对象创建

字面值：最直接的创建方式（42, [1,2,3], {"a":1}）

构造函数：int(), list(), dict() 等类型转换

推导式：[x for x ...], {k:v for ...}, {x for ...} 生成容器

运算符：+ 拼接列表，| 合并集合/字典(Python3.9+)

类型特征验证

type() 获取精确类型

isinstance() 检查类型（考虑继承关系）

id() 验证对象标识（内存地址）

运算符重载

数学运算符：+, //, % 等在不同类型中有不同实现

比较运算符：<, == 等根据类型定义比较逻辑

协议支持

可迭代协议：实现 __iter__() 方法

布尔转换：实现 __bool__() 或 __len__() 方法

下标访问：实现 __getitem__() 和 __setitem__()（可变对象）

常用方法

列表的 append()

字典的 get()

字符串的 upper()

异常处理

try/except 捕获预期异常

验证类型特有的限制（如元组不可变）

调试技巧

breakpoint() 进入调试模式

在报错前暂停程序检查变量状态

```
a = 42
b = int("3")
c = a + b
d = abs(-5)


assert type(a) is int  # type()返回对象的类型对象，此处验证a的类型标识
assert isinstance(b, int)  # isinstance检查对象是否属于某类，支持继承判断
assert id(a) != id(b)  # id()返回对象内存地址，不同对象地址不同
assert str(a) == "42"  # str()调用对象的__str__方法，返回字符串表示形式

assert 3 + 5 == 8      # 验证加法运算符重载
assert 10 // 3 == 3    # 验证整除运算符行为


assert 3 < 5


assert bool(0) is False
assert bool(42) is True


try:
    for _ in 42:       # 尝试迭代int对象，触发TypeError
        pass
except TypeError:
    print("int不可迭代")  # 捕获异常，验证int不可迭代特性

try:
    print(42[0])       # 尝试用[]操作符访问int，触发TypeError
except TypeError:
    print("int不支持索引")

assert (3).bit_length() == 2  # 调用int的bit_length方法，返回二进制位数
```

```
# 创建实例
lst1 = [1, 2, 3]
lst2 = [x*2 for x in range(3)]
lst3 = list("abc")
lst4 = lst1 + lst3
lst5 = lst1[::2]

# 验证属性
assert len(lst1) == 3
assert lst1[0] == 1

# 比较运算
assert [1, 2] < [1, 2, 3]

# 布尔转换
assert bool([]) is False

# 可迭代验证
for item in lst1:
    pass

# 方法演示
lst1.append(4)
assert lst1 == [1,2,3,4]
```

```python
d1 = {"a": 1, "b": 2}
d2 = {k: v*2 for k, v in d1.items()}
d3 = dict(a=1, b=2)
d4 = list(d1.items())[0]


# 布尔转换
assert bool({}) is False

# 可迭代验证（遍历键）
for key in d1:
    pass

# 方法演示
assert d1.get("a") == 1
```

```python
s = "hello"
s2 = str(3.14)
s3 = s.upper()
s4 = s + " world"

# 索引操作
assert s[0] == 'h'

# 比较运算
assert "apple" < "banana"
```

```python
def debug_example():
    x = 10
    y = "test"
    breakpoint()
    print(x + y)
```