Student's Name

Instructor's Name

Course Title and Number

Date

Python Dictionaries

The Python language provides four collection data types, of which dictionaries are a part. Dictionaries are similar to Lists in Python, given that they are collections and are mutable, meaning that they can be changed on the fly. The only difference between the two is that, while Lists' items are accessed using their index (position in the list), dictionary items are accessed using their keys (Sweigart 10-11). As such, dictionaries are defined as a series of key-value pairs with unique keys.

Dictionaries are said to be ordered, mutable, and do not allow duplicates. This means that loops can be used to iterate over keys or values, that they can grow or shrink as needed after creation, and that keys cannot be repeated in the dictionary, respectively. They are defined by enclosing a series of key-value pairs in curly braces. For instance, a dictionary defining a Student can be created as follows:

```
student = {
    "name": "James Sawyer",
    "age": 23,
    "course": "Bachelor of Engineering"
}
```

In the example above, the keys are "name", "age", and "course", with the corresponding values of "James Sawyer", 23, and "Bachelor of Engineering". Accessing these values can be done as follows:

```
print(student["name"])

print(student.get("name"))
```

As seen in the print statements above, the student's name can be accessed either by using the bracket notation and specifying the key (name) or by using the *get* function. Both methods are common, but the second method (using the get function) is preferred, as it prevents the Python-based program from crashing if the key is not defined. By using the get function, the return value will be either the value of the key or "None" if the key is missing.

To add a value to the dictionary, the key is specified in the square brackets and the value declared as shown below:

```
student["is_active"] = True
```

In a similar manner, an item is deleted by using the "del" keyword as shown below. Once deleted, the key is no longer accessible. The get function can be used as shown below, indicating that the key is not present.

```
del student["age"]

print(student.get("age")) # output None
```

In addition, dictionaries can be used in loops, as they are iterable. The keys() or values() functions can also be used to provide a list of items that can be iterated upon.

In conclusion, dictionaries store and manipulate data using key-value pairs. They provide an intuitive way of adding, retrieving, manipulating, and deleting data using keys. As such, they are a crucial data structure in the development of applications using Python.

Works Cited

Sweigart, Albert. “Dictionaries and Structuring Data.” *Automate the Boring Stuff with Python: Practical Programming for Total Beginners*, No Starch Press, San Francisco, CA, 2019, pp. 10–11.

“5. Data Structures.” *Python Documentation*, docs.python.org/3/tutorial/datastructures.html. Accessed 23 May 2023.

“HTML Introduction.” *Introduction to HTML*, www.w3schools.com/html/html_intro.asp. Accessed 22 May 2023.