PYTHON BASICS – STUDY NOTES

==========

---

1. BUILT-IN FUNCTIONS

Built-in functions are available everywhere in Python.
They can be used without importing anything.

print()
Displays text in the console.

Example:
print("Hello")

input()
Gets input from the user.
Always returns a string (str).

Example:
name = input("Your name: ")

type()
Returns the type of a value.

Example:
type(5)   -> int
type("hello") -> str
len()
Returns the length of a string or collection.

Example:

```
len("Python")      -> 6
len([1, 2, 3])     -> 3
```

---

2. TYPE CONVERSION

Used to convert one data type into another.

```
int("5")       -> 5
float("3.14")  -> 3.14
str(10)        -> "10"
bool(0)        -> False
bool(1)        -> True
```

Important:
input() always returns a string.

---

3. NUMBERS AND MATH

```
abs(-7)             -> 7
round(3.14159, 2)   -> 3.14
pow(2, 3)           -> 8
divmod(10, 3)       -> (3, 1)
```

---

4. STRINGS

Strings represent text.

Example:
text = "python"

Common string methods:

lower()
"text".lower() -> "text"

upper()
"text".upper() -> "TEXT"

title()
"anna nowak".title() -> "Anna Nowak"

strip()
" hi ".strip() -> "hi"

split()
"Anna Nowak".split(" ")
Result:
["Anna", "Nowak"]

join()
", ".join(["Anna", "Nowak"])
Result:
Anna, Nowak

---

5. LISTS

Lists store multiple values.

Example:
names = ["Anna", "Jan", "Ola"]

Accessing elements:
names[0] -> "Anna"

Length:
len(names)

Common list methods:
names.append("Kasia")
names.remove("Jan")
names.sort()

---

6. TUPLES

Tuples are like lists but cannot be changed.
Example:
coords = (10, 20)

Used when data should stay constant.

---

7. FOR LOOPS

Used to repeat actions.

```
Basic loop:
for word in words:
    print(word)

Loop with index:
for i in range(len(words)):
    print(i, words[i])

--------------------------------------------------
8. RANGE()
--------------------------------------------------

range(5)          -> 0 to 4
range(1, 6)       -> 1 to 5
range(0, 10, 2)   -> every second number



--------------------------------------------------
9. CONDITIONS
--------------------------------------------------

if condition:
    code
elif condition:
    code
else:
    code

Example:
if length > 5:
    print("Long word")
elif length == 5:
```

```
    print("Exactly five")
else:
    print("Short word")
```

---

10. MAX() AND MIN()

```
max([3, 7, 2]) -> 7
min([3, 7, 2]) -> 2

With words:
max(len(word) for word in words)
```

---

11. LIST COMPREHENSION

```
Shorter version of a for loop.

Example:
long_words = [w for w in words if len(w) > 5]

Equivalent to:
long_words = []
for w in words:
    if len(w) > 5:
        long_words.append(w)
```

---

12. BUILT-IN FUNCTIONS VS METHODS

Built-in functions:
Used like this:
function(value)

Examples:
len(text)
type(text)
max(numbers)

Methods:
Belong to an object and use a dot.

Examples:
text.lower()
text.split()
names.append()

Rule:
Function → outside
Method → after the dot

---

13. DICTIONARIES

Dictionaries store data in key–value pairs.

Structure:
key : value

Example:

```
person = {
    "name": "Anna",
    "age": 25,
    "city": "Gdansk"
}
```

Accessing values:

```
person["name"]     -> "Anna"
person["age"]      -> 25
```

Keys must be unique.
Values can repeat.

BASIC OPERATIONS

Add or update a value:

```
person["age"] = 26
```

Add new key:

```
person["job"] = "designer"
```

Remove a key:

```
del person["city"]
```

COMMON DICTIONARY METHODS

---

keys()
Returns all keys.

```
person.keys()
```

values()
Returns all values.

```
person.values()
```

items()
Returns key–value pairs.

```
person.items()
```

LOOPING THROUGH A DICTIONARY

Loop through keys:

```
for key in person:
    print(key)
```

Loop through values:

```
for value in person.values():
    print(value)
```

Loop through both:

```
for key, value in person.items():
    print(key, value)
```

CHECKING IF A KEY EXISTS

```
-------------------------------------------------

if "name" in person:
    print("Name exists")



-------------------------------------------------
NESTED DICTIONARIES
-------------------------------------------------

students = {
    "Anna": {"age": 20, "grade": "A"},
    "Jan": {"age": 22, "grade": "B"}
}

Access:
students["Anna"]["grade"]   -> "A"



-------------------------------------------------
WHEN TO USE A DICTIONARY
-------------------------------------------------

Use a dictionary when:
- data has meaning (labels)
- values belong to names
- you want fast access by key

Example:
user["email"]
settings["theme"]
product["price"]
```