МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ И. С. ТУРГЕНЕВА»

Кафедра программной инженерии

**ОТЧЕТ**

по лабораторной работе № 1

на тему: «Основные типы данных. Управляющие конструкции»

по дисциплине: «Программирование на языке Python»

Выполнил: Евдокимов Н.А.

Институт приборостроения, автоматизации и информационных технологий

Направление: 09.03.04 «Программная инженерия»

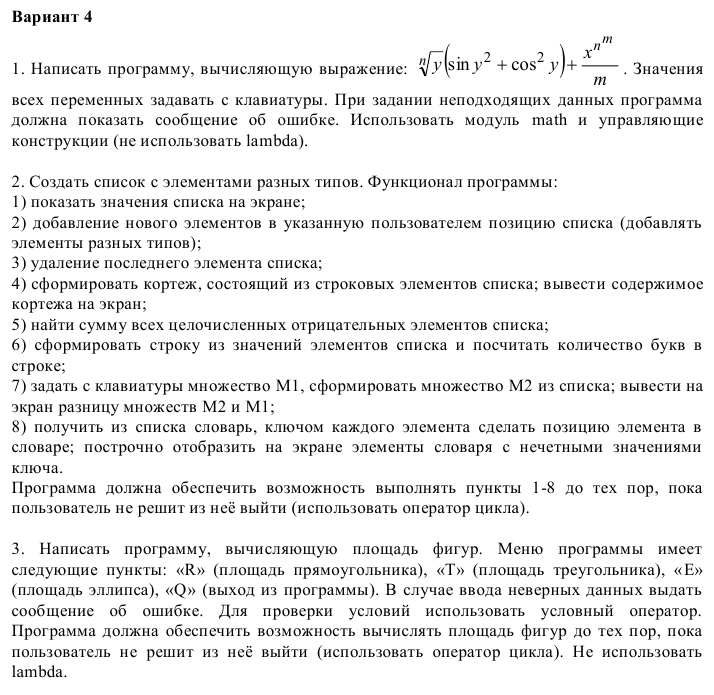
Группа: 71-ПГ

Проверила: Захарова О.В.

Отметка о зачете:

Дата: «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2019 г.

Орёл, 2019

Задание

Код

1.

import math as m

x = None

while x is None:

try:

x = float(input('Введите x: '))

except ValueError:

x = None

y = None

while y is None:

try:

y = float(input('Введите y: '))

except ValueError:

y = None

n = None

while n is None:

try:

n = float(input('Введите n: '))

except ValueError:

n = None

m = None

while m is None or m == 0:

try:

m = float(input('Введите m: '))

except ValueError:

m = None

result = m.pow(y, 1 / n) \* (m.sin(m.pow(y, 2)) + m.pow(m.cos(y), 2)) + m.pow(x, m.pow(n, m)) / m

print('Результат вычислений:', result)

2.

import textwrap as tw

import re

def print\_menu():

menu = """

1) Вывести список на экран

2) Добавить элемент в список

3) Удалить элемент из списка

4) Составить кортеж из четных элементов списка

5) Найти сумму всех целочисленных элементов

6) Сформировать строку и посчитать слова

7) Задать 2 множества и вывести их объединение

8) Сделать словарь и вывести на экран пары с ключом > 5

0) Выйти

"""

print(tw.dedent(menu))

def add\_item(target\_list):

if type(target\_list) != list:

raise ValueError

index = None

while index is None:

try:

index = int(input('Введите индекс: '))

except ValueError:

index = None

value = input('Введите значение: ')

if '.' in value:

# float

try:

typed\_value = float(value)

except ValueError:

typed\_value = value

elif 'j' in value:

# complex

try:

typed\_value = complex(value)

except ValueError:

typed\_value = value

else:

# int or string

try:

typed\_value = int(value)

except ValueError:

typed\_value = value

target\_list.insert(index, typed\_value)

def remove\_item(target\_list):

if type(target\_list) != list:

raise ValueError

index = None

while index is None:

try:

index = int(input('Введите индекс: '))

except ValueError:

index = None

target\_list.pop(index)

def make\_tuple\_of\_even(src\_list):

if type(src\_list) != list:

raise ValueError

list\_of\_even = src\_list[::2]

return tuple(list\_of\_even)

def sum\_of\_ints(src\_list):

if type(src\_list) != list:

raise ValueError

ints\_sum = 0

for item in src\_list:

if type(item) == int:

ints\_sum += item

return ints\_sum

def word\_count(src\_list):

if type(src\_list) != list:

raise ValueError

result\_str = ""

for item in src\_list:

result\_str += str(item)

words = re.findall('[a-zA-Zа-яА-Я]+', result\_str)

return len(words)

def make\_set\_union():

set\_str1 = input('Введите первое множество через запятую: ')

set\_str2 = input('Введите второе множество через запятую: ')

set1 = set(set\_str1.split(','))

set2 = set(set\_str2.split(','))

union = sorted(set1.union(set2))

print(union)

def make\_dict(src\_list):

if type(src\_list) != list:

raise ValueError

result\_dict = {}

for index, item in enumerate(src\_list):

if index <= 5:

continue

result\_dict[index] = item

print(result\_dict)

def run():

should\_run = True

item\_list = [1, None, 2.3, "sda"]

while should\_run:

print\_menu()

try:

user\_choice = int(input())

except ValueError:

user\_choice = None

if user\_choice is None:

print("Некоректный ввод")

continue

if user\_choice == 0:

should\_run = False

elif user\_choice == 1:

print(item\_list)

elif user\_choice == 2:

add\_item(item\_list)

elif user\_choice == 3:

remove\_item(item\_list)

elif user\_choice == 4:

print(make\_tuple\_of\_even(item\_list))

elif user\_choice == 5:

print(sum\_of\_ints(item\_list))

elif user\_choice == 6:

print(word\_count(item\_list))

elif user\_choice == 7:

make\_set\_union()

elif user\_choice == 8:

make\_dict(item\_list)

if \_\_name\_\_ == '\_\_main\_\_':

run()

3.

import textwrap as tw

import math as m

def print\_menu():

menu = """

R) Площадь прямоугольника

T) Площадь прямоугольного треугольника

M) Площадь правильного многоугольника

E) Выйти

"""

print(tw.dedent(menu))

def rectangle\_area():

a, b = None, None

while a is None:

try:

a = float(input('Введите первую сторону: '))

except ValueError:

a = None

while b is None:

try:

b = float(input('Введите вторую сторону: '))

except ValueError:

b = None

area = a \* b

print(area)

def right\_triangle\_area():

a, b = None, None

while a is None:

try:

a = float(input('Введите первую сторону: '))

except ValueError:

a = None

while b is None:

try:

b = float(input('Введите вторую сторону: '))

except ValueError:

b = None

area = a \* b / 2

print(area)

def right\_polygon\_area():

n, a = None, None

while n is None:

try:

n = float(input('Введите количество сторон: '))

except ValueError:

n = None

while a is None:

try:

a = float(input('Введите длину стороны: '))

except ValueError:

a = None

area = n / 4 \* a\*\*2 \* (1 / m.tan(m.pi / n))

print(area)

def run():

should\_run = True

while should\_run:

print\_menu()

user\_choice = input().lower()

if user\_choice == 'r':

rectangle\_area()

elif user\_choice == 't':

right\_triangle\_area()

elif user\_choice == 'm':

right\_polygon\_area()

elif user\_choice == 'e':

should\_run = False

else:

print("Некоректный ввод")

continue

if \_\_name\_\_ == '\_\_main\_\_':

run()