2 Programming	1 Programming	1 Programming
Total 40 hours 60% of grade	Total 50 hours 60% of grade	Total 15 hours 25% of grade	Total 15 hours 30% of grade
1 written exam	1 written exam	1 written exam	1 written exam 1 on-screen exam

IN CONTROL?

TASK SETTING

HIGH CONTROL

TASK TAKING

MEDIUM CONTROL

TASK MARKING

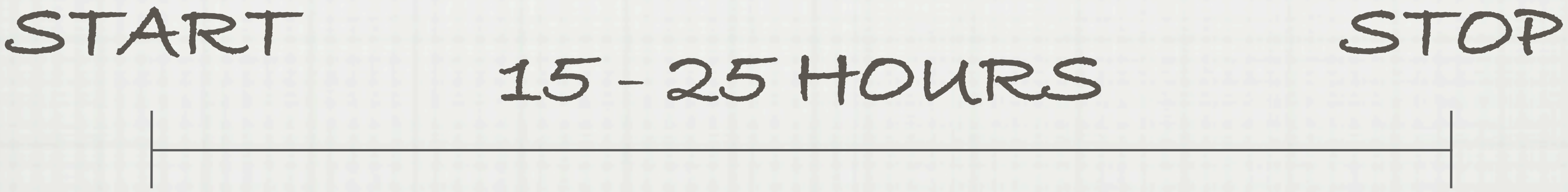
MEDIUM CONTROL

HTTP://WWW.FLICKR.COM/PHOTOS/63853971@N00/6017123783/

STRATEGIES FOR DELIVERY

STRATEGY ONE

INPUT - PROCESS - OUTPUT



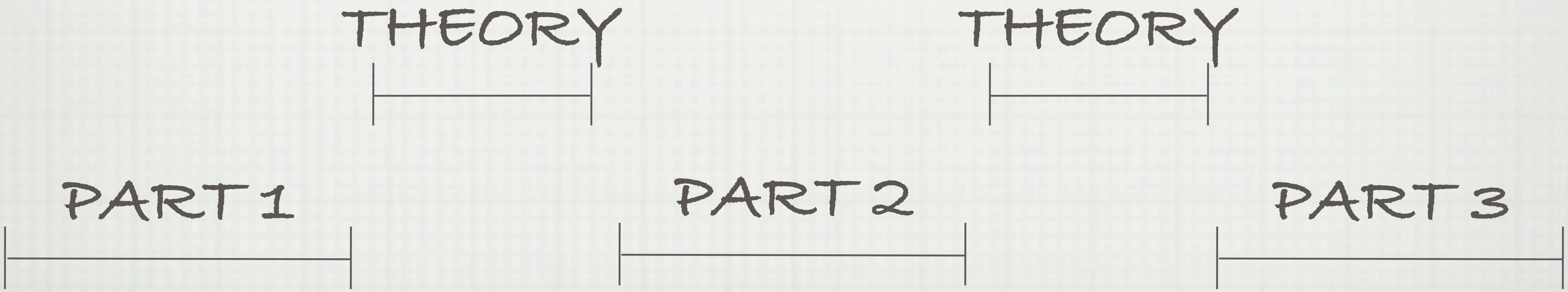
START

15 - 25 HOURS

STOP

STRATEGY TWO

ONE PIECE AT A TIME



THEORY

THEORY

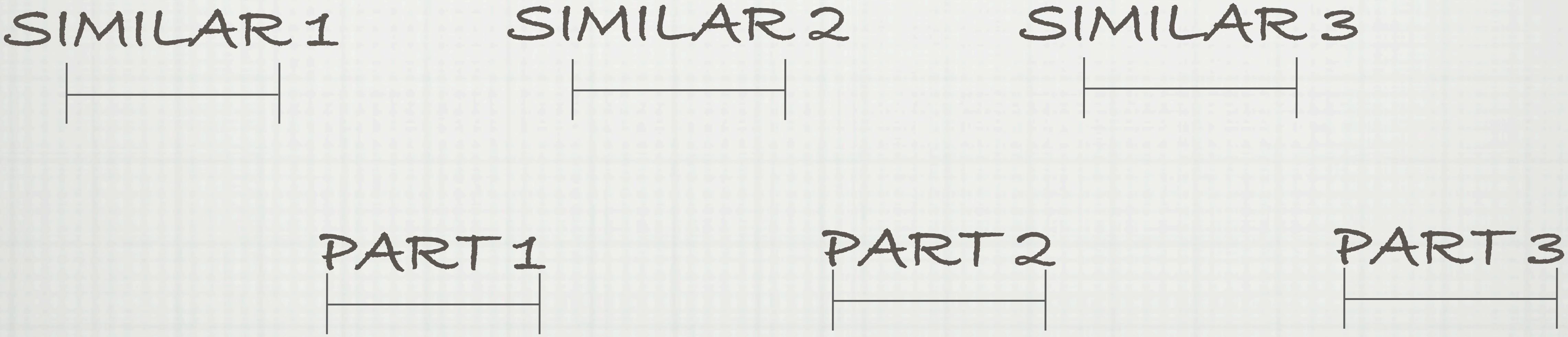
PART 1

PART 2

PART 3

STRATEGY THREE

SIMILAR BUT DIFFERENT



SIMILAR 1

SIMILAR 2

SIMILAR 3

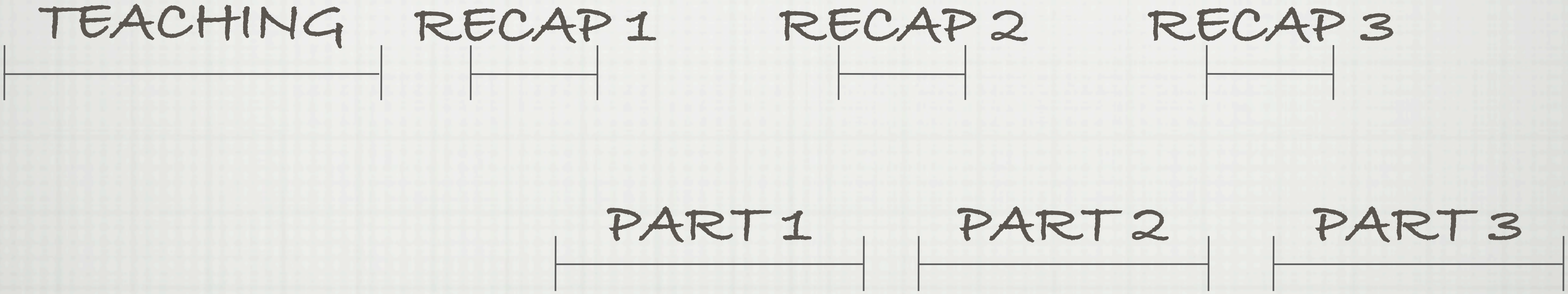
PART 1

PART 2

PART 3

STRATEGY FOUR

TIMELY REMINDER



TEACHING

RECAP 1

RECAP 2

RECAP 3

PART 1

PART 2

PART 3

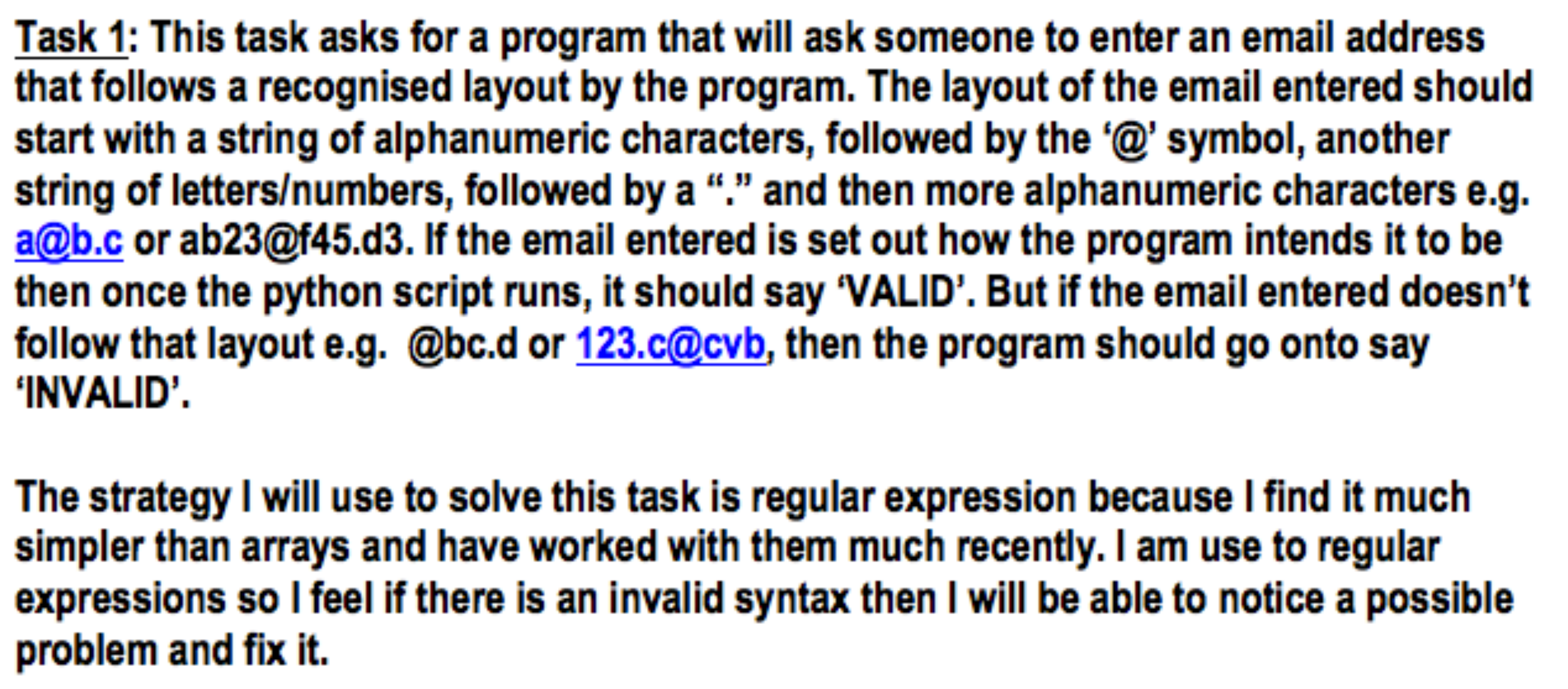
WALK THE LINE

- ☑ INTERNET ACCESS
- ☑ SOME GROUP WORK
- ☑ NOTES
- ☑ GENERIC FEEDBACK
- ☑ STUDENTS' OWN WORK
- ☑ SUPERVISED DURING WORK

HTTP://WWW.FLICKR.COM/PHOTOS/94322200@N08/8585885085/

YOUR TURN!

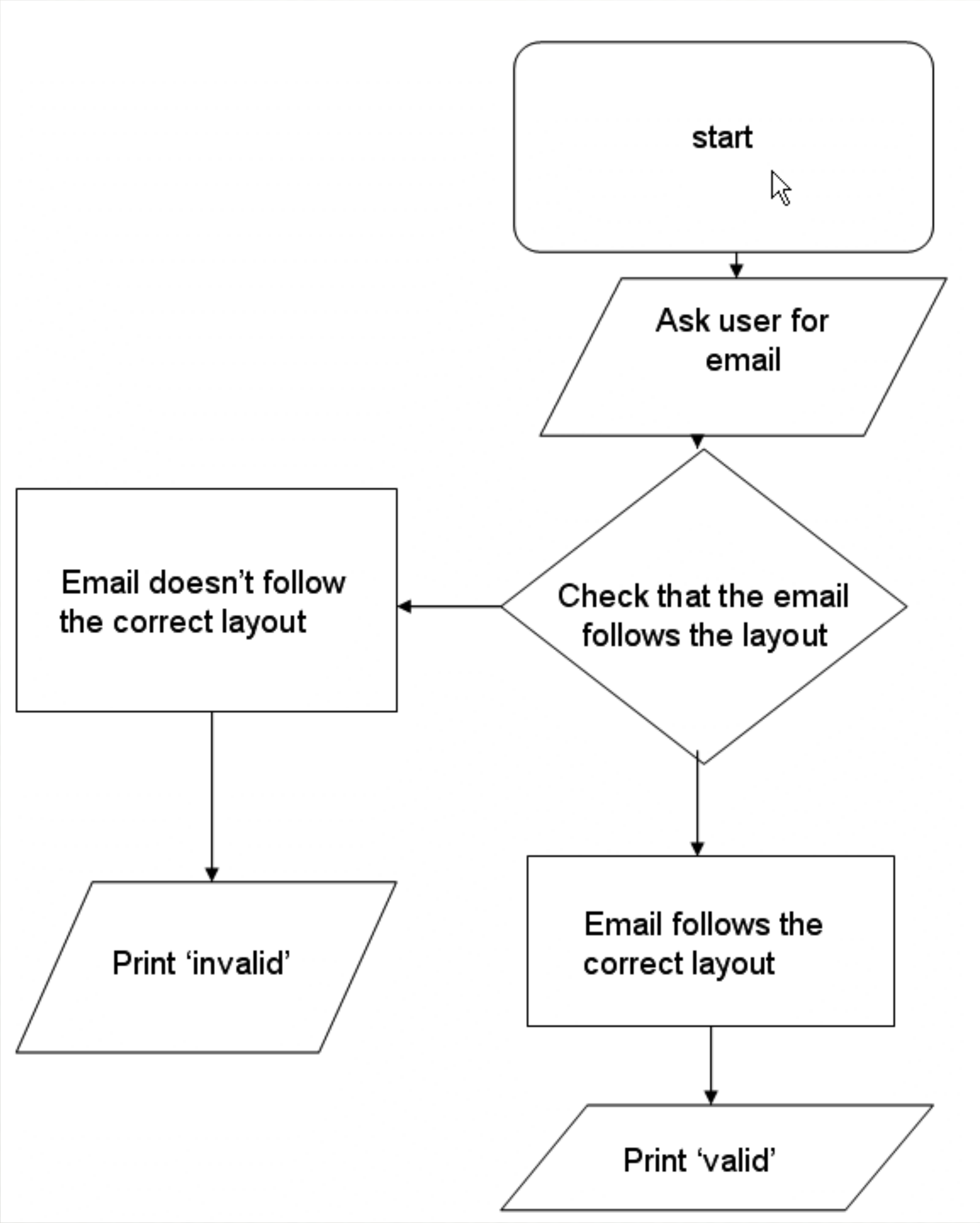
STEP 1 - STOP & THINK

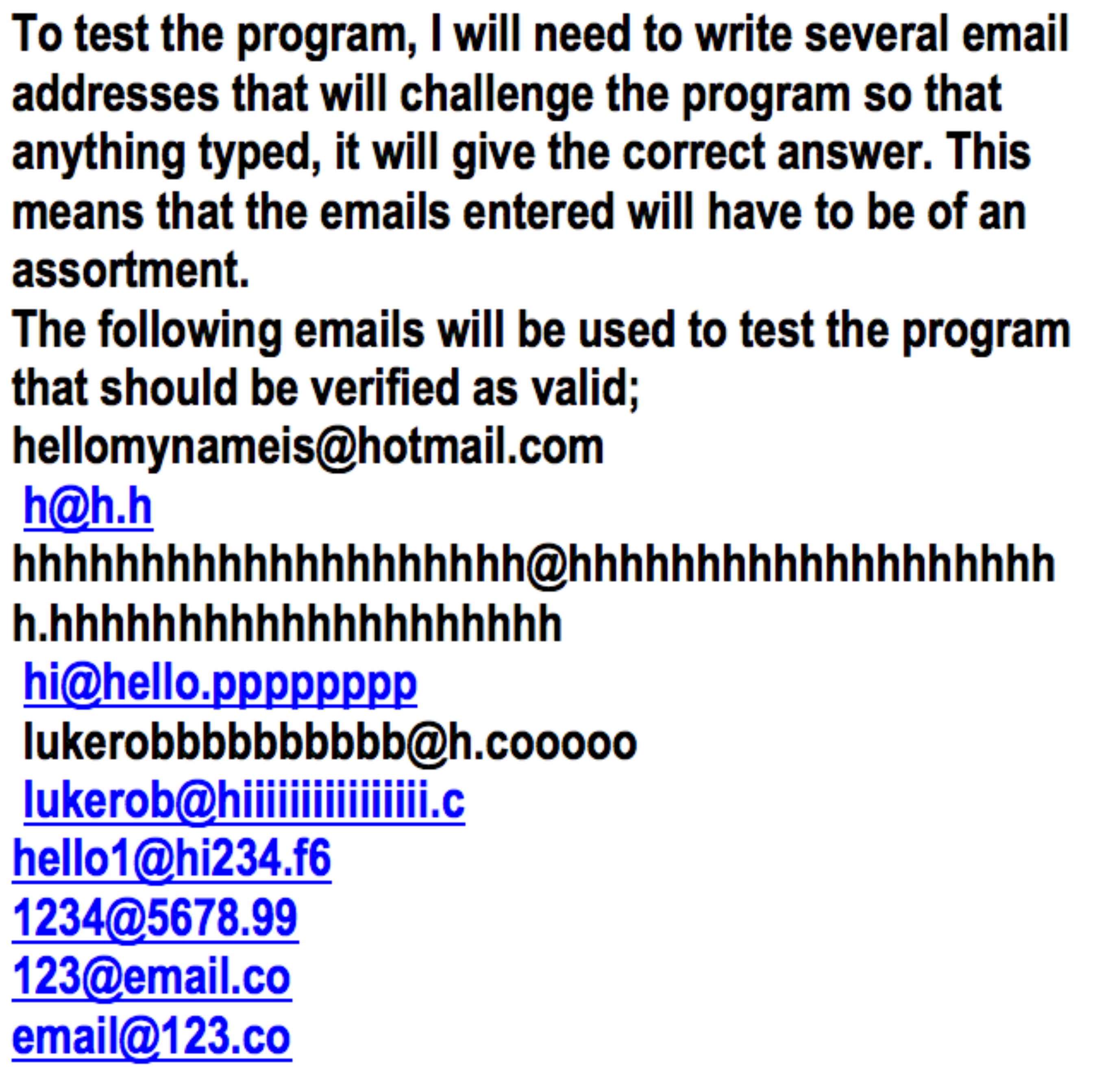


Task 1: This task asks for a program that will ask someone to enter an email address that follows a recognised layout by the program. The layout of the email entered should start with a string of alphanumeric characters, followed by the '@' symbol, another string of letters/numbers, followed by a "." and then more alphanumeric characters e.g. a@b.c or ab23@f45.d3. If the email entered is set out how the program intends it to be then once the python script runs, it should say 'VALID'. But if the email entered doesn't follow that layout e.g. @bc.d or 123.c@cvb, then the program should go onto say 'INVALID'.

The strategy I will use to solve this task is regular expression because I find it much simpler than arrays and have worked with them much recently. I am use to regular expressions so I feel if there is an invalid syntax then I will be able to notice a possible problem and fix it.

STEP 2 - PLAN


start
Ask user for email
Check that the email follows the layout
Email doesn't follow the correct layout
Print 'invalid'
Email follows the correct layout
Print 'valid'

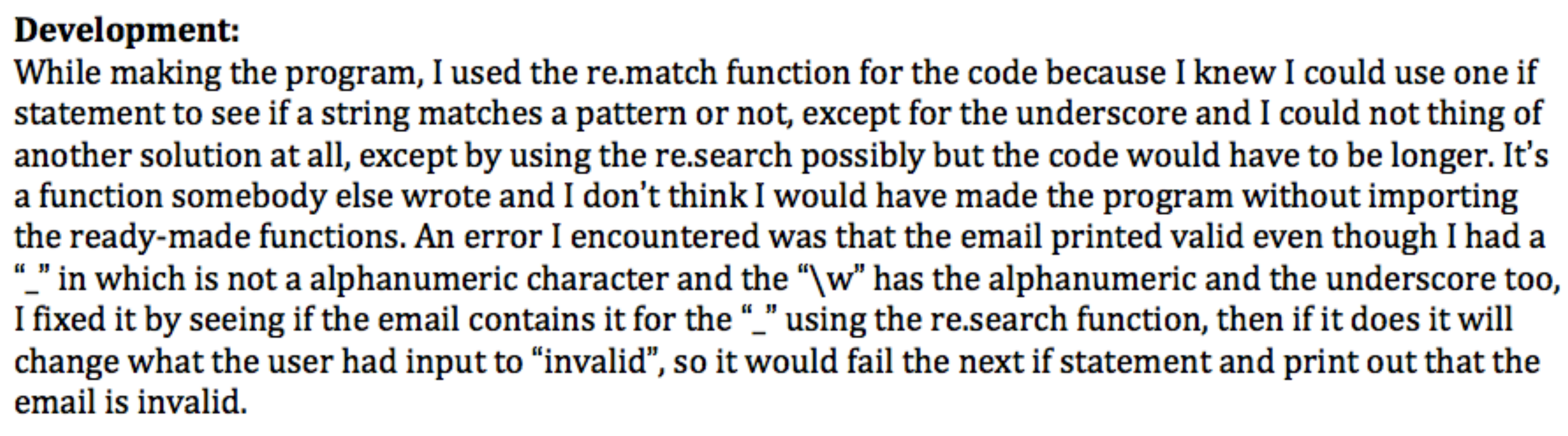


To test the program, I will need to write several email addresses that will challenge the program so that anything typed, it will give the correct answer. This means that the emails entered will have to be of an assortment.

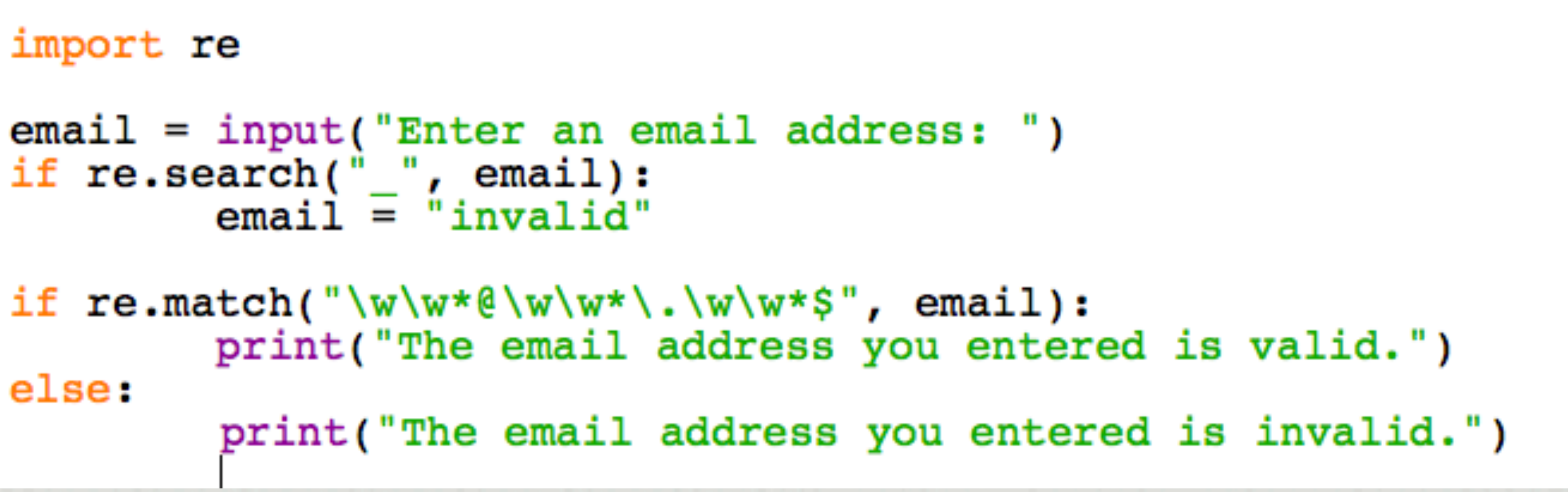
The following emails will be used to test the program that should be verified as valid;

hellomynameis@hotmail.com
h@h.h
hhhhhhhhhhhhhhhhhhhh@hhhhhhhhhhhhhhhhhhh
h.hhhhhhhhhhhhhhhhhhhh
hi@hello.ppppppp
lukerobbbbbbbbbb@h.cooooo
lukerob@hiiiiiiiiiiiiii.c
hello1@hi234.f6
1234@5678.99
123@email.co
email@123.co

STEP 3 - DEVELOP

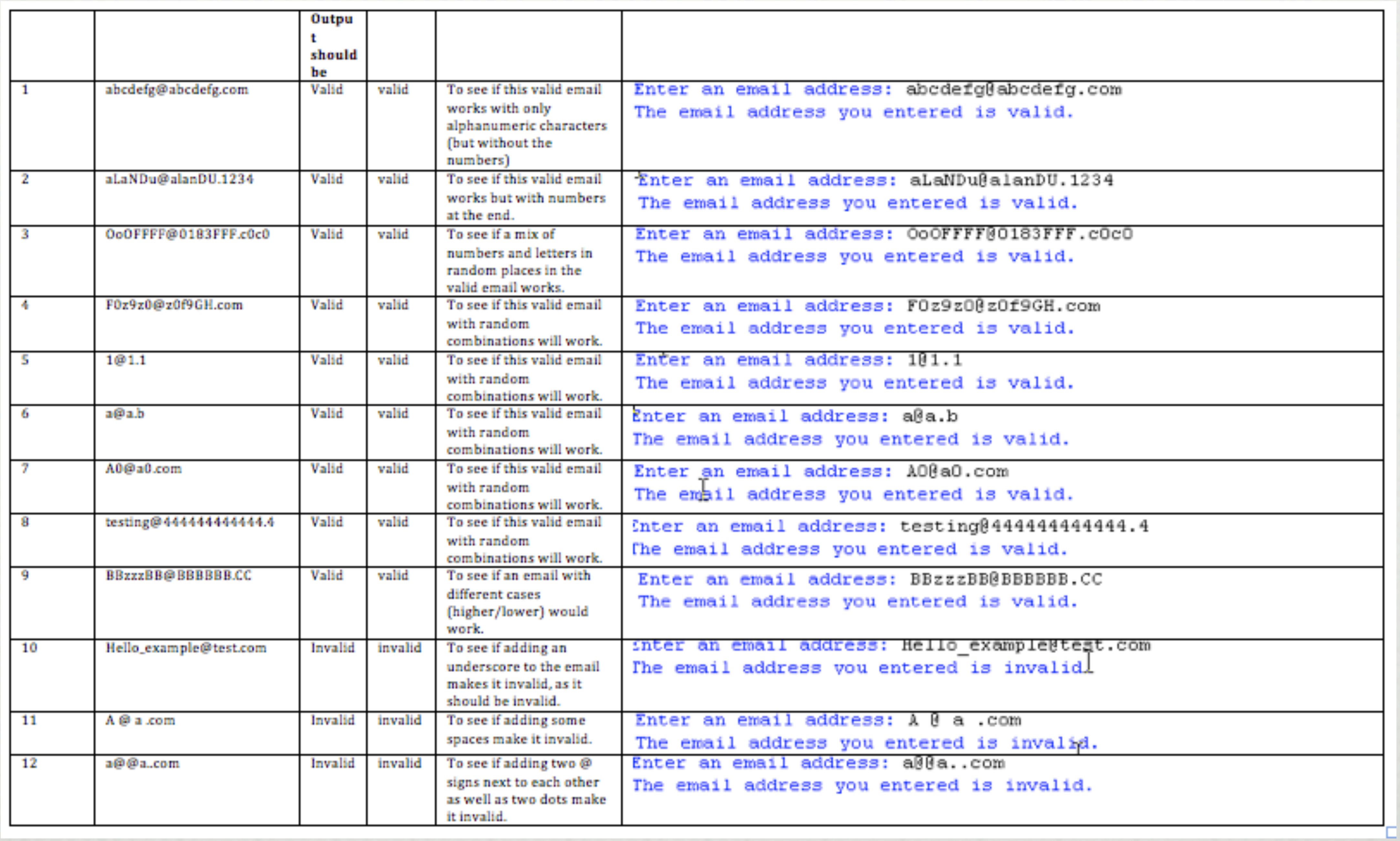


Development:
While making the program, I used the re.match function for the code because I knew I could use one if statement to see if a string matches a pattern or not, except for the underscore and I could not thing of another solution at all, except by using the re.search possibly but the code would have to be longer. It's a function somebody else wrote and I don't think I would have made the program without importing the ready-made functions. An error I encountered was that the email printed valid even though I had a "_" in which is not a alphanumeric character and the "\w" has the alphanumeric and the underscore too, I fixed it by seeing if the email contains it for the "_" using the re.search function, then if it does it will change what the user had input to "invalid", so it would fail the next if statement and print out that the email is invalid.



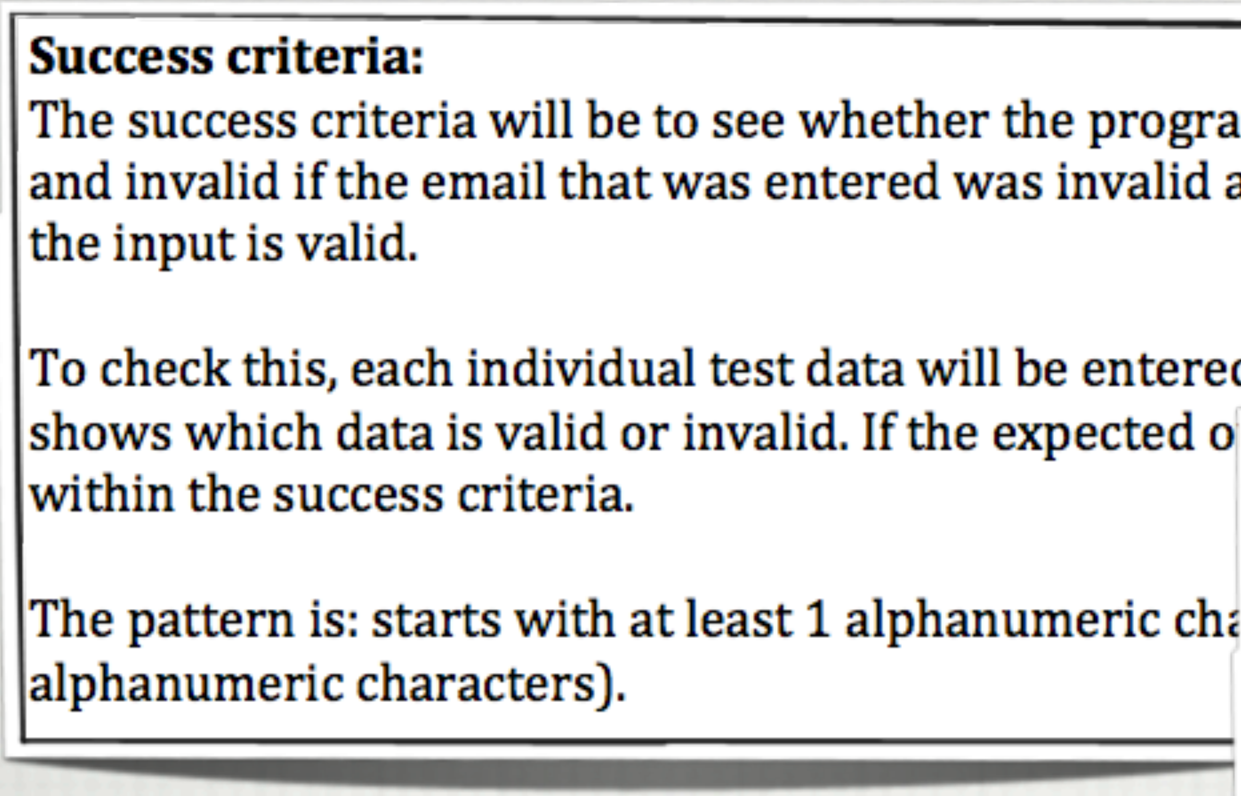
```
import re

email = input("Enter an email address: ")
if re.search("_", email):
        email = "invalid"

if re.match("\w\w*@\w\w*\.\w\w*$", email):
        print("The email address you entered is valid.")
else:
        print("The email address you entered is invalid.")
```

STEP 4 - TEST



		Output should be			
1	abcdefg@abcdefg.com	Valid	valid	To see if this valid email works with only alphanumeric characters (but without the numbers)	Enter an email address: abcdefg@abcdefg.com The email address you entered is valid.
2	aLaNDu@alanDU.1234	Valid	valid	To see if this valid email works but with numbers at the end.	Enter an email address: aLaNDu@alanDU.1234 The email address you entered is valid.
3	OoOFFFF@0183FFF.c0c0	Valid	valid	To see if a mix of numbers and letters in random places in the valid email works.	Enter an email address: OoOFFFF@0183FFF.c0c0 The email address you entered is valid.
4	F0z9z0@z0f9GH.com	Valid	valid	To see if this valid email with random combinations will work.	Enter an email address: F0z9z0@z0f9GH.com The email address you entered is valid.
5	1@1.1	Valid	valid	To see if this valid email with random combinations will work.	Enter an email address: 1@1.1 The email address you entered is valid.
6	a@a.b	Valid	valid	To see if this valid email with random combinations will work.	Enter an email address: a@a.b The email address you entered is valid.
7	A0@a0.com	Valid	valid	To see if this valid email with random combinations will work.	Enter an email address: A0@a0.com The email address you entered is valid.
8	testing@444444444444.4	Valid	valid	To see if this valid email with random combinations will work.	Enter an email address: testing@444444444444.4 The email address you entered is valid.
9	BBzzzBB@BBBBBB.CC	Valid	valid	To see if an email with different cases (higher/lower) would work.	Enter an email address: BBzzzBB@BBBBBB.CC The email address you entered is valid.
10	Hello_example@test.com	Invalid	invalid	To see if adding an underscore to the email makes it invalid, as it should be invalid.	Enter an email address: Hello_example@test.com The email address you entered is invalid.
11	A @ a .com	Invalid	invalid	To see if adding some spaces make it invalid.	Enter an email address: A @ a .com The email address you entered is invalid.
12	a@@a..com	Invalid	invalid	To see if adding two @ signs next to each other as well as two dots make it invalid.	Enter an email address: a@@a..com The email address you entered is invalid.

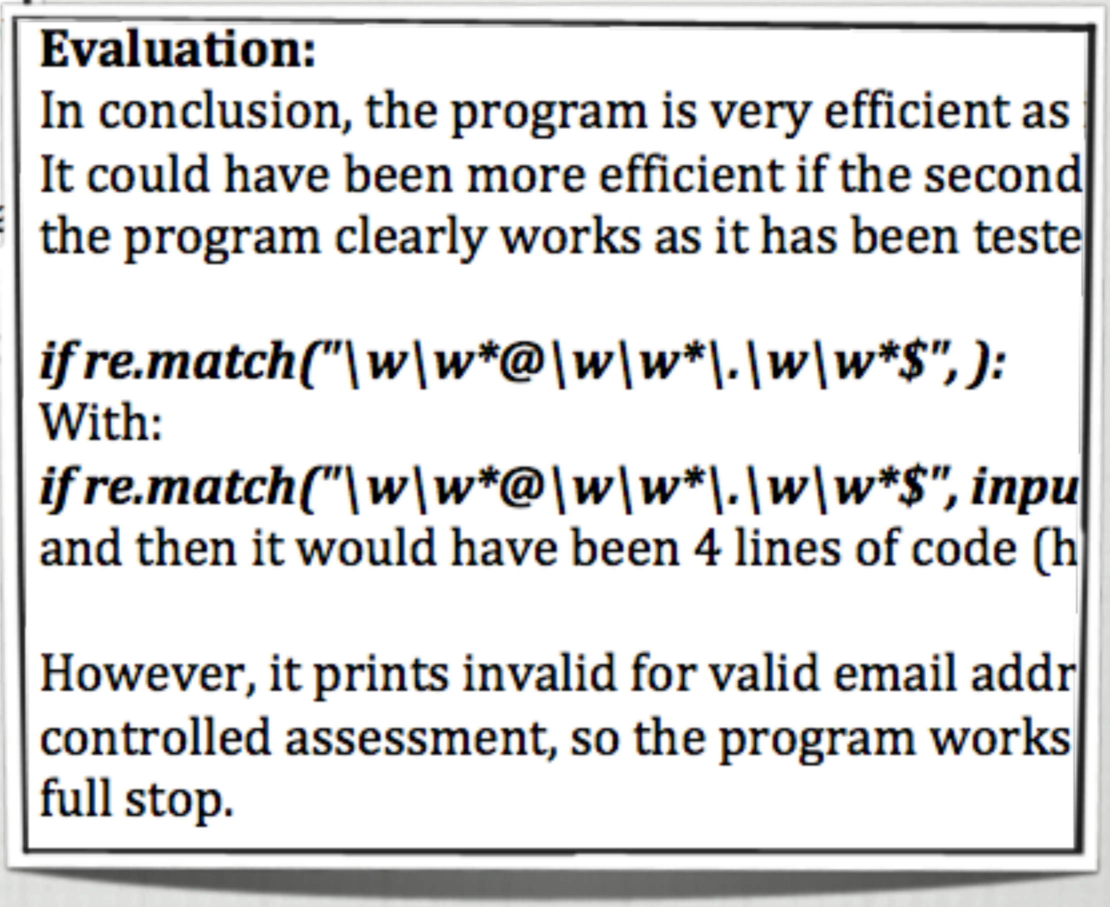
STEP 5 - EVALUATE



Success criteria:
The success criteria will be to see whether the progran
and invalid if the email that was entered was invalid ar
the input is valid.

To check this, each individual test data will be entered
shows which data is valid or invalid. If the expected o
within the success criteria.

The pattern is: starts with at least 1 alphanumeric cha
alphanumeric characters).



Evaluation:
In conclusion, the program is very efficient as
It could have been more efficient if the second
the program clearly works as it has been teste

if re.match("\w\w*@\w\w*\.\w\w*$",):
With:
if re.match("\w\w*@\w\w*\.\w\w*$", inpu
and then it would have been 4 lines of code (h

However, it prints invalid for valid email addr
controlled assessment, so the program works
full stop.

QUESTIONS?

TEESSIDEHUB.EVENTBRITE.CO.UK

- □ MARK CLARKSON
- □ PI.MWCLARKSON.CO.UK
- □ @MWCLARKSON
- □ MWCLARKSON@GMAIL.COM