Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектронике»

Кафедра инженерной психологии и эргономики

Отчет по

ЛАБОРАТОРНОЙ РАБОТЕ №1

Использование IF. Простые и вложенные циклы. Вычисление сумм рядов. Обработка исключений

Подготовил:

Студент гр. 010101

Листванович А.А.

Проверила:

Василькова А.Н.

Минск 2023

Задача №1

Условие:

Дано натуральное число N. Напишите программу для вычисления следующей суммы ряда (начиная с единицы):

Обратите внимание: если мы воспользуемся формулой для расчёта члена ряда, что

при n = 0, член ряда равен 1;

при n = 1, мы получим -1/2;

при n = 2, мы получим 1/4;

при n = 3, мы получим -1/8.

Решение:

def prim\_1():  
 number = int(input('Введите натуральное число: '))  
 s = 1  
 x = 1  
 for n in range(1, number + 1):  
 x = ((-1) \*\* n) / (2 \*\* n)  
 s = x + s  
 print('Член ряда ', x, 'сумма ряда ', s)

Результат работы программы отображен на рисунке 1.

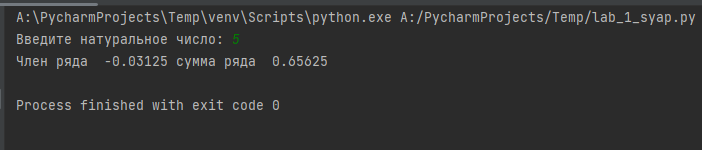


Рисунок 1 – Результат первой задачи

Задача №2

Условие:

1. Начните новую программу на Python с инициализации целочисленной переменной.

day = 32

2. Теперь добавьте блок инструкции *try*, в котором проверяется значение переменной, затем указывается исключение и соответствующее сообщение для пользователя.

try :

if day > 31 :

raise ValueError( 'Invalid Day Number' )

# Сюда добавляем операторы.

3. Затем добавьте блок инструкции except для вывода сообщения в случае появления исключения ValueError.

except ValueError as msg :

print( 'The Program found An' , msg )

4. Теперь добавьте блок инструкции *finally* для вывода сообщения после успешной обработки исключения.

finally :

print('But Today Is Beautiful Anyway.' )

5. Сохраните файл в рабочем каталоге, откройте командную строку и запустите вашу программу.

Решение:

def prim\_2():  
 title = 'Python In Easy Steps'  
 try:  
 print(titel)  
 except (NameError, IndexError) as msg:  
 print(msg)  
 day = 32  
 try:  
 if day > 31:  
 raise ValueError('Invalid Day Number')  
 except ValueError as msg:  
 print('The Program found An', msg)  
 finally:  
 print('But Today Is Beautiful Anyway.')

Результат работы программы отображен на рисунке 2.

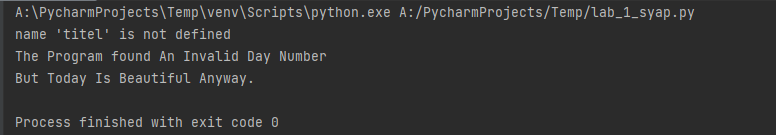


Рисунок 2 – Результат второй задачи

Задача №3 – Калькулятор комплексных чисел

Условие:

Работа с числами. Операции с комплексными числами. Разработать калькулятор для комплексных чисел на основе функции eval().

Решение:

def calculator() -> None:  
 *"""  
 Задание №3 - Калькулятор комплексных выражений через функцию eval()* ***:return****: None  
 """* try:  
 input\_str = input("Введите выражение:")  
 result: complex = complex(eval(input\_str))  
 print('Результат: ' + str(result))  
 except Exception as msg:  
 print(msg)

Результат работы программы отображен на рисунке 3.

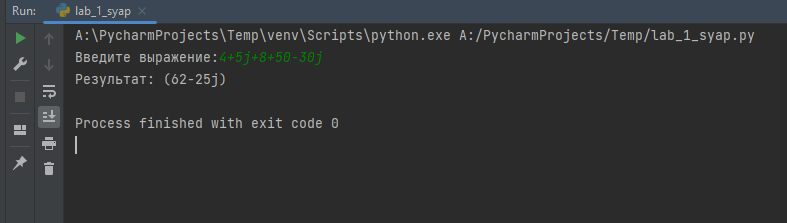


Рисунок 3 – Результат третьей задачи

Задача №4:

Условие задачи изображено на рисунке 4.



Рисунок 4 – условие четвертой задачи

Решение:

def if\_statements() -> None:  
 *"""  
 Задание №4 - Вывод строк в зависимости от введенного значения  
 Реализовано через граф(все зависимости хранятся в словаре)  
 Текстовые сообщения организованы в Enum для исключения повторений* ***:return****: None  
 """* class Answer(enum.Enum):  
 first = 'Ты попал в бассейн'  
 second = 'Тебя съел тиранозавр'  
 third = 'Улетел в космос'  
 fourth = 'Улетел на бали'  
 fifth = 'Улетел на луну'  
 sixth = 'Ты попал в лужу'  
  
 graph\_visual = {  
 1: [11, 12],  
 2: [21, 22],  
 3: [31, 32],  
 4: Answer.second,  
 11: Answer.first,  
 12: Answer.second,  
 21: Answer.second,  
 22: [221, 222],  
 31: [222, 311],  
 32: [321, 322],  
 33: [1, ],  
 221: Answer.third,  
 222: Answer.fourth,  
 311: Answer.fifth,  
 321: Answer.sixth,  
 322: Answer.fifth  
 }  
 try:  
 temp\_variable = int(input("Введите первоначальное значение: "))  
  
 # Итерируемся, пока зависимый объект не будет сообщением  
 while not isinstance(graph\_visual[temp\_variable], Answer):  
 # Сохраняем зависимый объект  
 potentials = graph\_visual[temp\_variable]  
 # Спрашиваем новое значение  
 print("Выберите следующее значение из следующих: ", graph\_visual[temp\_variable])  
 temp\_variable = int(input())  
 # Валидируем новое значение  
 if isinstance(potentials, list) and temp\_variable not in potentials:  
 raise KeyError  
  
 print(graph\_visual[temp\_variable].value, end='')  
 except KeyError:  
 print("Вы ввели некорректное значение!")

Результат работы программы отображен на рисунке 5.

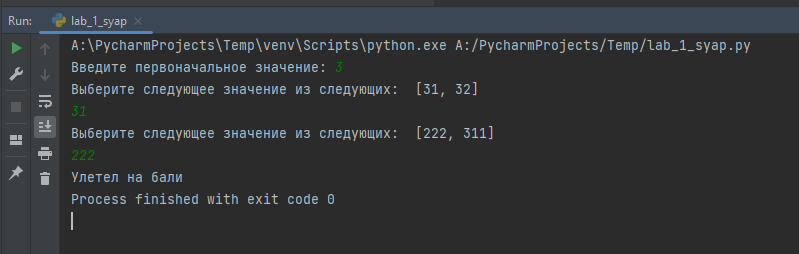


Рисунок 5 – Результат четвертой задачи