Introduction to Coding

Tan Sein Jone

University of British Columbia

July 15, 2024

Table of contents

Table of Contents

Hello World

- The first program that every programmer writes
- How to start a Python program
- How to print to the console

In Terminal

```
touch hello_world.py
```

hello_world.py

```
print("Hello, World!")
```

Table of Contents

What are Variables?

- Variables are used to store data
- Variables are assigned a value
- Variables can be changed

Variable Naming Rules

- Variables must start with a letter or underscore
- Variables can only contain letters, numbers, and underscores
- Variables are case-sensitive
- Variables cannot be reserved words

Variable Naming Conventions

- Camel Case: myVariableName
- Pascal Case: MyVariableName
- Snake Case: my_variable_name

Scope

- Global Variables: Variables declared outside of a function
  - Can be accessed anywhere
- Local Variables: Variables declared inside of a function
  - Can only be accessed within the function

```
def my_function()\:
    x = 10
x = 20
my_function()
print(x)
```

Table of Contents

Data Types

- Integers: Whole numbers
- Floats: Numbers with decimals
- Strings: Text
- Booleans: True or False
- Lists: Ordered collection of items
- Tuples: Ordered collection of items that cannot be changed
- Dictionaries: Unordered collection of items
- Sets: Unordered collection of unique items

Importance of Type Checking

- There are advantages and disadvantages to using each data type
- Interacting with different data types can cause errors

```
random_int = 10
random_float = 10.0
random_string = "10"
```

Type Casting

- Converting between data types
- Can be done using built-in functions
- Not all conversions are possible

```
converted_int = int(random_float)
```

Data Types with Multiple Values

- Lists, Tuples, Dictionaries, and Sets can store multiple values
- Each value can be a different data type
- Each value can be accessed using an index
- Each value can be changed

```
random_list = [10, 10.0, "10"]
random_tuple = (10, 10.0, "10")
random_dict = {"int": 10, "float": 10.0, "string": "10"}
random_set = {10, 10.0, "10"}
```

Common Usecases

- Lists and dicitonaries will likely be the most used data types
- Lists are used to store multiple values
- Dictionaries are used to store key-value pairs

Combining Data Types

- Data types can be combined
- Lists can store dictionaries
- Dictionaries can store lists

Lists of dictionaries

- Say for example we have a list of students
- Each student has a name, age, and grade
- We can store this information in a list of dictionaries
- Each dictionary will represent a student

```
students = [
    {"name": "Alice", "age": 20, "grade": 90},
    {"name": "Bob", "age": 21, "grade": 85},
    {"name": "Charlie", "age": 22, "grade": 80}
]
```

Dictionaries of lists

- Say for example we have a dictionary of students
- Each student has a list of grades
- We can store this information in a dictionary of lists
- Each key will represent a student

```
students = {
    "Alice": [90, 85, 80],
    "Bob": [85, 80, 75],
    "Charlie": [80, 75, 70]
}
```

Common methods for lists and dictionaries

- append(): Adds an element to the end of the list
- insert(): Adds an element at a specific index
- remove(): Removes an element from the list
- get(): Gets the value of a key in a dictionary
- keys(): Gets all the keys in a dictionary
- values(): Gets all the values in a dictionary

Table of Contents

What are Conditionals?

- Conditionals are used to make decisions
- Conditionals are used to execute code based on a condition
- Conditionals are used to compare values

Comparison Operators

- ==: Equal to
- !=: Not equal to
- <: Less than
- >: Greater than
- <=: Less than or equal to
- >=: Greater than or equal to

Logical Operators

- and: Returns True if both statements are true
- or: Returns True if one of the statements is true
- not: Returns True if the statement is false

If Statements

- If statements are used to execute code if a condition is true
- If statements can be followed by an else statement
- If statements can be followed by an elif statement

```
x = 10
if x == 10:
    print("x is 10")
elif x == 20:
    print("x is 20")
else:
    print("x is not 10 or 20")
```

Table of Contents

What are Loops?

- Loops are used to repeat code
- Loops are used to iterate over a sequence
- Loops are used to execute code a specific number of times

For Loops

- For loops are used to iterate over a sequence
- For loops are used to execute code a specific number of times
- For loops can be used with lists, tuples, dictionaries, and sets

```
for x in range(10):
    print(x)
```

While Loops

- While loops are used to execute code as long as a condition is true
- While loops are used to execute code a specific number of times
- While loops can be used with lists, tuples, dictionaries, and sets

```
x = 0
while x < 10:
    print(x)
    x += 1
```