

5.7-5.10: Hybrid Session 2

Prefer to complete this activity offline? Download the following documents:

* Full PowerPoint slides: [Week 5 Hybrid Session 2 Full Slides.pptx](https://winchester.instructure.com/files/1678203/download?wrap=1)
* Walkthroughs:
  + [5.7 Nesting for within a for loop.docx](https://winchester.instructure.com/files/1678197/download?wrap=1)
  + [5.7 Nesting (If Within a Loop).docx](https://winchester.instructure.com/files/1678196/download?wrap=1)
  + [5.8 Validating Input Using Nested Control Structures.docx](https://winchester.instructure.com/files/1678198/download?wrap=1)
  + [5.9 The break Keyword.docx](https://winchester.instructure.com/files/1678199/download?wrap=1)
  + [5.9 The continue Keyword.docx](https://winchester.instructure.com/files/1678200/download?wrap=1)

In this topic, we will be focussing on the following learning outcome for this week:

* Use iteration-based control structures within problems utilising Python
* Recognise situations where nesting of control structures is required

You will have the following learning opportunities:

* To demonstrate the use of nesting with loops (with other loops and other control structures)
* To explore the use of nesting to validate user input
* To appreciate the use of break and continue within loops



For this session, you will be engaging in acquisition, investigation, practice, and production learning activities.

# 5.7: Nesting with Loops

Last week we talked about how we can nest control structures. In this section we will be developing that concept to consider two further uses of nesting:

* Loops within Loops
* If statements within Loops

After recapping what nesting was, we focussed on two types of nesting which were not covered last week. We looked at how we can nest loops (so a loop within a loop) and how we can nest an if within a loop. You were then asked to complete two questions to practice your nesting.

# 5.8: Validating User Entry

In the sessions last week, we encouraged users to enter a number and then told them whether the number is even or odd and divisible by three. One of the flaws with this program is that it will produce an error if the user enters anything other than a number. In this section, you will be investigating how you can make use of nesting and loops to ensure that the type entered by the user is correct.

As we saw in the session, there is a need to ensure that we validate user entries when using the input() command – especially where numbers are involved – as it can result in an error which is not particularly helpful. In this part of the session, you were challenged to come up with a solution to this problem. We saw that it is possible to nest an if statement within a loop to “check” the user input before converting the string to a number. This eliminated the problem and removed the error produced by Python.

# 5.9: The break and continue Keywords

The break and continue keywords allow us to have more control over the loops that we are running. In this part of the week we will be looking at what these keywords are and how they operate in relations to looping.

The last part of new content covered this week was to investigate the break and continue keywords. As you saw in the session, this can allow us to have more control over the loops which are running. This is of particular importance when using a while True: loop (which is infinite unless we “break” out of its running).

# 5.10: Week Summary and Overview

In this final part of this session, we will be reiterating the topics we have covered during the pre-session materials, and the online session. We will talk about what you need to do next in terms of finishing the content for this week - which includes the practice questions and stretch tasks. Finally, we will talk about the topics we will be covering next week in the pre-session content and timetabled session.