Dictionaries in Python

Student's Name

Department, Institutional Affiliation

Course Number and Name

Instructor's Name

Due Date

Dictionaries in Python

A dictionary in Python is a collection of key-value pairs, where each key is unique. According to Simplilearn (2022), dictionaries are mutable and unordered, meaning that elements can be added, removed, or updated, and their order can change. They are defined using curly brackets {} and can contain elements of different data types. As a result, they provide an easy way to retrieve data by calling a key.

Dictionaries provide a more flexible and organized way of storing data compared to lists or tuples. Unlike lists and tuples, where data is stored in a linear fashion, dictionaries allow you to associate keys with values. This enhances the accessibility and modifiability of the data and also improves its readability and comprehensibility. For example, in a list of student names and their corresponding grades, it can be challenging to find a particular student's grade, but with a dictionary, it's as simple as accessing the value associated with the student's name as the key (Moreno, 2021). In conclusion, dictionaries are indispensable tools for organizing and managing data in Python.

Creating a Dictionary

Dictionaries can be created as described in the screenshot below:

```
# Method 1: using curly braces {}
dict_1 = {}

# Method 2: using the dict() function
dict_2 = dict()

# Method 3: using a list of tuples
dict_3 = dict([("name", "John"), ("age", 25)])

# Method 4: using keyword arguments
dict_4 = dict(name="John", age=25)
```

Figure 1 Creating a Dictionary

Accessing Values

A key in a dictionary can be used as an index to access the corresponding value. However, if the specified key is not found in the dictionary, a key error is generated. To prevent this, the get() method can be employed, which returns a value of "None" if the key is not found in the dictionary (Moreno, 2021). This allows for a graceful fallback when the key is not found rather than raising an error.

Adding and Updating Values

To add a new element to a dictionary, the key is assigned a value using the assignment operator (=). If the key is already present in the dictionary, its value is updated. This makes it easy to add new data or update existing data in a dictionary, making dictionaries a versatile tool for organizing and managing information in Python.

Removing Values

The pop() method in Python allows for the removal of a specific element from a dictionary. However, if the key being removed is not found in the dictionary, a KeyError will be

raised. On the other hand, the popitem() function deletes and returns a random key-value pair from the dictionary.

Demonstration of Dictionaries Operations in Python

```
# Creating a dictionary
dict_4 = dict(name="Hellen", age=20)
# Getting the value of the key "name"
print("Printing the value of an element in a dictionary")
print(dict_4["name"])
# Getting the value of the key "address" which doesn't exist
print("\ntrying to access a value from a key that doesn't exist")
print(dict_4.get("address"))

# Adding an element to a dictionary
print("\nDictionary after adding a value into a dictionary")
dict_4["address"] = "123 Main St."
# printing dict
print(dict_4)

# Removing a value
print("\nDictionary after removing the value with the key 'age'")
dict_4.pop("age")
# Printing the dict again
print(dict_4)
```

Figure 2 Adding, Accessing, and Deleting Elements from a Dictionary

```
C:\Users\home\PycharmProjects\test\venv\Scripts\python.exe C:/Users/home/PycharmProj
Printing the value of an element in a dictionary
Hellen

trying to access a value from a key that doesn't exist
None

Dictionary after adding a value into a dictionary
{'name': 'Hellen', 'age': 20, 'address': '123 Main St.'}

Dictionary after removing the value with the key 'age'
{'name': 'Hellen', 'address': '123 Main St.'}

Process finished with exit code 0
```

Figure 3 The Output

In conclusion, dictionaries are an important data structure in Python and are used to store key-value pairs. They are mutable, unordered, and can contain elements of different data types. They are used to store data in an organized manner, making it easier to access and update that data. Dictionaries can be created in multiple ways, and values can be accessed, added, updated, and removed using different methods.

References

Moreno, A. I. (2021, July 1). *15 things you should know about dictionaries in Python.* Medium. https://towardsdatascience.com/15-things-you-should-know-about-dictionaries-in-python-44c55e75405c

Simplilearn. (2022, August 1). *What is a dictionary in Python?* Simplilearn.com. https://www.simplilearn.com/dictionary-in-python-article