

**МИНИСТЕРСТВО 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 №12
Дисциплина: «Основы программной инженерии»
Тема: «Работа с множествами в языке Python»**

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

Ставрополь 2022

Цель работы: приобретение навыков по работе с множествами при написании программ с помощью языка программирования Python версии 3.x.

Выполнение работы

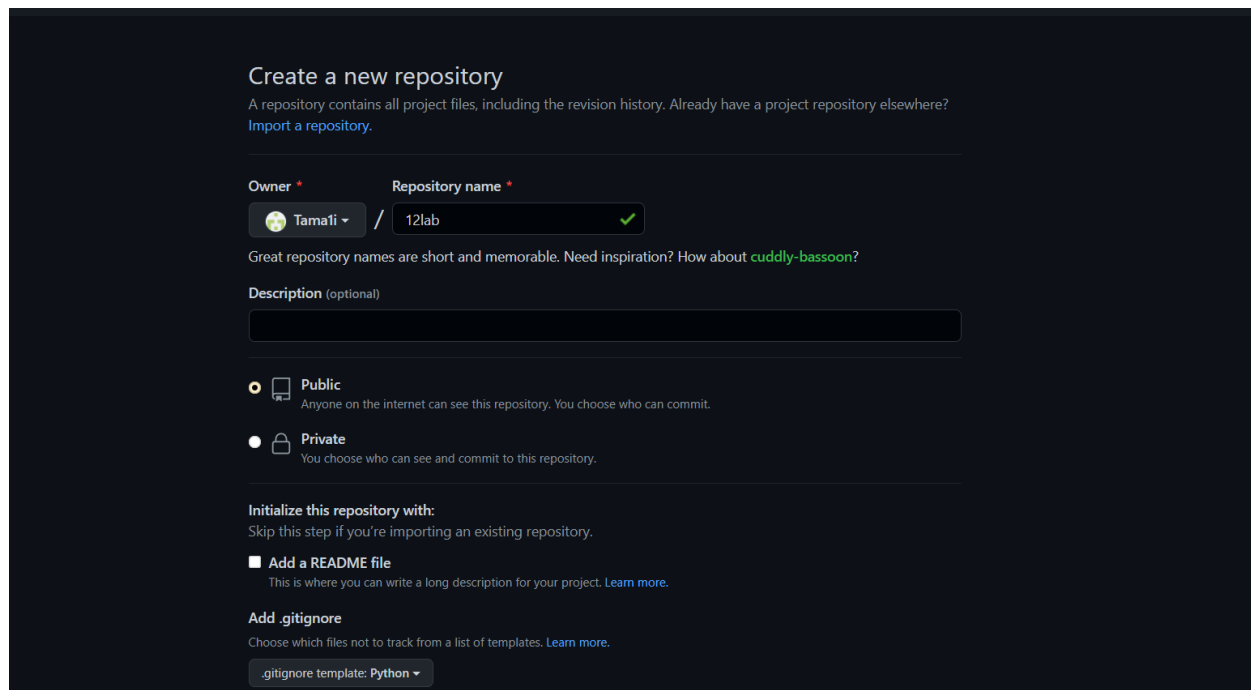


Рисунок 1 -создание репозитория

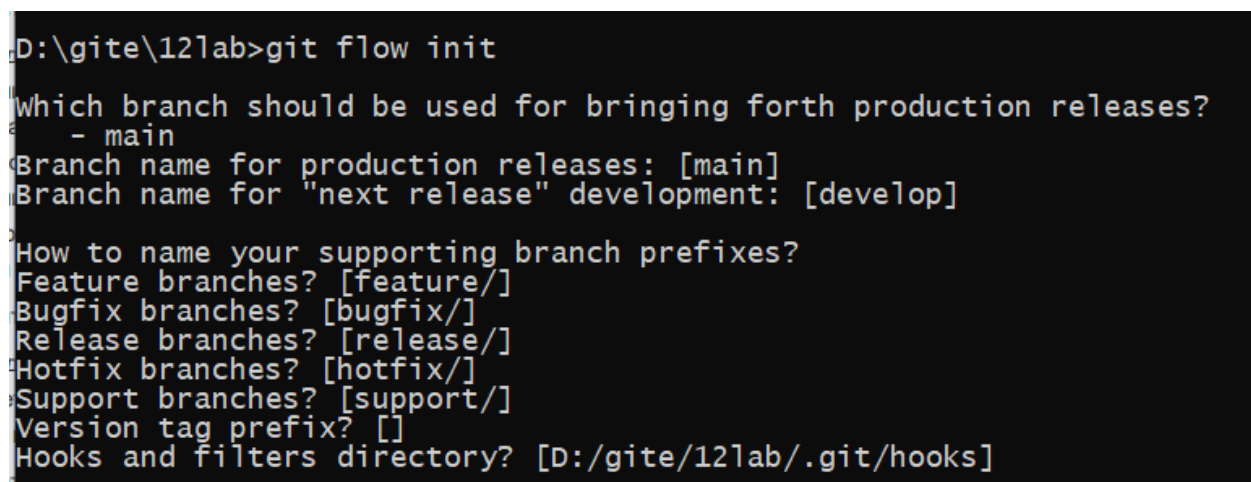


Рисунок 2 - Организация репозитория по модели ветвления git flow

Код общего задания:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

# Эта программа показывает работу декоратора, который производит оптимизацию
# хвостового вызова. Он делает это, вызывая исключение, если оно является его
# прародителем, и перехватывает исключения, чтобы вызвать стек.
```

```
import sys

class TailRecurseException:
    def __init__(self, args, kwargs):
        self.args = args
        self.kwargs = kwargs

def tail_call_optimized(g):
    """
    Эта программа показывает работу декоратора, который производит оптимизацию
    хвостового вызова. Он делает это, вызывая исключение, если оно является его
    прародителем, и перехватывает исключения, чтобы подделать оптимизацию хвоста.

    Эта функция не работает, если функция декоратора не использует хвостовой вызов.
    """
    def func(*args, **kwargs):
        f = sys._getframe()
        if f.f_back and f.f_back.f_back and f.f_back.f_back.f_code == f.f_code:
            raise TailRecurseException(args, kwargs)
        else:
            while True:
                try:
                    return g(*args, **kwargs)
                except TailRecurseException, e:
                    args = e.args
                    kwargs = e.kwargs
    func.__doc__ = g.__doc__
    return func
```

Индивидуальное задание

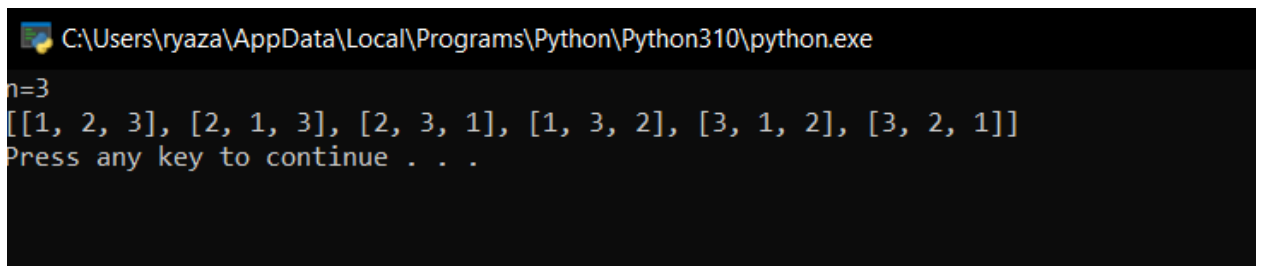
Код программы:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys

def ap(arr):
    if len(arr)==1:
        return [arr]
    else:
        a=arr[0] #берем 1 "болт, сотню кротов, два ведра силикона, 2 стопки картона, 12
        # 3 крышки от пива и шутку админа, охапку дров - французский танк готов"
        p=ap(arr[1:])
        r=[]
        for k in p: #цикл для перебора начального числа
            for i in range(len(k)): #цикл для перебора всех позиций 1
                t=k[0:i]+[a]+k[i:] #делаем разрез куда вставляем 1
                r.append(t) #добавляем новую комбинацию в список комбинаций
            r.append(k+[a])
        return r

if __name__ == '__main__':
    n=int(input("n="))
    print(ap([i for i in range(1,n+1)]))
```



```
C:\Users\ryaza\AppData\Local\Programs\Python\Python310\python.exe
n=3
[[1, 2, 3], [2, 1, 3], [2, 3, 1], [1, 3, 2], [3, 1, 2], [3, 2, 1]]
Press any key to continue . . .
```

Рисунок 6 – результат работы программы илз

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: а.
Программа
сталкивается с вызовом функции. б. Создается фрейм стека, который
помещается в стек. с. Процессор переходит к точке начала выполнения
функции. d. Инструкции внутри функции начинают выполняться.
После
завершения функции, выполняются следующие шаги: е. 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`