МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БРАТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

Факультет Экономики и Строительства

Базовая кафедра менеджмента и информационных технологий

**Информатика и программирование**

Лабораторная работа № 9

**множества**

Вариант 15

|  |  |  |
| --- | --- | --- |
| Работу выполнила: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Рыженко В.И.,  обучающаяся группы ПИЭ-23 |
| Приняла: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | А.Н. Ефремова,  ст. преподаватель кафедры ИМиФ |

Братск 2024 г.

**Цель работы:** познакомиться с основными конструкциями, используемыми для работы с множеством; приемами программной реализации на языке программирования Python; произвести отладку и тестирование полученных программ.

**Задача 1**:

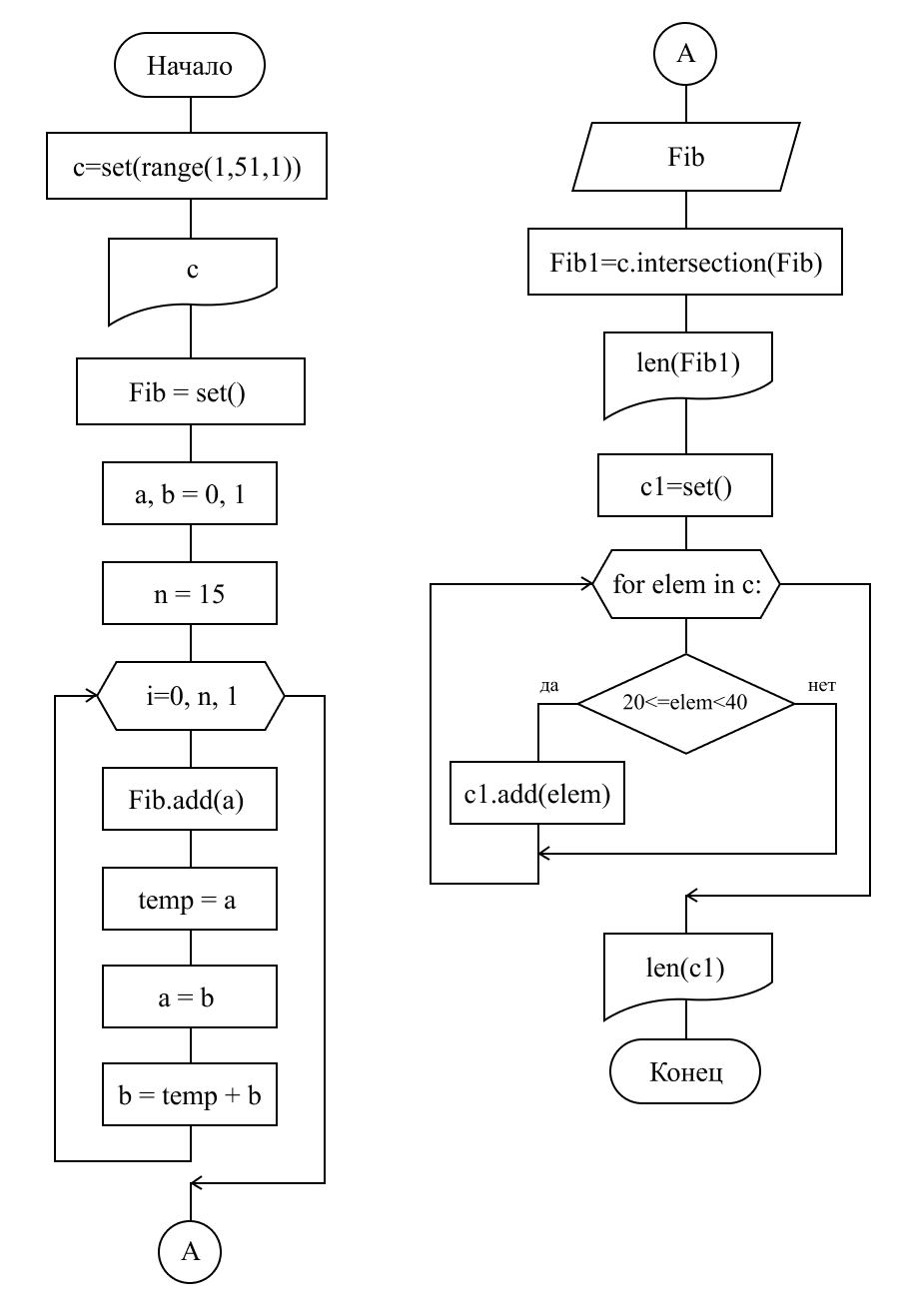
Даны целые числа от 1 до 50. Определить, сколько среди них чисел Фибоначчи и сколько чисел, первая значащая цифра в десятичной записи которых 2 или 3.

**Постановка задачи:** Необходимо образовать множество чисел от 1 до 50 и множество чисел Фибоначчи. Найти пересечение этих множеств и его длину.

**Метод решения**: Образовываем множество в диапазоне от 1 до 51 с шагом 1. Формируем список Фибоначчи: задаём переменные a=0 и b=1 и с помощью цикла for добавляем их в множество, параллельно изменяя по формуле. С помощью метода intersection() находим пересечение между исходным и новым множеством. Затем находим длину пересечения.

Чтобы найти количество чисел со значащими цифрами 2 и 3 в десятичной записи, отберём их с помощью условия if 20<= elem < 40 и добавим во множество c1.

**Блок-схема алгоритма**

****

**Программный код:**

c=set(range(1,51,1))

print("Числа от 1 до 50:", \*c)

Fib = set()

a, b = 0, 1

n = 15

for i in range(n):

Fib.add(a)

temp = a

a = b

b = temp + b

print("Числа Фибоначчи:", Fib)

Fib1=c.intersection(Fib)

print("Количество чисел Фибоначи в первом списке:",len(Fib1))

c1=set()

for elem in c:

if 20<= elem < 40:

c1.add(elem)

print("Количество чисел c 2 или 3 значащими цифрами в десятичной записи:",len(c1))

*Результат:*

Числа от 1 до 50: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50

Числа Фибоначчи: {0, 1, 2, 3, 34, 5, 8, 233, 377, 13, 144, 21, 55, 89}

Количество чисел Фибоначи в первом списке: 8

Количество чисел c 2 или 3 значащими цифрами в десятичной записи: 20

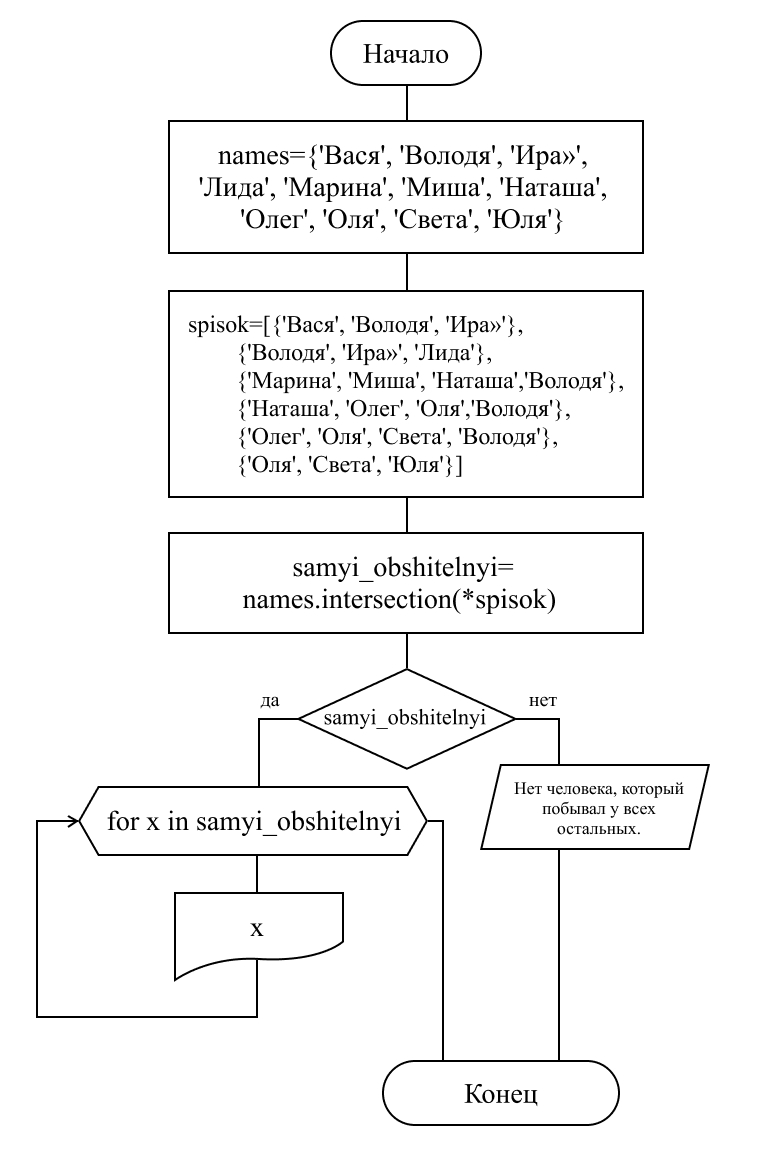
**Задача 2**:

Дано множество имен = {«Вася», «Володя», «Ира», «Лида», «Марина», «Миша», «Наташа», «Олег», «Оля», «Света», «Юля»}. Есть список, каждый элемент которого – множество людей, побывавших у какого-либо члена группы. Определить, есть ли хотя бы один человек, побывавший в гостях у всех остальных из группы.

**Постановка задачи:** необходимо найти пересечение данного множества со элементами списка гостей и вывести на экран.

**Метод решения:** образуем множество имён names и список гостей spisok (каждый элемент списка – множество, означающее гостей одного человека из множества имён names). С помощью метода intersection() находим пересечение и записываем в множество samyi\_obshitelnyi. Проверяем наличие пересечения с помощью цикла for. Если оно есть, выводим samyi\_obshitelnyi на экран. Если нет, выводим: "Нет человека, который побывал у всех остальных."

**Блок-схема алгоритма**



**Программный код:**

names={'Вася', 'Володя', 'Ира»', 'Лида', 'Марина', 'Миша', 'Наташа', 'Олег', 'Оля', 'Света', 'Юля'}

spisok=[{'Вася', 'Володя', 'Ира»'},

{'Володя', 'Ира»', 'Лида'},

{'Марина', 'Миша', 'Наташа','Володя'},

{'Наташа', 'Олег', 'Оля','Володя'},

{'Олег', 'Оля', 'Света', 'Володя'},

{'Оля', 'Света', 'Юля', 'Володя'}]

samyi\_obshitelnyi=names.intersection(\*spisok)

if samyi\_obshitelnyi:

for x in samyi\_obshitelnyi:

print("Самый(-ые) общительный(-ые):", x)

else:

print("Нет человека, который побывал у всех остальных.")

*Результат:*

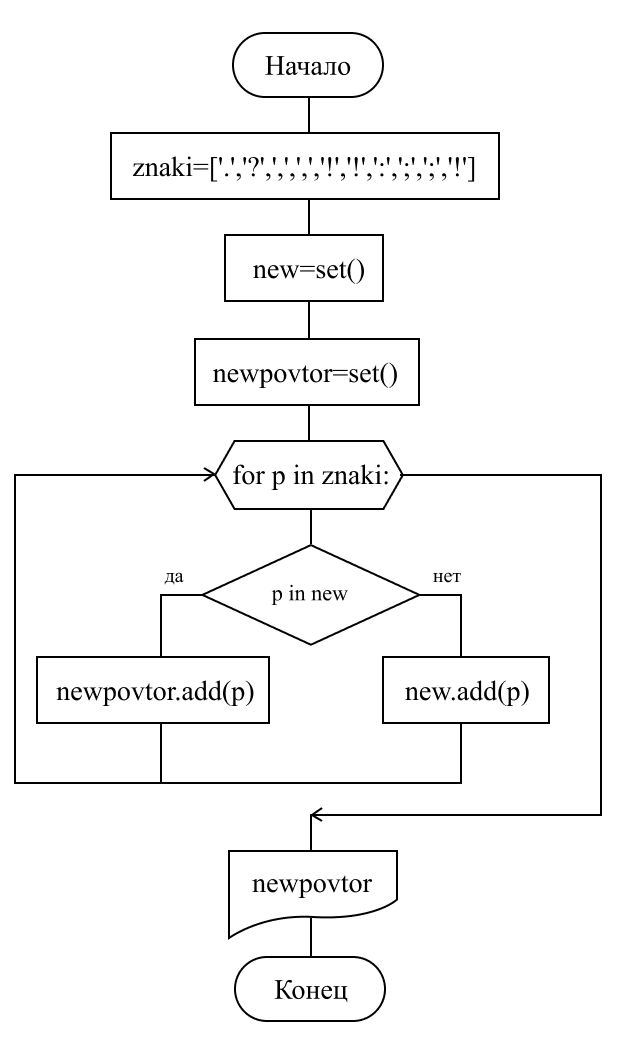
Самый(-ые) общительный(-ые): Володя

**Задача 3**: Дана последовательность символов. Требуется построить и напечатать множество, элементами которого являются встречающиеся в последовательности знаки препинания.

**Постановка задачи:** образовать множество любых символов. Затем образовать ещё одно, состоящее из повторяющихся символов первого множества. Вывести его на экран.

**Метод решения:** в переменной znaki задаём множество символов. Следом задаём ещё два множества – new и newpovtor. С помощью цикла for проверяем их на наличие символов из множества znaki. Если символ есть в new, добавляем его в newpovtor. Если его там нет, добавляем его в new – и так проверяем все элементы множества znaki. Выводим на экран множество newpovtor – это и будет ответом к задаче.

**Блок-схема алгоритма**

****

**Программный код:**

znaki=['.','?',',',',','!','!',':',';',';','!']

new=set()

newpovtor=set()

for p in znaki:

if p in new:

newpovtor.add(p)

else:

new.add(p)

print("Повторяющиеся символы:", newpovtor)

*Результат:*

Повторяющиеся символы: {';', ',', '!'}

**Вывод:** удалось познакомиться с основными конструкциями, используемыми для работы с множеством; приемами программной реализации на языке программирования Python; была произведена отладка и тестирование полученных программ.