CS50 Python

Overview

There are many programming languages used in the world of computer science. Many of these languages can accomplish equivalent tasks, but programmers choose a language depending on the project they’re working on. Think of different programming languages as different kinds of shoes. Shoes all share core functionality: protecting your feet from the ground. But while athletic shoes are great for running, they are much worse in the cold than a pair of warm boots. Similarly, while programming languages all share the same basic functionality, each emphasizes particular features that are optimal for specific uses.

Key Terms

- Python
- high-level
- low-level
- list
- tuple
- dict
- dynamically typed

Python

Python is a **high-level** programming language developed to be easy to learn, easy to read, and broadly applicable. By abstracting away **low-level** technical details like memory management, Python reads more similarly to a natural language than a low-level language like C. Python is also different from C in that it is interpreted at run time rather than compiled to machine code beforehand. This allows us to run a single command (for example, `python hello.py`) to run a program. Because of its convenience, Python is widely used and compatible with other technologies such as databases, graphical user interfaces (GUIs), and web programming.

Syntax

```
from cs50 import get_string
```

```
def main():
    print(“hello, world”)

if __name__ == “__main__”:
    main()
```

```
while True:
    print(“hello, world”)
```

```
for i in range(50):
    print(“hello, world”)
```

```
if x < y:
    print(“x is less than y”)
elif x > y:
    print(“x is greater than y”)
else:
    print(“x is equal to y”)
```

```
import sys

for s in sys.argv:
    print(s)
```

Built-in data types

list - an ordered and changeable collection of items (can be updated and length can be changed)

`>>> mylist = [“foo”, “bar”]` create a new list
`>>> mylist.append(“baz”)` append “baz” to mylist
`>>> mylist` show mylist
`[“foo”, “bar”, “baz”]` value of mylist

tuple - an ordered and unchangeable collection of items

`>>> mytuple = (“foo”, “bar”, “baz”)` create a new tuple
`>>> mytuple[0]` show the value at the 0 index of mytuple
`“foo”` the value at the 0 index of mytuple

dict (dictionary) - an unordered, changeable list of key value pairs in which the key is the index to access a value

`>>> fruitcolor = {“apple”: 3, “lime”: 10}` create a new dict
`>>> fruitcolor[“apple”]` show the value of “apple”
`“red”` value of “apple”

Things to Remember

- Tabs and new lines are used to denote the end of commands and functions, no semicolons here!
- Python uses colons similar to the way we use open curly braces ({) in C, but these should be on the same line as the code, not a line all by themselves.
- Python is a **dynamically typed** language, meaning it infers data types at the time of assignment. **+** acts as both arithmetic addition and string concatenation depending on the variable types.
- Python comes with more functions out of the box than C. Additionally, there are lots of libraries that exist for Python so it is typically a good idea to check if functions exist in Python’s many libraries before you implement them yourself.