10 головних базових блоків Python — шпаргалка

Це короткий конспект для щоденної практики. Приклади мінімальні, але робочі.

1 Змінні та типи даних

Змінна — це ім’я для значення. Основні типи: int, float, str, bool, а також колекції: list, tuple, set, dict.

```
name = "Оля"
age = 25
is_active = True
```

- Перевір тип: use type(x).
- Перетворення типів: int('3'), float('3.14'), str(123).
- None означає «немає значення».

2 Умовні оператори (if / elif / else)

Дають програмі змогу приймати рішення на основі істинності умови.

```
if age >= 18:
print("Повнолітній")
elif age >= 16:
print("Майже повнолітній")
else:
print("Неповнолітній")
```

- Порівняння: ==, !=, <, <=, >, >=.
- Логіка: and, or, not.

3 Цикли (for, while)

Повторюють дію. for перебирає послідовності; while працює доти, доки умова істинна.

```
for i in range(5):
print(i)

x = 0
while x < 3:
print(x)
x += 1
```

- Керування циклом: break — вийти, continue — пропустити крок.
- range(a, b) дає числа від a до b-1.

4 Функції (def)

Групують код у багаторазові блоки. Можуть повертати значення через return.

```
def greet(name="гість"):
return f"Привіт, {name}!"

print(greet("Іра"))
print(greet())
```

- Аргументи за замовчуванням задають типові значення.
- Докстрінг: перший рядок у функції у потрійних лапках — опис призначення.

5 Списки та перебір

Списки зберігають впорядковані елементи. Можна змінювати вміст.

```
nums = [1, 2, 3]
nums.append(4) # додати в кінець
nums[0] = 10 # змінити елемент
for n in nums:
print(n)
```

- Корисні методи: append, extend, insert, remove, pop, sort, reverse.
- Срізи: nums[1:3], nums[-1].

6 Словники (dict)

Пара «ключ — значення». Ключі зазвичай рядки.

```
user = {"name": "Макс", "age": 30}
print(user["name"])
user["city"] = "Львів"
```

- Методи: keys(), values(), items(), get('key', default).
- Перебір: for k, v in user.items(): ...

7 Обробка помилок (try / except)

Дозволяє реагувати на помилки без падіння програми.

```
try:
x = int("abc")
except ValueError as e:
print("Помилка перетворення:", e)
```

- finally виконується завжди; else — коли помилки не було.
- Власні помилки: raise ValueError('повідомлення').

8 Робота з файлами

with відкриває файл і автоматично закриває його.

```
# запис
with open("data.txt", "w", encoding="utf-8") as f:
f.write("Привіт, файл!")
```

```
# читання
with open("data.txt", "r", encoding="utf-8") as f:
text = f.read()
print(text)
```

- Режими: 'r' читання, 'w' перезапис, 'a' дозапис.
- Краще завжди вказувати encoding='utf-8'.

9 Імпорт модулів

Стандартна бібліотека покриває багато задач.

```
import math
from datetime import datetime, timedelta

print(math.sqrt(16))
print(datetime.now() + timedelta(days=1))
```

- Корисні модулі: os, sys, pathlib, json, random, re, itertools.
- pip встановлює сторонні бібліотеки: pip install requests

Спискові вирази (list comprehensions)

Швидке створення списків на основі інших колекцій.

```
nums = [1, 2, 3, 4, 5]
squares = [n**2 for n in nums]
evens = [n for n in nums if n % 2 == 0]
print(squares)
print(evens)
```

- Є також dict/set comprehensions: {x:x**2 for x in nums}.
- Комбінуйте з функціями sum, max, min для швидкої аналітики.

Швидкі підказки

• len(x) — довжина; type(x) — тип; print() — вивід.
• ' '.join(list_of_strings) — склеїти рядки; split() — розбити.
• enumerate(iterable) дає (індекс, значення) у циклі.
• zip(a, b) з’єднує послідовності поелементно.
• list(range(a, b, step)) — діапазон чисел.
• any/ all — чи є хоч одне True / чи всі True.
• Коментарі: # однорядковий, '''три лапки''' — багато рядків у докстрінгах.