**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  
**Topic 28 - Dictionaries: Versatile Keys and Values**  
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Flexible Key-Value Pairs in Python Dictionaries**

In previous examples, we created dictionaries with both keys and values as **strings**. For instance:

python

Copy code

customer\_29876 = {

"first name": "David",

"last name": "Elliott",

"address": "4803 Wellesley St."

}

Here, each **key** is a string (e.g., "first name") and each **value** is also a string (e.g., "David"). However, Python dictionaries allow flexibility: keys and values can be **different types**, such as **numbers** or **mixed types**.

**Keys as Numbers**

Keys don’t have to be strings; they can also be **numbers**. For example:

python

Copy code

rankings = {

5: "Finland",

2: "Norway",

3: "Sweden",

7: "Iceland",

}

In this dictionary, each **key** is a number, while each **value** is a string.

**Accessing a Value with a Number Key:**

To retrieve Norway’s ranking:

python

Copy code

second\_ranking\_country = rankings[2]

The value for rankings[2] is "Norway".

**Values as Numbers**

Values in a dictionary can also be **numbers**:

python

Copy code

country\_ranks\_so\_far = {

"Finland": 5,

"Norway": 2,

"Sweden": 3,

"Iceland": 7,

}

In this example, each **key** is a string (country name), and each **value** is a number (ranking).

**Accessing a Value with a String Key:**

To retrieve Norway’s rank:

python

Copy code

norway\_ranking = country\_ranks\_so\_far["Norway"]

The value for country\_ranks\_so\_far["Norway"] is 2.

**Mixing Data Types in Keys and Values**

Python dictionaries allow you to **mix and match** data types for both keys and values. Here’s an example:

python

Copy code

things\_to\_remember = {

0: "the lowest number",

"a dozen": 12,

"snake eyes": "a pair of ones",

13: "a baker's dozen",

}

This dictionary has:

* Key 0 with value "the lowest number"
* Key "a dozen" with value 12
* Key "snake eyes" with value "a pair of ones"
* Key 13 with value "a baker's dozen"

**Readability in Longer Dictionaries**

When defining a dictionary with multiple key-value pairs, you can improve **readability** by placing each pair on a separate line. Also, adding a comma after the last pair helps avoid errors if you add more pairs later.

**Example:**

python

Copy code

things\_to\_remember = {

0: "the lowest number",

"a dozen": 12,

"snake eyes": "a pair of ones",

13: "a baker's dozen",

}

By using different types for keys and values, Python dictionaries can store various kinds of information flexibly and efficiently.