Обработка ошибок и исключений

Что это

При программировании на Python мы можем столкнуться с двумя типами ошибок. Первый тип представляют синтаксические ошибки (syntax error). Они появляются в результате нарушения синтаксиса языка программирования при написании исходного кода.

Что это

Второй тип ошибок представляют ошибки выполнения (runtime error). Они появляются в уже в процессе выполнения программы. Подобные ошибки еще называются исключениями.

```
string = "5"
number = int(string)
print(number)
```

```
string = "hello"
number = int(string)
print(number)
```

try..except

Конструкция try..except имеет следующее формальное определение:

```
try:
    инструкции
except [Тип_исключения]:
    инструкции
```

try..except

Весь основной код, в котором потенциально может возникнуть исключение, помещается после ключевого слова try. Если в этом коде генерируется исключение, то работа кода в блоке try прерывается, и выполнение переходит в блок except.

try..except

После ключевого слова except опционально можно указать, какое исключение будет обрабатываться (например, ValueError или KeyError).

```
try:
    number = int(input("Введите число: "))
    print("Введенное число:", number)
except:
    print("Преобразование прошло неудачно")
print("Завершение программы")
```

try..except

Вводим строку:

```
Введите число: hello
Преобразование прошло неудачно
Завершение программы
```

Вводим правильное число:

```
Введите число: 22
Введенное число: 22
Завершение программы
```

Блок finally

Отличительной особенностью этого блока является то, что он выполняется вне зависимости, было ли сгенерировано исключение:

```
try:
    number = int(input("Введите число: "))
    print("Введенное число:", number)
except:
    print("Преобразование прошло неудачно")
finally:
    print("Блок try завершил выполнение")
print("Завершение программы")
```

Встроенные типы исключений

Встроенные типы исключений

В примере выше обрабатывались сразу все исключения, которые могут возникнуть в коде. Однако мы можем конкретизировать тип обрабатываемого исключения, указав его после слова except:

```
try:
    number = int(input("Введите число: "))
    print("Введенное число:", number)
except ValueError:
    print("Преобразование прошло неудачно")
print("Завершение программы")
```

Типы исключений

IndexError: исключение возникает, если индекс при обращении к элементу коллекции находится вне допустимого диапазона

KeyError: возникает, если в словаре отсутствует ключ, по которому происходит обращение к элементу словаря.

OverflowError: возникает, если результат арифметической операции не может быть представлен текущим числовым типом (обычно типом float).

Типы исключений

IndexError: исключение возникает, если индекс при обращении к элементу коллекции находится вне допустимого диапазона

KeyError: возникает, если в словаре отсутствует ключ, по которому происходит обращение к элементу словаря.

OverflowError: возникает, если результат арифметической операции не может быть представлен текущим числовым типом (обычно типом float).

Типы исключений

RecursionError: возникает, если превышена допустимая глубина рекурсии.

TypeError: возникает, если операция или функция применяется к значению недопустимого типа.

ValueError: возникает, если операция или функция получают объект корректного типа с некорректным значением.

ZeroDivisionError: возникает при делении на ноль.

Типы исключений

NotImplementedError: тип исключения для указания, что какие-то методы класса не реализованы

ModuleNotFoundError: возникает при при невозможности найти модуль при его импорте директивой import

OSError: тип исключений, которые генерируются при возникновении ошибок системы (например, невозможно найти файл, память диска заполнена и т.д.)

try..except

```
try:
    number1 = int(input("Введите первое число: "))
    number2 = int(input("Введите второе число: "))
    print("Результат деления:", number1/number2)
except ValueError:
    print("Преобразование прошло неудачно")
except ZeroDivisionError:
    print("Попытка деления числа на ноль")
except BaseException:
    print("Общее исключение")
print("Завершение программы")
```

Генерация исключений

Иногда возникает необходимость вручную сгенерировать то или иное исключение. Для этого применяется оператор raise.

```
try:
    age = int(input("Введите возраст: "))
    if age > 110 or age < 1:
        raise Exception("Некорректный возраст")
    print("Ваш возраст:", age)
except ValueError:
    print("Введены некорректные данные")
except Exception as e:
    print(e)
print("Завершение программы")
```

Создание своих типов исключений

В языке Python мы не ограничены только встроенными типами исключений и можем, применяя наследование, при необходимости создавать свои типы исключений.

```python
class PersonAgeException(Exception):
    def __init__(self, age, minage, maxage):
        self.age = age
        self.minage = minage
        self.maxage = maxage

    def __str__(self):
        return f"Недопустимое значение: {self.age}. " \
               f"Возраст должен быть в диапазоне от {self.minage} до {self.maxage}"


class Person:
    def __init__(self, name, age):
        self.__name = name  # устанавливаем имя
        minage, maxage = 1, 110
        if minage < age < maxage:   # устанавливаем возраст, если передано корректное значение
            self.__age = age
        else:                       # иначе генерируем исключение
            raise PersonAgeException(age, minage, maxage)

    def display_info(self):
        print(f"Имя: {self.__name}  Возраст: {self.__age}")

try:
    tom = Person("Tom", 37)
    tom.display_info()  # Имя: Tom  Возраст: 37

    bob = Person("Bob", -23)
    bob.display_info()
except PersonAgeException as e:
    print(e)    # Недопустимое значение: -23. Возраст должен быть в диапазоне от 1 до 110
```

Задание

Задача 1: Калькулятор с проверкой ввода

Напишите программу, которая запрашивает у пользователя два числа и операцию (+, -, *, /), а затем выводит результат. Обработайте возможные исключения:

Если введены не числа — ValueError.

Если деление на ноль — ZeroDivisionError.

Если неверная операция — KeyError, или какое-то другое