# Инвариантная самостоятельная работа №1

**Задание 1.1**

**Постановка задачи:**

1.1. Разработать программу с реализацией функции для считывания json-данных из файла и вывод их в табличном виде на экран. Реализовать базовый синтаксис для обработки исключений (try .. except)

**Решение:**

[**https://repl.it/@AnastasiaTs/Isr11**](https://repl.it/@AnastasiaTs/Isr11)

**Код программы:**

"""

Разработать программу с реализацией функции для считывания

json-данных из файла и вывод их в табличном виде на экран.

Реализовать базовый синтаксис для обработки исключений (try .. except)

"""

import json

try:

with open('MOCKDATAS.json') as f:

data\_dict = json.load(f)

except FileNotFoundError as e:

print(e)

try:

with open('MOCKDATA.json') as f:

data\_dict = json.load(r)

except NameError as e:

print(e)

try:

for el in range(len(data\_dict)):

temp = data\_dit[el]

res = sstr.format(\*\*temp)

table.append(res)

except NameError as e:

print(e)

def json\_table(fl):

with open(fl) as f:

data\_dict = json.load(f)

table = []

sstr = '| {id:^3} | {first\_name:^10} | {last\_name:^15} | {email:^30} | {gender:^6} | {ip\_address:^16} |'

t\_caption = '| {:^3} | {:^10} | {:^15} | {:^30} | {:^6} | {:^16} |'.format('id','first\_name','last\_name','email','gender','ip\_address')

roof = '-'\*len(t\_caption)

table.append(roof)

table.append(t\_caption)

table.append(roof)

for el in range(len(data\_dict)):

temp = data\_dict[el]

res = sstr.format(\*\*temp)

table.append(res)

table.append(roof)

return table

def main():

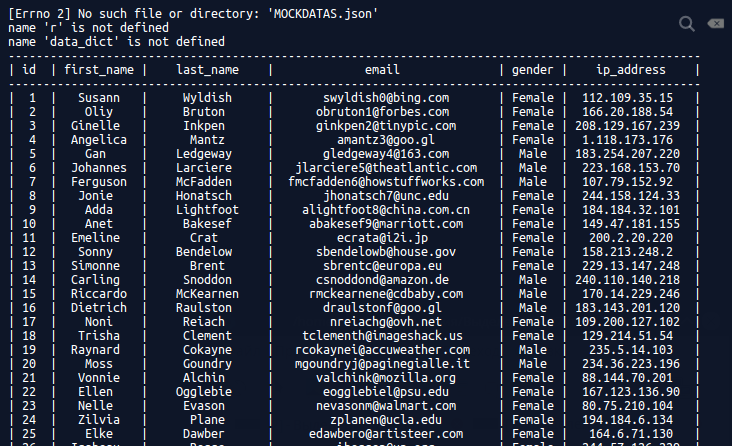
a = json\_table('MOCKDATA.json')

for el in a:

print(el)

main()

**Результат работы программы:**

****

**Задание 1.2**

**Постановка задачи:**

1.2. Дополнение программы для считывания данных проверкой утверждений или высказываний (assert). Создание отдельного блока для такой проверки (с помощью \_\_name\_\_) и скрипта командной строки для запуска этих проверок.

**Решение:**

[**https://repl.it/@AnastasiaTs/Isr12**](https://repl.it/@AnastasiaTs/Isr12)

**Код программы:**

"""

1.2. Дополнение программы для считывания данных проверкой

утверждений или высказываний (assert). Создание отдельного

блока для такой проверки (с помощью \_\_name\_\_) и скрипта командной

строки для запуска этих проверок.

"""

import json

def json\_table(fl):

with open(fl) as f:

data\_dict = json.load(f)

table = []

sstr = '| {id:^3} | {first\_name:^10} | {last\_name:^15} | {email:^30} | {gender:^6} | {ip\_address:^16} |'

t\_caption = '| {:^3} | {:^10} | {:^15} | {:^30} | {:^6} | {:^16} |'.format('id','first\_name','last\_name','email','gender','ip\_address')

roof = '-'\*len(t\_caption)

table.append(roof)

table.append(t\_caption)

table.append(roof)

for el in range(len(data\_dict)):

temp = data\_dict[el]

res = sstr.format(\*\*temp)

table.append(res)

table.append(roof)

return(table)

def main():

a = json\_table('MOCKDATA.json')

print(type(a))

for el in a:

print(el)

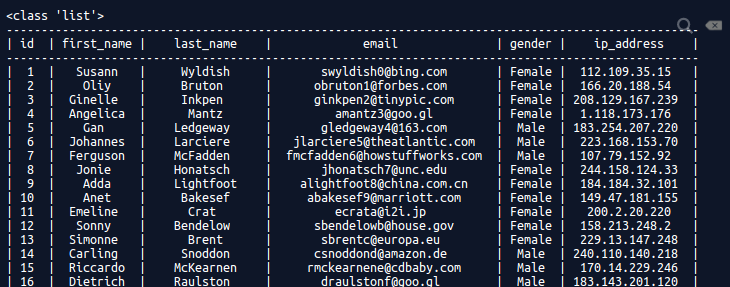
if \_\_name\_\_ == "\_\_main\_\_":

assert type(json\_table('MOCKDATA.json')) == list, "Типы не совпадают"

assert type(json\_table('MOCKDATA.json')) != tuple

main()

**Результат работы:**



**Задание 1.3**

**Постановка задачи:**

1.3. Дополнение программы для считывания данных с использованием менеджера контекстов и реализации расширенного синтаксиса для обработки исключений.

**Решение:**

[**https://repl.it/@AnastasiaTs/Isr13**](https://repl.it/@AnastasiaTs/Isr13)

**Код программы:**

"""

Дополнение программы для считывания данных с использованием

менеджера контекстов и реализации расширенного синтаксиса для

обработки исключений.

"""

import json

try:

with open('MOCKDATAS.json') as f:

data\_dict = json.load(f)

except FileNotFoundError as e:

print(e)

try:

with open('MOCKDATA.json') as f:

data\_dict = json.load(r)

except NameError as e:

print(e)

try:

for el in range(len(data\_dict)):

temp = data\_dit[el]

res = sstr.format(\*\*temp)

table.append(res)

except NameError as e:

print(e)

else:

print("Congratulations, your program work right!")

finally:

print("You can't make an omelette without breaking a few eggs.")

def json\_table(fl):

with open(fl) as f:

data\_dict = json.load(f)

table = []

sstr = '| {id:^3} | {first\_name:^10} | {last\_name:^15} | {email:^30} | {gender:^6} | {ip\_address:^16} |'

t\_caption = '| {:^3} | {:^10} | {:^15} | {:^30} | {:^6} | {:^16} |'.format('id','first\_name','last\_name','email','gender','ip\_address')

roof = '-'\*len(t\_caption)

table.append(roof)

table.append(t\_caption)

table.append(roof)

for el in range(len(data\_dict)):

temp = data\_dict[el]

res = sstr.format(\*\*temp)

table.append(res)

table.append(roof)

return table

def main():

a = json\_table('MOCKDATA.json')

for el in a:

print(el)

main()

**Результат работы:**

