1 try ... except (where code might fail)

In perfect code all eventualities will be anticipated and code written for all eventualities. Sometimes though we might want to allow for more general failure of code and provide an alternative route.

In the following code we try to perform some code on a list. For the code to work the list must be long enough and the number must be positive. We use try .... except to perform the calculation where possible and return a null value if not.

In Python an error is called an 'exception'.

```
import math

x = [1, 2, -4]

# Try to take the square root of elements of a list.
# As a user try entering a list index too high (3 or higher)
# or try to enter index 2 (which has a negative value, -4)

test = int(input('Index position?: '))

try:

    print (math.sqrt(x[test]))

except:

    print ('Sorry, no can do')


OUT:

Index position?: 2
Sorry, no can do
```

Try .... except may be used in final code (but it is better to allow for all specific circumstances) or may be used during debugging to evaluate variables at the time of code failure