```
print(user_name)
```

```
num = 5
text = "books"
print(num + text)
```

```
x=int("hello")
```

```
x = 5 / 0
```

```
x=open("nonexistent_file.txt")
```

```
lst = [1, 2, 3]
print(lst[5])
```

```
d = {"a": 1}
print(d["b"])
```

```
x = 5
x.append(2)
```

```
len(100)
```

```
from math import cube
```

```
import fakepackage
```

```
from datetime import timez
```

```
def greet():
print("Hi")
```

```
while True
    pass
```

What can I help you build?

```
class Person: pass
p = Person()
print(p.name)
```

```
class A:
    def __init__(self):
        self.x = 1
a = A()
del a.x
print(a.x)
```

```
class Car:
    print(engine)
```

```
class A:
    def __str__(self):
        return 5
print(str(A()))
```

```
my_list = [1, 2, 3]
my_list["1"]
```

```
d = {"a": 5}
print(d["b"])
```

```
lst = [1, 2, 3]
for i in range(5):
    print(lst[i])
```

```
lst = ["1", "two", "3"]
nums = list(map(int, lst))
```

```
d = {"a": 1}
d += {"b": 2}
```

```
d = {"x": 5}
print(d.get("xyz"))
```

```python
s = "abc"
print(s[10])
```

```python
class A:
    print(x)
```

```python
f = lambda x: x +
```

```python
a = 1
a += "2"
```

```python
"hello".remove("e")
```

```python
data = [[1, 2], [3, 4]]
print(data[2][0])
```

```python
for i in range(5, -1, -1):
    print(10 / i)
```

```python
lst = [1, 2, 3]
lst.remove(5)
```

```python
x = 5
print(x.upper())
```

```python
from mymodule import myfunc
```

```python
try:
finally:
    print("done")
```

```python
x = None
x.append(1)
```

```python
f = open("missing.txt")
```

```python
def f():
    print(a)
    a = 5
f()
```

```python
if 5 in 12345:
    print("yes")
```

```python
x = 123
x.split()
```

```python
a, b = [1, 2, 3]
```

```python
lst = [1, 2, 3]
lst.update([4, 5])
```

```python
5 = x
```

```python
lst = [10, 20]
print(lst[-3])
```

```python
lst = [1, "two", 3]
lst.sort()
```

```python
data = {"x": 1}
data.pop("y")
```

```python
print(int("19"))
```

```python
a = 10
b = 0
print(a // b)
```

```python
class Test:
    def show():
        print(self.x)
t = Test()
t.show()
```

```python
x = "123"
x.push("4")
```

```python
lst = [1, 2, 3]
lst.remove(2)
print(lst[2])
```

```python
print(10 / 0)
```

```python
with open("no_such_file.txt") as f:
    print(f.read())
```

```python
f = open("ghost.txt")
```

```python
float("abc")
```

```python
["a", "b", "c"].index("z")
```

```python
a, b = [1, 2, 3]
```

```python
[1, 2] > "a"
```

```python
print("a" * "b")
```

```python
s = {1, 2}
print(s[0])
```

```python
len(5)
```

```python
lst = [1, 2]
print(lst[-3])
```

```python
print(x)
x = 5
```

```
str = "hello"
str()


x = 5
x.upper()


if True:


int("101")


"hello".push("!")


class A: pass
a = A()
print(a.name)


class Test:
    def greet():
        print(self.name)
Test().greet()


class A:
    def __init__(self): pass
A.name


f = lambda x:
    x+1


from math import cs


try:
    x = int("abc")
except ValueError:
    print("ValueError")
except TypeError:
    print("TypeError")
```

```python
try:
    open("no.txt")
except Exception as e:
    print("Error:", e)


try:
    x = 10 / 2
except ZeroDivisionError:
    print("ZeroDivisionError")
else:
    print("Division successful:", x)


try:
    num = int(input("Enter number: "))
except ValueError:
    print("Invalid input")
else:
    print("You entered:", num)


try:
    result = 10 / 0
except ZeroDivisionError:
    print("ZeroDivisionError")
else:
    print("Success")


try:
    lst = [1, 2, 3]
    print(lst[1])
except IndexError:
    print("Index out of range")
else:
    print("Element found")


try:
    f = open("file.txt")
except FileNotFoundError:
    print("File not found")
else:
    print("File opened successfully")
```

```
try:
    x = 5 / 0
except ZeroDivisionError:
    print("Handled")
finally:
    print("Finally executed")


try:
    print("Hello")
except:
    print("Error")
finally:
    print("Always runs")


try:
    f = open("demo.txt", "r")
except FileNotFoundError:
    print("File not found")
finally:
    print("Done")


try:
    x = int("abc")
except:
    print("Error in conversion")
finally:
    print("Finally block")


try:
    x = 5 / 0
except:
    print("Caught error")
finally:
    print("Final cleanup")


x = "hello"
if not x.isdigit():
    raise ValueError("Not a number")


def check_age(age):
    if age < 0:
        raise ValueError("Age cannot be negative")
check_age(-5)


def add(a, b):
    if not isinstance(a, int):
        raise TypeError("Only integers allowed")
add("2", 3)


try:
    raise ZeroDivisionError("Custom error")
except ZeroDivisionError as e:
    print("Caught:", e)


try:
    raise ValueError("Original error")
except ValueError as e:
    raise RuntimeError("New context") from e


try:
    num = int(input("Enter number: "))
except ValueError:
    print("Not a valid number")


try:
    f = open("demo.txt", "w")
    f.write("Hello")
finally:
    f.close()
    print("File closed")


try:
    lst = [1]
    print(lst[5])
except IndexError:
    print("IndexError handled")


try:
    num = float("abc")
except ValueError:
    print("Cannot convert")
```

```python
def validate_email(email):
    if "@" not in email:
        raise ValueError("Invalid email")
validate_email("test.com")


while True:
    try:
        x = int(input("Enter integer: "))
        break
    except ValueError:
        print("Try again")


try:
    num = int("5")
except:
    print("Error")
else:
    print("Valid:", num)
finally:
    print("Execution complete")


try:
    f = open("sample.txt", "w")
    f.write("Hi")
finally:
    f.close()


try:
    print("Trying something")
finally:
    print("Always executes")


try:
    print(5 / 0)
except (ZeroDivisionError, TypeError):
    print("Math error!")


def check(n):
    if n < 0:
        raise Exception("Negative number!")
check(-5)


def divide(a, b):
    try:
        result = a / b
    except ZeroDivisionError:
        print("Cannot divide by zero.")
    else:
        print(f"Result is {result}")
    finally:
        print("Division operation complete.")

divide(10, 5)



def read_file(filename):
    try:
        with open(filename, 'r') as f:
            data = f.read()
    except FileNotFoundError:
        print("File not found.")
    else:
        print("File content:", data)
    finally:
        print("File read attempt finished.")

read_file("test.txt")



def check_age(age):
    try:
        if age < 0:
            raise ValueError("Negative age not allowed")
    except ValueError as e:
        print("Error:", e)
    else:
        print("Age is valid:", age)
    finally:
        print("Check completed.")

check_age(-3)
```

```
def find_key(d, key):
    try:
        value = d[key]
    except KeyError:
        print("Key not found in dictionary.")
    else:
        print(f"Value found: {value}")
    finally:
        print("Lookup finished.")

find_key({"a": 1}, "b")
```

```
try:
    try:
        x = int("xyz")
    except ValueError:
        print("Inner block caught ValueError")
except:
    print("Outer block caught an error")
finally:
    print("Nested blocks done")
```

```
lst = [10, 20, 30]
try:
    val = int(input("Index to access: "))
    print("Value:", lst[val])
except (ValueError, IndexError):
    print("Invalid index or type")
finally:
    print("End of list operation")
```

```
def safe_divide(x, y):
    try:
        result = x / y
    except ZeroDivisionError:
        print("Can't divide by zero")
    else:
        print("Division result:", result)
    finally:
        print("Done dividing")
```

```
safe_divide(10, 2)
```

Start coding or generate with AI.