



CoGrammar

Python Basics

**SKILLS
FOR LIFE**

SKILLS BOOTCAMPS



Department
for Education

Software Engineering Lecture Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly.
(FBV: Mutual Respect.)
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Open Classes.
You can submit these questions here: [Open Class Questions](#)

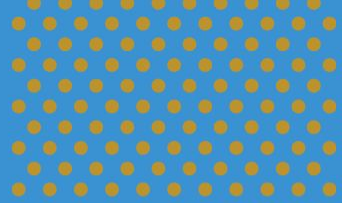
Software Engineering Lecture Housekeeping cont.

- For all **non-academic questions**, please submit a query:
www.hyperiondev.com/support
- Report a **safeguarding** incident:
www.hyperiondev.com/safeguardreporting
- We would love your **feedback** on lectures: [Feedback on Lectures](#)

Lecture Objectives

1. **Explain the concept of variables and their role in storing and managing data in Python.**
2. **Use conditional statements to control the execution of code.**
3. **Incorporate while and for loops to remove repetitive code.**

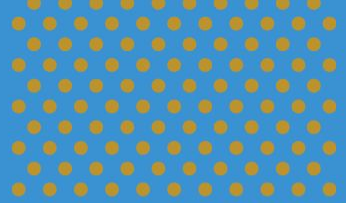
Python



- Python is a powerful and versatile programming language highly regarded in the field of software engineering and data science.
- Clean and readable syntax, along with a vast ecosystem of libraries and frameworks.
- Has a wide range of applications such as building web applications, data analysis, scientific computing and automation.
- Strong community support.



Output



- We use output to **communicate** with our users.
- You get different types of output but we will focus on output to the terminal using python's built in **print()** function.
- Name of function is **print**
- Execute function by adding **parentheses** after function name
- We can add all the **values** we would like **to print** to the terminal **inside** the **parentheses**.

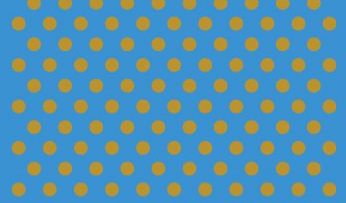


Output

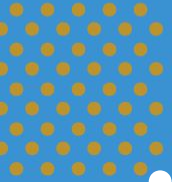
```
print("Hello World")
```

● PS D:\Work\DfE\CW> & C:/Python311/python.exe d:/Work/DfE/CW/test.py
Hello World

Input



- Input is how we **receive data** in our programs
- We will use a few different ways of getting input such as **hard coding** input, **terminal input** using the built-in **input()** function and **external files**.
- Hard coding is where we **set** the **values** for our program **directly in** our **code** instead of getting it from another source.
- Using **input()** we can execute Python's input function to **receive input** from the user **through** the **terminal**.
- Similar to print we can **add values** to the input function **to print** to the terminal **before** listening for **user input**.

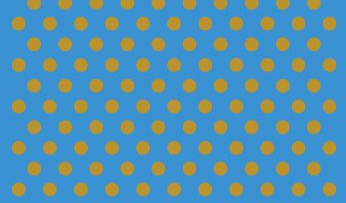


Input

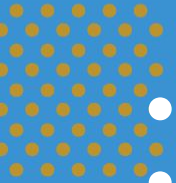
```
name = input("Please enter your name: ")
```

PS D:\Work\DfE\CW> & C:/Python311/python.exe d:/Work/DfE/CW/test.py
Please enter your name: Armand

Variables



- We use variables to **store data** for **later use**.
- We can **give** a **variable** a **name** and **provide** it a **value** to **reference back to**.
- E.g. `my_variable = "Hello World"`
- The variable `my_variable` now **references back** to the value "Hello World"
- Anywhere in my code I would like to use the value "Hello World" I can just use `my_variable`.



Variables

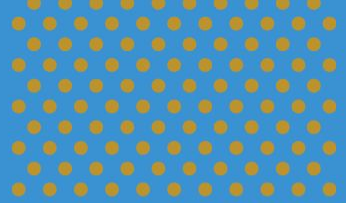
```
print("Hello World")
```

↓

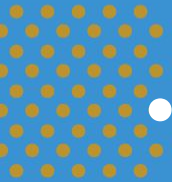
```
my_variable = "Hello World"  
print(my_variable)
```

• PS D:\Work\DfE\CW> & C:/Python311/python.exe d:/Work/DfE/CW/test.py
Hello World

Variables Types



- There are a bunch of data types in Python.
- We will focus on the more common types such as **strings**, **integers**, **floats** and **booleans**.
- Strings are a **sequence of characters** that we usually use to **represent text**.
- **Integers** and **floats** are **numerical values** and can be used to perform **mathematical operations**.
- Booleans are a **binary data type** that can be only **True** or **False**.



Variables Types

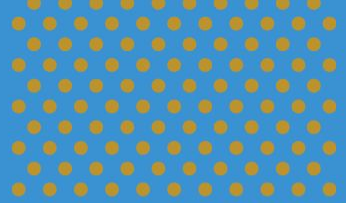
```
my_str = "This is a string"
```

```
my_int = 13
```

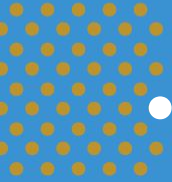
```
my_float = 23.54
```

```
my_bool = True
```

Conditional statements



- We can **control** the **execution** of our code by only **running** certain **parts of** our **code** when a **certain condition** has been met.
- Conditions **equate** to either a **True** or **False** value that can be **used** within an **if-statement** and a **while loop**.
- Using conditions we can **determines** things such as if **values are equal or not** and **bigger or smaller** than each other.
- We can then **add** the **code to execute** when the condition is met **inside** the **if-statement** or **while loop** with the condition.

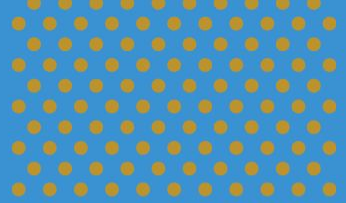


Conditional statements

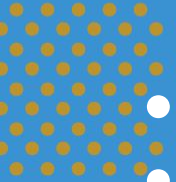
```
same_word = "Bird" == "Bird" # True
same_word = "Bird" == "bird" # False
not_same_word = "Duck" != "Duck" # False

larger_num = 20 > 15 # True
smaller_num = 3 < 10 # True
same_num = 5 == 8 # False
```


If and while



- If-statements and while loops have more in common than we think
- They both **execute code based on** a **condition** being **True**.
- An if statement will **execute the code** inside the statement **once** if the condition is True.
- A while loop will **continuously execute** the code inside the loop **as long as** the **condition is True**.

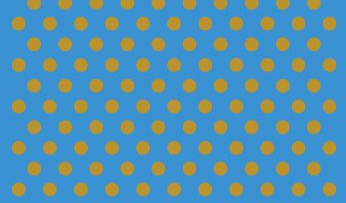




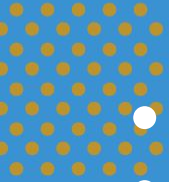
Condition is True


```
Condition is True
Condition is True
Condition is True
Condition is True
Condition is True
Condition is True
Condition is True
```

For Loops




- We can use for loops to **execute** a block of **code** for a **set amount of iterations**.
- This helps us **get rid** of **repetitive code**.
- If we wanted to print the **same thing 10 times** it will be **easier** to print **using** a **for loop** that has **10 iterations** and add the print inside the loop.
- For loops can iterate over **any iterable object** such as a range, string, list and many more.



For Loops

```
print("Welcome")
print("Welcome")
print("Welcome")
print("Welcome")
print("Welcome")
print("Welcome")
print("Welcome")
print("Welcome")
print("Welcome")
print("Welcome")
```



```
for i in range(10):
    print("Welcome")
```

CoGrammar

Questions



CoGrammar

Thank you for joining

