# МИНИCTEPCTBO НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

**ФЕДЕРАЦИИ**

# Федеральное государственное автономное образовательное учреждение высшего образования

**«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

# Кафедра инфокоммуникаций Институт цифрового развития

**ОТЧЁТ**

# по лабораторной работе №13

Дисциплина: «Основы программной инженерии»

Тема: «Функции с переменным числом параметров в Python»

Выполнил: студент 2 курса группы Пиж-б-о-21-1

Рязанцев Матвей Денисович

Ставрополь 2022

Цель работы: приобретение навыков по работе с функциями с переменным числом

**Выполнение работы**

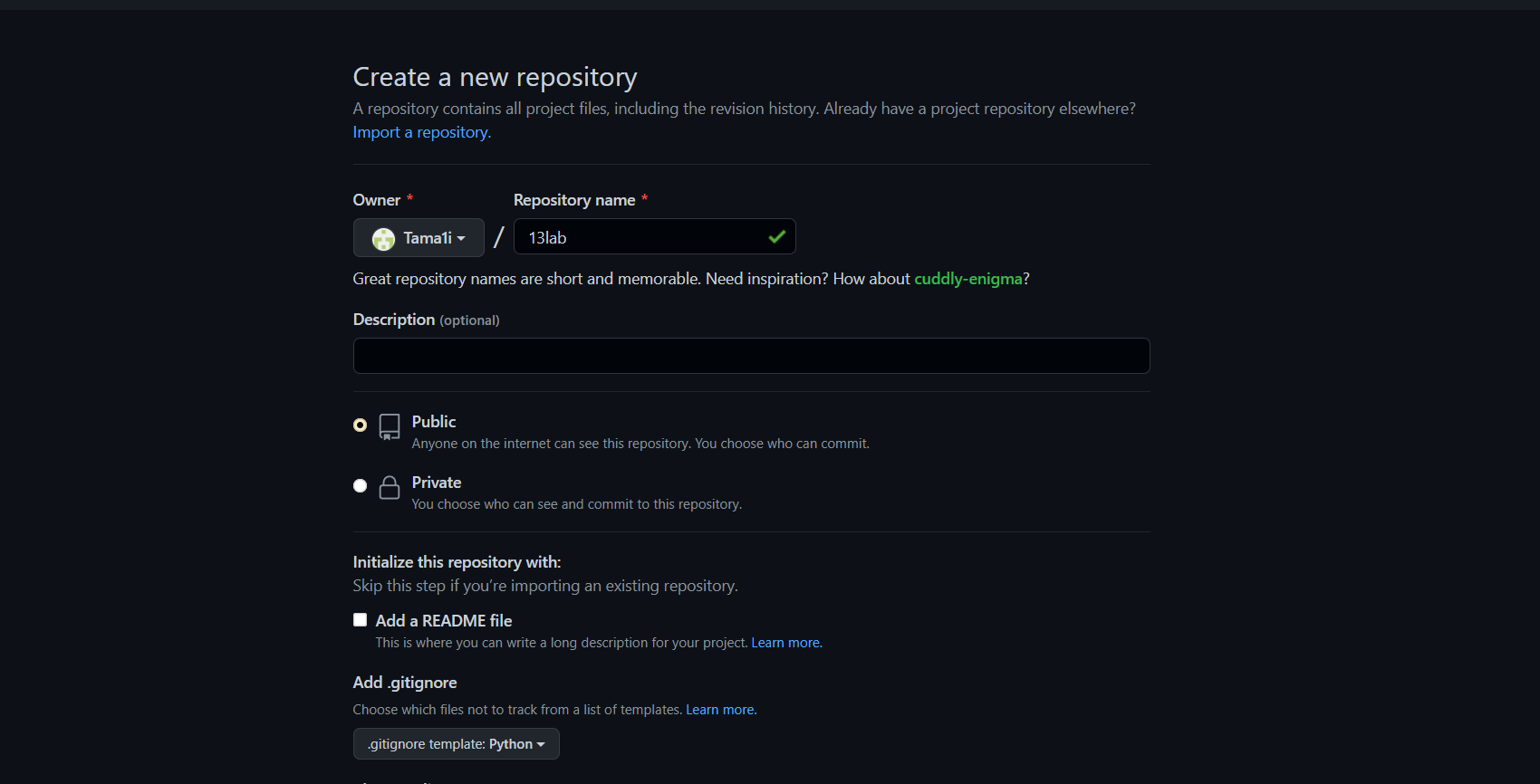


Рисунок 1 -создание репозитория

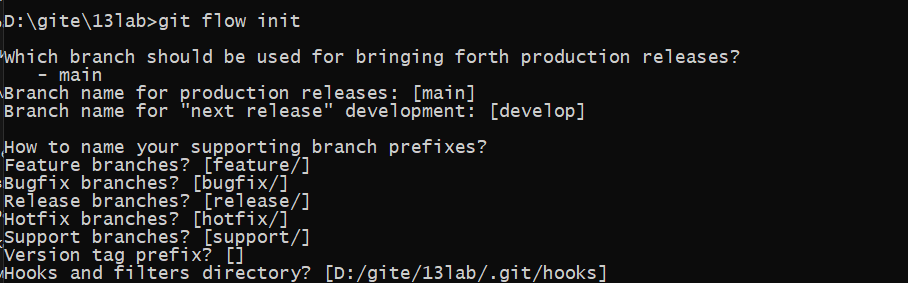


Рисунок 2 - Организация репозитория по модели ветвления git flow

Код общего задания:

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

def median(\*args):

if args:

values = [float(arg) for arg in args]

values.sort()

n = len(values)

idx = n // 2

if n % 2:

return values[idx]

else:

return (values[idx - 1] + values[idx]) / 2

else:

return None

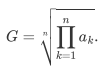
if \_\_name\_\_ == "\_\_main\_\_":

print(median())

print(median(3, 7, 1, 6, 9))

print(median(1, 5, 8, 4, 3, 9))

Решить поставленную задачу: написать функцию, вычисляющую среднее геометрическое своих аргументов 



Если функции передается пустой список аргументов, то она должна возвращать значение None.

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

import math

def aver\_geom(\*args):

if args:

values = [float(arg) for arg in args]

n = 1

sz = len(args)

for arg in values:

n = n \* arg

g = math.pow(n, 1 / sz)

return g

else:

return None

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

print(f'geometric mean is: {aver\_geom(1, 2, 3, 4, 5, 6, 7)}')

print(f'geometric mean is: {aver\_geom()}')

print(f'geometric mean is: {aver\_geom(2.3, 6.5, 9.2, 3.4)}')

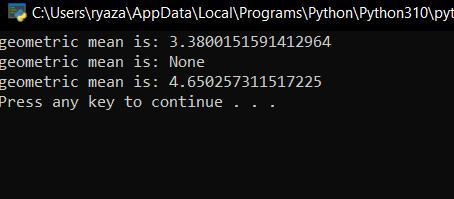


Рисунок 3 – результат работы проограммы

Решить поставленную задачу: написать функцию, вычисляющую среднее гармоническое своих аргументов 



#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

import math

def aver\_harm(\*args):

if args:

values = [float(arg) for arg in args]

sz = len(args)

s = 0

for arg in values:

s += 1 / arg

g = sz / s

return g

else:

return None

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

print(f'harmonic mean is: {aver\_harm(1, 2, 3, 4, 5, 6, 7, 8)}')

print(f'harmonic mean is: {aver\_harm()}')

print(f'harmonic mean is: {aver\_harm(1.5, 4.6, 9.3, 8.0, 10.5)}')

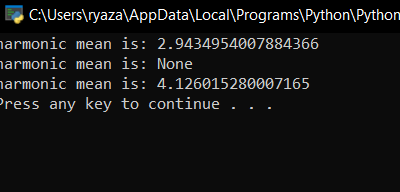


Рисунок 4 – результат работы программы

Индивидуальное задание В5

Код программы:

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

import sys

def s(\*args):

k = 0

o = 0

s1 = 0

for i in args:

k += 1

if i < 0:

kk = k

for i in args:

if i < kk:

if o == 1:

s1 += i

if i < 0:

o = 1

return (s1)

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

print(s(3, -7, 1, 6, 9, -5, -6, 9, 4, -8, 3, 5))

print(s(-2,3,1,4,24,-5,-4,-4, 2,46,-2 ,25,3))

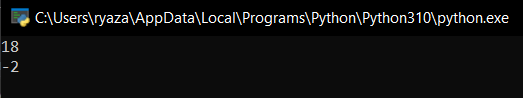


Рисунок 5 – результат работы программы идз

Самостоятельно подберите или придумайте задачу с переменным числом именованных аргументов. Приведите решение этой задачи.

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

import sys

def s(\*args):

k = 0

o = 0

s1 = 0

res = []

for i in args:

res.append(i)

return (res)

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

print(s("toyota "," mark 1 ","mark 2 ", " mark3 "))

print(s("nissan "," 350 Z "," 450 Z "," 250fx "))

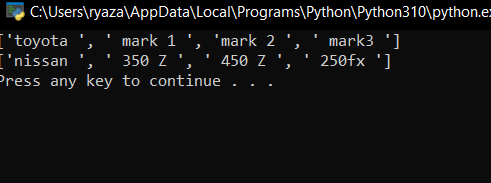


Рисунок 6 – результат работы программы

**Контрольные вопросы**

1. Рекурсия существенно сокращает объем кода и входит во многие

встроенные функции языков.

2. База рекурсии – это тривиальный случай, при котором решение задачи

очевидно, то есть не требуется обращение функции к себе.

3. Компьютер использует стек вызовов — специальную область памяти, где

хранит данные о точках перехода между фрагментами кода.

последовательность шагов, выполняемых при вызове функции: a. Программа

сталкивается с вызовом функции. b. Создается фрейм стека, который

помещается в стек. c. Процессор переходит к точке начала выполнения

функции. d. Инструкции внутри функции начинают выполняться. После

завершения функции, выполняются следующие шаги: e. Регистры

восстанавливаются из стека вызовов. f. Фрейм стека вытягивается из стека.

Освобождается память, которая была выделена для всех локальных

переменных и аргументов. g. Обрабатывается возвращаемое значение. h. ЦП

возобновляет выполнение кода (исходя из обратного адреса).

4. Чтобы получить текущее значение максимальной глубины рекурсии

следует вызвать функцию sys.getrecursionlimit()

5. Когда предел достигнут, возникает исключение: RuntimeError: Maximum

Recursion Depth Exceeded