CoGrammar

Functions

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. Define what a function is in programming
2. Expand on User-defined functions and libraries.
3. Apply the above knowledge to improve data management in programs

Poll:

Assessment

What is a Function?

- ★ Reusable and Organised block of code.
- ★ Sometimes called a ‘method’.
- ★ Similar to functions in maths - f(x) takes input x and produces some output.
- ★ Useful for abstraction.
- ★ For example, “make a cup of tea” vs “boil water, add tea bag, add sugar, add milk, stir”.

Calling Functions

- Functions with one required positional input:
  - `my_function(input1)`
- Functions with two required positional inputs:
  - `my_function(input1, input2)`
- Functions with one required positional input and one optional keyword input:
  - `my_function(input1, keyword_arg=input2)`

Functions in Python

- ★ Python comes bundled with in-built functions.
- ★ Examples:
  - print**(string)** - prints string to console.
  - input**(string)** - prints string to console, then reads input from the console as string.
  - len**(array)** - returns the length of an array.
  - int**(data)** - converts the value to an integer.

...but wait! There's more!

- ★ The list of functions that you can use in Python doesn't just stop with what is built in.
- ★ Using Pip (python package manager), you can install various packages containing modules.
- ★ Note: Some packages are already installed by default in Python, such as the Math package.
- ★ These modules can be imported into your script using an import statement.

Importing Libraries

★ Let's take a look at the maths module. Let's say that you want to use round(), which rounds a number off. There are two ways to access this:

a.
```
import math
my_result = math.round(my_num, 2)
```

b.
```
from math import round
my_result = round(my_num, 2)
```

Defining our own Functions

- Uses the def keyword (for define):
  - `def add_one(x): # function called add_one`
  - `    y = x + 1`
  - `    return y`
- Important keywords:
  - def - tells Python you are defining a function
  - return - if your function returns a value, then use this keyword to return it.

Some Important Terms

- **Function** - A block of code that performs an action(in the form of code).
- **Method** - Similar to a function(Called Differently).
- **Parameters** - The defined input of a function.
- **Arguments** - The values passed to parameters.

Why Functions?

- ★ **Reusable code** - Sometimes you need to do the same task over and over again.
- ★ **Error checking/validation** - Makes this easier, as you can define all rules in one place.
- ★ **Divide code up into manageable chunks** - Makes code easier to understand.
- ★ **More rapid application development** - The same functionality doesn't need to be defined again.
- ★ **Easier maintenance** - Code only needs to be changed in one place.

Poll:

Assessment

Wrapping Up

Libraries

Libraries in programming are collections of pre-written code, providing ready-to-use functions.

User defined functions

User-defined functions in programming are custom-created blocks of reusable code that perform specific tasks within a program.

CoGrammar

Questions around Functions

CoGrammar

Thank you for joining