Python Quick Sheet

From basic syntax to a small, checkable script | Studio01 Learning Team

Use this sheet as a reference. Run each example in a small file, change one thing at a time, and check the result. It does not promise a fixed learning timeline or outcome.

1. Values and variables

A variable gives a value a name. Strings hold text; numbers hold quantities; booleans hold True or False.

```
name = "Mina"
minutes = 20
ready = minutes > 0
print(f"{name}: {minutes} minutes")
```

2. Conditions and loops

Use `if` to choose a path and `for` to repeat an action over items.

```
scores = [72, 88, 91]
for score in scores:
    if score >= 80:
        print("review", score)
```

3. Lists and dictionaries

A list keeps an ordered sequence. A dictionary connects keys to values.

```
topics = ["strings", "lists", "files"]
minutes_by_topic = {"strings": 15, "lists": 20}
print(minutes_by_topic[topics[0]])
```

4. Functions

Put a repeated task behind a descriptive function name. Return a result so you can test it.

```
def word_count(text):
    return len(text.split())

assert word_count("learn by doing") == 3
```

Python 基礎クイックシート

基本文法から、小さく確かめられるスクリプトまで | Studio01 学習チーム

このシートは学習用の参考資料です。 小さなファイルで例を動かし、一度に一つずつ変更して結果を確認してください。習得期間や学習成果を約束するものではありません。

1. 値と変数

変数は値に名前を付けます。文字列はテキスト、数値は量、真偽値は True / False を表します。

```
name = "Mina"
minutes = 20
ready = minutes > 0
print(f"{name}: {minutes} minutes")
```

2. 条件分岐と繰り返し

`if` で処理を選び、`for` で複数の項目に同じ処理を行います。

```
scores = [72, 88, 91]
for score in scores:
    if score >= 80:
        print("review", score)
```

3. リストと辞書

リストは順序のある項目を並べ、辞書はキーと値を結び付けます。

```
topics = ["strings", "lists", "files"]
minutes_by_topic = {"strings": 15, "lists": 20}
print(minutes_by_topic[topics[0]])
```

4. 関数

繰り返し使う処理には内容が分かる関数名を付けます。結果を返すとテストしやすくなります。

```
def word_count(text):
    return len(text.split())

assert word_count("learn by doing") == 3
```

5. ファイルを読む

文字コードを明示すると、テキストの読み込みを分かりやすくできます。存在しないファイルなども試してください。

```
from pathlib import Path

path = Path("notes.txt")
text = path.read_text(encoding="utf-8")
print(text)
```

6. 次に試すこと

入力、期待する出力、小さな例を先に決めます。通常の入力、空の入力、想定外の入力を試し、実行結果をメモしましょう。詳しい順序は 無料Pythonロードマップ と 用語集 を参照してください。