

**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Факультет Информационных технологий  
Кафедра Информатики и информационных технологий**

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ (ПРАКТИЧЕСКАЯ РАБОТА) № 4**

**Дисциплина: Backend-разработка**

**Тема: Исключения**

**Выполнил(а): студент(ка) группы 221-373**

**Ким Вячеслав Михайлович**

(Фамилия И.О.)

**Дата, подпись** \_\_\_\_\_

(Дата)

(Подпись)

**Проверил:** \_\_\_\_\_

(Фамилия И.О., степень, звание)

(Оценка)

**Дата, подпись** \_\_\_\_\_

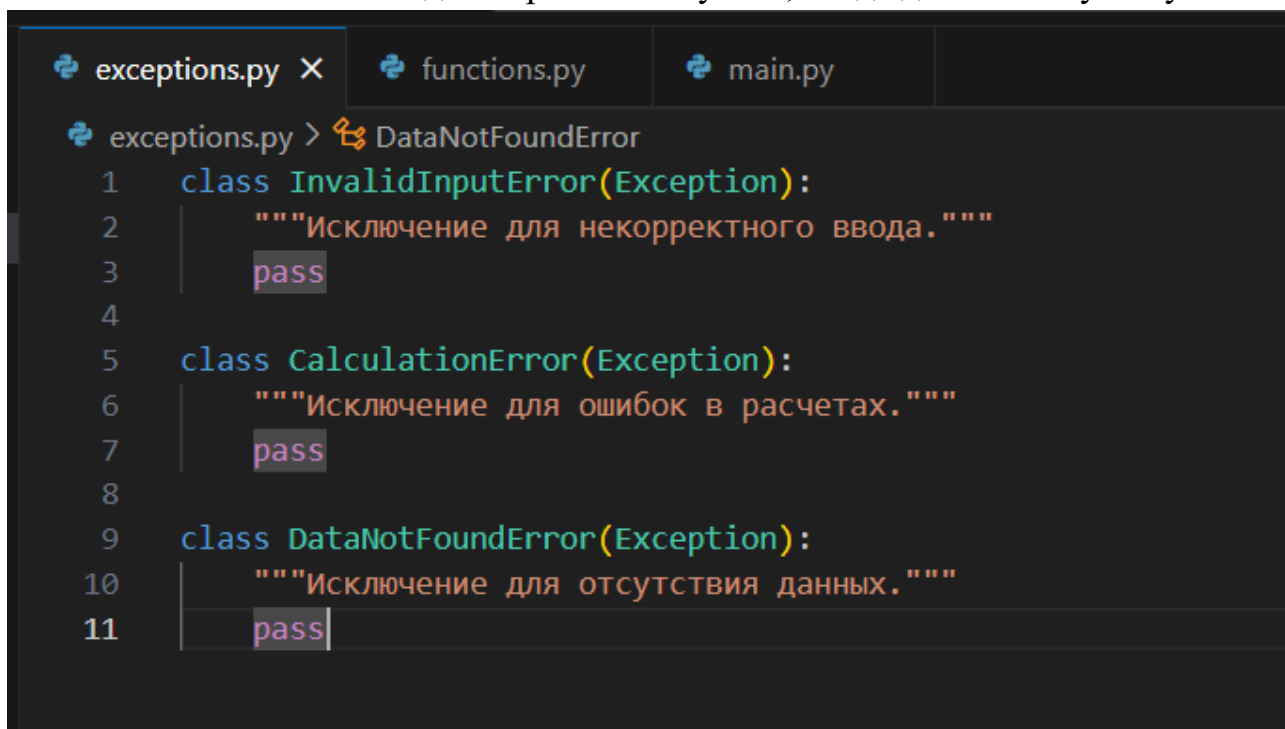
(Дата)

(Подпись)

**Москва**

Создадим файл exceptions.py, в котором определим три пользовательских исключения.

- `InvalidInputError`: для обработки некорректного ввода.
- `CalculationError`: для обработки ошибок, возникающих в процессе расчетов.
- `DataNotFoundError`: для обработки случаев, когда данные отсутствуют.



```
exceptions.py X  functions.py  main.py
exceptions.py > DataNotFoundError
1  class InvalidInputError(Exception):
2      """Исключение для некорректного ввода."""
3      pass
4
5  class CalculationError(Exception):
6      """Исключение для ошибок в расчетах."""
7      pass
8
9  class DataNotFoundError(Exception):
10     """Исключение для отсутствия данных."""
11     pass
```

Создадим файл functions.py, в котором реализуем функции, выбрасывающие исключения.

Создадим функции, которые принимают параметры и выбрасывают исключения при определенных условиях:

- `divide(a, b)`: выбрасывает `ZeroDivisionError`, если второй параметр равен нулю.
- `square_root(x)`: выбрасывает `ValueError`, если входное значение отрицательное.
- `calculate_average(numbers)`: выбрасывает `InvalidInputError`, если список пуст.
- `process_data(data)`: выбрасывает `TypeError`, если входные данные не являются списком, и `DataNotFoundError`, если список пуст.
- `risky_calculation(x)`: выбрасывает `CalculationError`, если входное значение отрицательное, или `ZeroDivisionError`, если оно равно нулю.
- `generate_exception(value)`: выбрасывает `InvalidInputError` или `CalculationError` в зависимости от значения.

```
from exceptions import InvalidInputError, CalculationError, DataNotFoundError
```

```

def divide(a, b):
    """Деление двух чисел."""
    if b == 0:
        raise ZeroDivisionError("Деление на ноль.")
    return a / b

def square_root(x):
    """Квадратный корень."""
    if x < 0:
        raise ValueError("Нельзя извлекать квадратный корень из отрицательного числа.")
    return x ** 0.5

def calculate_average(numbers):
    """Среднее значение."""
    try:
        if not numbers:
            raise InvalidInputError("Список не должен быть пустым.")
        return sum(numbers) / len(numbers)
    except Exception as e:
        print(f"Ошибка: {e}")

def process_data(data):
    """Обработка данных."""
    try:
        if not isinstance(data, list):
            raise TypeError("Ожидается список.")
        if len(data) == 0:
            raise DataNotFoundError("Данные отсутствуют.")
        return [x * 2 for x in data]
    except Exception as e:
        print(f"Обработка данных завершилась с ошибкой: {e}")
    finally:
        print("Завершение обработки данных.")

def risky_calculation(x):
    """Рискованные вычисления."""
    try:
        if x < 0:
            raise CalculationError("Некорректный ввод для вычислений.")
        return 100 / x
    except CalculationError as ce:
        print(f"Ошибка вычисления: {ce}")
    except ZeroDivisionError as zde:
        print(f"Ошибка деления: {zde}")
    except Exception as e:
        print(f"Общая ошибка: {e}")
    finally:

```

```

        print("Завершение рискованных вычислений.")

def generate_exception(value):
    """Генерация исключений."""
    try:
        if value < 0:
            raise InvalidInputError("Отрицательное значение.")
        if value == 0:
            raise CalculationError("Нулевое значение.")
    except InvalidInputError as ie:
        print(f"Обработка InvalidInputError: {ie}")
    except CalculationError as ce:
        print(f"Обработка CalculationError: {ce}")
    finally:
        print("Завершение генерации исключений.")

```

Создадим файл main.py, который будет вызывать все созданные функции.

```

# main.py > ...
1  from functions import divide, square_root, calculate_average, process_data, risky_calculation, generate_exception
2
3  def main():
4      try:
5          print(divide(10, 2))
6          print(square_root(16))
7          print(calculate_average([1, 2, 3, 4, 5]))
8          process_data([1, 2, 3])
9          risky_calculation(-10)
10         generate_exception(-5)
11     except Exception as e:
12         print(f"Произошла ошибка в главной функции: {e}")
13
14 if __name__ == "__main__":
15     main()
16

```

Протестируем работу программы

```

PS C:\labs\lab4> python main.py
5.0
4.0
3.0
Завершение обработки данных.
Ошибка вычисления: Некорректный ввод для вычислений.
Завершение рискованных вычислений.
Обработка InvalidInputError: Отрицательное значение.
Завершение генерации исключений.
PS C:\labs\lab4>

```