Welcome to the Python Intermediate Course

Object Oriented Programming

Welcome to the Python Intermediate Course

Object Oriented Programming

Lesson 1

Estimated time: 15min

Off-screen activity

Let's introduce ourselves!

What is this course about?

- We assume you already know the basics of Python:
  - syntax - commas, brackets, indentation….
  - functions - calling and defining
  - conditional statements (if, elif, else)
  - loops (for and while)
  - logical operators (AND, OR, NOT)
  - comparison operators (==, =!, >, <, >=, <=)
  - inputs
  - modules - importing and using
  - general principles: comment your code, don't repeat yourself, etc…..

But if you don't remember all of it don't worry!

What is this course about?

- **We will first do some revision to make sure you are all back up to speed before we move on to the next stage!**
- We will then download some software that will help us with this course:
  - An “adult” IDE (code editor) in place of Mu
  - Github - who knows what this is?
  - Python Arcade - a special module that will help us write more sophisticated games in Python
- The main part of the course will concern Object Oriented Programming (OOP).
  - Who has heard of this before?
  - It is used by almost all professional programmers when writing their coding projects
- We need OOP to write complex games in Python Arcade

What we will do today?

- Let’s remember the basics!
- Some revision exercises on Turing Lab

First let's check:

- **Does everybody already have:**
  - Python downloaded on your laptop
  - Turing Lab Account
  - Mu editor downloaded on your laptop (only for this lesson)

What can you remember about Python?

What can you remember about Python?

- **What do you understand by syntax?**
  - When do we need to use:
    - Quote marks “ or ‘
    - Round Brackets ()
    - Square Brackets []
    - Colon :
    - Indentations

Coding challenge 1

- Using your Mu editor in Python mode
- Define variables called name and age and give them the name and age of a person (you choose)
- Write a print command to say hello to the person named in the variable and tell them how old they are.
- *Bonus challenge: also print their year of birth.*

What can you remember about Python?

- **What is a variable and what do we use it for?**
  - What types of data can we store in a variable?
    - String
    - Integer
    - Float
    - Boolean
  - What different ways can we store something in a variable:
    - Directly assign it: e.g. age = 12
    - Store an input: e.g. age = input("what is your age?")
    - Store a return value from a function: e.g. age = get_age()

Coding challenge 2

- **Using your Mu editor in Python mode**
- **Ask the user their name and their country of birth using the input function and store the answers in variables**
- **Write a print command to introduce the person named in the variable and say where they were born.**
- ***Bonus challenge: also ask their date of birth and display that in the print function.***

What can you remember about Python?

- **What is a loop and what types are there?**
  - For loops:
    - with index variable e.g. for i in range(10):
    - looping through a list e.g. for animal in zoo:
  - While loops:
    - e.g. while timer < 50:
- **What is a list variable and how do we write it?**
  - e.g. zoo = [“tiger”, “lion”, “elephant”, “penguin”]

Coding challenge 3

- **Using your Mu editor in Python mode**
- **Write a for loop to print the 2x table**
  - the print line should say something like 1 x 2 = 2
  - go up to 10 x 2 = 20
  - each answer is calculated!
- ***Bonus challenge: create a list variable with each person's name in class. Use a for loop to print each name in turn.***

What can you remember about Python?

- **What do we mean by a conditional statement?**
- **How do we write it?**
  - e.g. if light > 800:
        turn_streetlights_on()
- If two alternatives: use else
- If three or more: use elif for the middle ones
- Use comparison operators to write the condition e.g. ==, !=, >, <, >=, <=
- Combine or alter comparisons with and, or and not

Coding challenge 4

- Using your Mu editor in Python mode
- A theme park has a minimum height of 140cm for the adult rides and 110cm for the junior rides
- Ask the user for their height
- Use conditionals to tell them if they are allowed on all rides, only the junior rides or not on any rides
- *Bonus challenge: there is also a minimum age of 11 for the adult rides. Ask the user their age and add this condition to the if statements*

Break time

What can you remember about Python?

- **What is a module and why do we use them?**
- **How do we use a module in our programs?**
  - e.g. from random import randint
  - but first make sure the module has been downloaded in your code editor/IDE!

Coding challenge 5

- Using your Mu editor in Python mode
- import the randint function from the random module
  - random is already built-in to Mu!
- using a for loop print 5 random integers between 1 and 100

What can you remember about Python?

- **What is a function?**
- **How do we use a function?**
  - call a function
- **How do we define a function?**
- **What is an argument?**
- **What is a return value?**

Time for think of some everyday functions!

Coding challenge 6

- **Using your Mu editor in Python mode**
- **define a function to square a number (multiply by itself)**
  - it takes an integer as an argument
  - it returns the square: e.g. with an input of 5 it returns 25, with an input of 9 it returns 81 etc…
- **in the main program, ask the user for a number**
- **print out the square of that number by using the function**

To finish our revision of basic Python - we will complete a special revision exercise in Turing Lab

- Log on to your Turing Lab account
- The Module is called “GCSE Revision Pack”
- You will already have been assigned to this class and been assigned this mini course

Upcoming...

In the next lesson:

- We will download a fully functional IDE
- We will learn about version control and Github
- We will download Github desktop and sign up to get a Github account
- We will download the Python Arcade module
- We will start to learn how to use the module

Welcome to the Python Intermediate Course

Object Oriented Programming

Lesson 2

What we will do today?

- We will download a fully functional IDE
- We will learn about version control and Github
- We will download Github desktop and sign up to get a Github account

Time for the data type game!

Let’s download an IDE! (1)

- What is an IDE?
  - Google*: difference between IDE and text editor
  - Google*: Popular IDE
- Mu is an IDE for Python beginners with a built in game design module (Pygame Zero) for basic game design
- Pycharm is an IDE specifically for Python and is used by professional programmers
- Python Arcade works best with this type of IDE

*other search engines are available!

Let’s download an IDE! (2)

- If you are going to download Pycharm, you can Google it or use the download link is as follows:
- www.jetbrains.com/pycharm/download
- The Windows and Apple versions are different, but the link should automatically select the right one
- Scroll down to download the community edition
- If you want to use a different IDE, e.g. Visual Studio: similarly make sure to choose the free edition. Check with an instructor if in doubt.

Using your IDE

- Creating a project in Pycharm
    - What is a virtual environment?
- Lets try writing some simple programs
- What do you notice about the text editor window?

Watch at home: here's a useful beginners guide to using Pycharm:
https://www.youtube.com/watch?v=HHcZbXsZtm0

Estimated time: 15min

Break time

What is version control?

- https://www.youtube.com/watch?v=2ReR1YJrNOM

- Git = the software we have on our computer to manage the version control / backup of our projects
- Github = the website which allows us to save our project repositories on the internet
- First you will set up a Github account
- Next you will download Github Desktop
  - This is a version of Git which is easier for beginners
  - It works by point/click rather than having to type Git commands in the command line interface

Setting up your Github account

- First you will need access to an e-mail account
- You will need to type in the e-mail address, create a password and verify the account
- Make sure you let your browser save your password
- After verification answer the questions, but make sure you choose the free account version
- Now you can personalise your account!

Installing Github Desktop

- First you will need access to an e-mail account
- You will need to type in the e-mail address, create a password and verify the account
- Make sure you let your browser save your password
- After verification answer the questions, but make sure you choose the free account version
- Now you can personalise your account!
- https://www.youtube.com/watch?v=8Dd7KRpKeaE

Upcoming...

In the next lesson:

- We will introduce the Python Arcade module and its documentation
- We will “fork” a repository containing tutorial files for learning Python Arcade
-

Welcome to the Python Intermediate Course

Object Oriented Programming

Lesson 3

What we will do today?

- We will introduce the Python Arcade module and its documentation
- We will “fork” a repository containing tutorial files for learning Python Arcade

Python Arcade

- You already used a Python module to help you write computer games: Pygame Zero
- This is a bit limited in what you can do with it, so in this course we will use Python Arcade
- You can find all the documentation for this module at this website:
    - https://learn.arcade.academy
- We will follow this loosely for the rest of the course
    - We are currently following the topics in chapters 2 & 3
    - We will skip most of chapters 4 to 15 as these topics were covered in Python Beginners and our revision in the first lesson

Remember the key advantages of version control systems and code repositories:

- you can go back to a previous version if your code changes have gone wrong
- you have a copy of all your work on the cloud if you have a hardware failure
- you can code in teams so that more than one person can work on a program at the same time
- you can find useful code and code ideas on the internet repos (GitHub)

We will now fork code repositories to help us learn OOP and Python Arcade:

- first let's fork a repository that contains info for our course
- first make sure you are logged in to your GitHub account on your browser
- now go to this GitHub repo webpage:
  - https://github.com/batecsw/OOP_class_work
- press the button saying "fork" - this creates a copy of the repo on your GitHub account
- however, this is on the cloud, not your computer: next you need to make a copy on your hard drive, called a "clone"
  - go to GitHub desktop, from the file menu select "clone repository", by default your GitHub repos are shown
  - if this is your first repo, you will instead see a "Let's get started" welcome screen, and you can select "Clone a repository from the internet" instead
- select "OOP_class_work" and change the "local path" to "Pycharm Projects" - this means the folder will be in the right place for your Pycharm IDE.
- now click on "Clone", select "for my own purposes" when asked how you are going to use it.

Forking and Cloning (2):

- this is a bit complicated, but the good news is that GitHub Desktop will now look after keeping the repo on your computer and the copy in the cloud on github.com synced
- go to Pycharm - you should be able to open this folder
- open the Word document - I will keep this document updated to contain all the useful links you need, so you don't have to type them out!
- next go to this GitHub repo webpage:
    - https://github.com/pythonarcade/learn-arcade-work
- this contains the tutorials and example programs for Arcade
- repeat the process to fork the repo to your GitHub and clone it to your hard drive

Install Python Arcade

- after forking/cloning the Arcade tutorial repo, go back to Pycharm and open "Learn Arcade Work" as a new project
- in the lab 01 folder, open the lab_01 python file and type in a "Hello World" print command
- Pycharm saves changes automatically - but after the first save it realises you don't have the Arcade module needed to run some of the programs inside the project
- It will prompt you to install the Arcade module - go ahead and install it
- IMPORTANT: Arcade is not yet compatible with Python 3.12. If you recently installed or updated Python, you will need to roll it back to the 3.11 version. Your Python version is shown on the bottom right of the Pycharm window. Ask for help if this applies to you!

Committing your code!

- go back to GitHub Desktop and select the learn-arcade-work repo if it isn't the one already selected
- the learn-arcade-work project folder is a local repository which is being monitored by GitHub Desktop
- it has noticed that the Lab_01.py file has changed!
- you can now "commit" this to the code repo on your hard drive
- to complete the process you should now "push commits to origin remote" - this means saving the changes to your GitHub repo in the cloud
- you should commit your code changes AT LEAST once per lesson at the end, more often if you make a lot of changes

Break time

Using the Learn Arcade Tutorial

- the tutorial is split into “chapters” which contain the information and “labs” which are practice exercises for you to code
- let’s follow along with “5. How to Draw with Your Computer” chapter
- at home you can open the Lab_2.py file and draw your own picture

Upcoming...

In the next lesson:

- We will learn about classes and objects - Object Oriented Programming
- We will use these to create a text based adventure game

Welcome to the Python Intermediate Course

Object Oriented Programming

Lesson 4

What we will do today?

- We will learn how to draw to the screen using Python Arcade functions
- We will learn about Classes and Objects - Object Oriented Programming
- We will use these to create a text based adventure game

Using the Learn Arcade Tutorial

- the tutorial is split into “chapters” which contain the information and “labs” which are practice exercises for you to code
- let’s follow along with “5. How to Draw with Your Computer” chapter
- at home you can open the Lab_2.py file and draw your own picture
- Note that you can look up how to use all the Arcade functions at this address:
  - https://api.arcade.academy/en/latest/quick_index.html

Object Oriented Programming

- What is OOP?
- What is a Class
- What is an object?

Object Oriented Programming

- **A class is like a blueprint or design for all the objects created using the class**
- **Variables inside an object are called attributes or instance variables**
- **Functions within an object are called methods**
- **Class names start with a capital letter**
- **Classes contain a special ___init___ function**
    - it is an example of a “magic” or “dunder” function, and has two underscores before and after the name
    - the init function is sometimes called the “constructor” function
    - the job of the constructor is to set up the variables
- **You can find a full description in Chapters 16 and 17 of the Arcade Academy**

Object Oriented Programming

- Who has done OOP Before?

Break time

Text Based Adventure Game

- **We will use classes and objects to build a multi-room house/castle/dungeon**
- **On the Arcade Academy website you can find the full instructions for this on the Lab 6 page:**
  - https://learn.arcade.academy/en/latest/labs/lab_06_text_adventure/adventure.html
- **First draw a plan of your house/dungeon!**

Welcome to the Python Intermediate Course

Object Oriented Programming

Lesson 5

What we will do today?

- Let's see what pictures you made!
- Recap on Classes and Objects - Object Oriented Programming
- We will use these to create a text based adventure game
-

Drawing Pictures

- When you make your own game, you will probably use a combination of drawing shapes like in this exercise plus using sprites as well
- If you have not done this already, make sure you read and follow chapter 5 in Arcade Academy and watch the linked YouTube video
- Doing this plus the lab_02 exercise helps practice this skill and familiarises you with using coordinates on the screen
- Let's see what you made!

Object Oriented Programming

- A class is like a blueprint or design for all the objects created using the class
- Variables inside an object are called attributes or instance variables
- Functions within an object are called methods
- Class names start with a capital letter
- Classes contain a special __init__ function
  - it is an example of a “magic” or “dunder” function, and has two underscores before and after the name
  - the init function is sometimes called the “constructor” function
  - the job of the constructor is to set up the variables
- You can find a full description in Chapters 16 and 17 of the Arcade Academy

Object Oriented Programming

- **Who has done OOP Before?**

- **In Scratch sprites are a class and each sprite is an object**
  - each sprite has attributes (x, y, size, direction, costume)
  - the code you write for each sprite are its methods
- **In Pygame Zero all sprites are objects of the Actor class**
  - You already used the Python “dot notation”
  - For example this line sets “ship” as an object of the Actor class and passes the image file into the constructor:
    - ship = Actor("playership1_red")
  - We just did not discuss what the underlying meaning was, as it was a tough topic for the beginners course
  - Pygame Zero already had the Actor class built-in so you did not have to do the class definition yourself

OOP - a simple example

The following code defines a class called “Dog”, each time a new dog object is set up, constructor will give the dog an age, a name and a weight as attributes, and the dog will have the bark method (function). In this example the attributes are given default values by the constructor:

```
class Dog():
    def __init__(self):
        self.age = 0
        self.name = ""
        self.weight = 0

    def bark(self):
        print("Woof")
```

OOP - a simple example

We can now create an instance of the Dog class like this:

```
my_dog = Dog()
```

And set its attributes like this:

```
my_dog.name = "Spot"
my_dog.weight = 20
my_dog.age = 3
```

Note the “dot notation” when referring to the attributes of an object e.g.
my_dog.age
The same notation is used when calling a method (function) of the object:

```
my_dog.bark()
```

The brackets tell you it is a method and not an attribute.

OOP - a simple example

We can also modify the constructor to make the program define the attributes when the object is set up:

```
class Dog(age, name, weight):
    def __init__(self):
        self.age = age
        self.name = name
        self.weight = weight
```

Now, the class has name, age and weight as arguments, this requires that these are passed as arguments when an object of the class is set up:

```
my_dog = Dog(3, “Spot”, 20)
```

Now the my_dog object has its attributes set up straight away. This is the most usual of working with objects and classes.

Break time ⏰

Text Based Adventure Game

- **We will use classes and objects to build a multi-room house/castle/dungeon**
- **On the Arcade Academy website you can find the full instructions for this on the Lab 6 page:**
    - https://learn.arcade.academy/en/latest/labs/lab_06_text_adventure/adventure.html
- **First draw a plan of your house/dungeon!**

Text Based Adventure Game

- **What more can you do with this game?**
- **Add additional rooms:**
  - You can do this by adding to the set-up table, to create the room object within the existing loop
  - Or you can set up the object individually
- **Add an additional floor/basement/dungeon level to use the up/down directions**
- **To develop the game further:**
  - Have objects which you can pick up and use
  - Have other characters/monsters which you meet