Python

План занятия

- Обработка ошибок
- Функции
- Самостоятельная работа

Программы ломаются

```
s = int(some_value)
d = 3 / some_delta
l = len(some_var)
```

Обработка исключений - еще один способ управления логикой программы

Практика

Файл "tryexcept.py"

Функция в Python

```
def my_function(input_var1, input_var2):
    print(input_var1, input_var2)
    return input_var1 + input_var2
```

Где:
def - ключевое слово для объявления функции
my_function - имя функции
input_var1, input_var2 - входные параметры
return - ключевое слово, чтобы вернуть значение туда, где функция была вызвана

«Зачем нам нужны функции?»

–Не знающие люди

Практика

course2/functions.py

Еще больше концепций про функции

Функции могу принимать

- Нулевое количество аргументов
- Позиционные (обычные) аргументы
- Аргументы с параметром по-умолчанию (keyword)
- Любое количество позиционных аргументов
- Любое количество keyword-аргументов

Практика

Файл "functions/args_and_kwargs.py"

Полезные концепции из функционального программирования

- map - выполняет функцию для каждого объекта из массива входных параметров

- filter - оставляет из массива только что удовлетворяет условию

- reduce - выполняет функцию для каждого объекта из массива входных параметров, оперируя остатком

Практика

Файл "functions/fp.py"

«Функция - тоже объект».

–Знающие люди

Практика

Файл "functions/functions_are_objects.py"

Замыкание - возможность функции использовать чужие переменные

Практика

Файл "functions/enclosing.py"