Python Basics

Basic Syntax

```
if True:
    print("Hello, Python!")
```

This code checks if the condition True is met and then prints "Hello, Python!" because the condition is always true.

Variables and Data Types

Variables	Data Types
Used to store data values.	Different types of collections to store multiple items.

`x = 10 # Integer`
`y = 3.14 # Float`
`name = "Alice" # String`
	`# List`
	`fruits = ["apple", "banana", "cherry"]`
	
	`# Tuple`
	`coordinates = (10.0, 20.0)`
	
	`# Dictionary`
	`person = {"name": "Alice", "age": 25}`

Variables x, y, and name are assigned different types of values: integer, float, and string respectively.	Fruits is a list, coordinates is a tuple, and person is a dictionary, each storing multiple values.

Basic Operations

Arithmetic	Comparison	Logical
Basic mathematical operations.	Compare values.	Combine conditional statements

`addition = 5 + 3`
`# 8`
 	
`subtraction = 5 - 3`
`# 2`
 	
`multiply = 5 * 3`
`# 15`
 	
`division = 5 / 3`
`# 1.6667`
	`is_equal = 5 == 3`
	`# False`
	
	`not_equal = 5 != 3`
	`# True`
	
	`greater_than = 5 > 3`
	`# True`
		`and_oper = True and False`
		`# False`
		
		`or_oper = True or False`
		`# True`
		
		`not_oper = not True`
		`# False`

Performing addition, subtraction, multiplication, and division operations on numbers.	Comparing two values to check equality, inequality, and relative magnitude.	Using logical operators to combine or invert boolean values.

Control Structures (IF statements)

Execute code based on conditions

```
if x > 0:
    print("x is positive")
elif x == 0:
    print("x is zero")
else:
    print("x is negative")
```

This code checks if x is positive, zero, or negative and prints the corresponding message

Loops

Execute a block of code multiple times

For Loop	While Loop
Iterate over a sequence.	Execute as long as a condition is true.

`for i in range(5):`
`  print(i)`
 	
`# --- Output`
`# 0`
`# 1`
`# 2`
`# 3`
`# 4`
	`count = 0`
	`while count < 5:`
	`  print(count)`
	`  count += 1`
	
	`# --- Output`
	`# 0`
	`# 1`
	`# 2`
	`# 3`
	`# 4`

This loop goes through numbers 0 to 4 and prints each one.	The while loop runs until “count” is 5, printing count and then increasing it.

Functions

Break the code into reusable parts.

```
def add(a, b):
    return a + b

print(add(3, 5))
# 8
```

This function “add” takes two inputs, a and b, and returns their sum. It then calls the function with 3 and 5, and prints the result.

Built-in Functions and Libraries

Built-in Functions	Using Libraries
Predefined functions provided by Python.	Import and use additional functionality.

`print("Hello, World!")`
`# Prints Hello, World!`
 	
`user_input = input("Enter something: ")`
`length = len("Python")`
`# 6`
	`import random`
	
	`random_number = random.randint(1, 10)`
	`# Random number between 1 and 10`
	
	`import math`
	
	`square_root = math.sqrt(16)`
	`  # 4.0`

print displays output, input reads user input, and len returns the length of a string.	The “random” module generates random numbers and the “math” module provides mathematical functions like sqrt for calculating square roots.

Additional Resources:

https://www.datacamp.com/cheat-sheet/getting-started-with-python-cheat-sheet
https://indico.cern.ch/event/865287/attachments/1971788/3280306/beginners_python_cheat_sheet_pcc_all.pdf