

Департамент образования города Москвы

Государственное автономное образовательное учреждение высшего  
образования города Москвы «Московский Городской Педагогический  
Университет»

Институт цифрового образования  
Департамент информатики, управления и технологий

## ЛАБОРАТОРНАЯ РАБОТА №1.2

по дисциплине «Практикум по Python»

Направление подготовки 38.03.05 – «бизнес-информатика»

Профиль подготовки «Аналитик бизнес-процессов: автоматизация и  
управление бизнесом» (очная форма обучения)

Тема: «Написание скриптов с использованием условных  
операторов. Решение задач с использованием циклов.»

Выполнила: Губайдуллина А. И.

Студентка группы АБП – 231

Проверил: Босенко Т. М. доцент

Москва

2025

### Цели:

1. Изучить основные управляющие конструкции языка Python: условный оператор и циклы.
2. Научиться использовать управляющие структуры для решения задач различной сложности.
3. Закрепить навыки обработки данных с использованием коллекций и итераторов.
4. Развить умение комбинировать условия и циклы для оптимизации алгоритмов.

### Задачи:

1. Освоить условный оператор if-elif-else и научиться применять его для решения задач с разветвлением логики.
2. Понять работу циклов while и for, включая их особенности.
3. Изучить методы работы с коллекциями в циклах (например, списки, словари, множества).
4. Разобраться с командами управления циклами break и continue.
5. Изучить коллекционные включения (list comprehensions) для создания новых коллекций.
6. Практически применять комбинации условий и циклов для построения сложных алгоритмов.

№1.2.1. Рассчитать значение  $f$  при заданном значении вещественного числа  $x$ :

$$f(x) = \begin{cases} \sqrt{x} + x^2, & \text{при } x \geq 0 \\ \frac{1}{x}, & \text{в противном случае} \end{cases}$$

При выводе на экран оставьте 2 знака после запятой.

Решение:

```
# Задание task_01_2_01.


# Выполнила: Губайдуллина А. И.
# Группа: АБП-231

x = int(input())

if x >= 0:
    f = x ** (1/2) + x ** 2
else:
    f = 1 / x

print(round(f, 2))
```

---



```
10
103.16
```

Рис. 1. Результат нахождения  $f$

№1.2.2. Определите максимальное и минимальное значения из двух различных целых чисел.

Решение:


```
# Задание task_01_2_02.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

a1 = int(input())
a2 = int(input())

# Переменная 'a_min' должна содержать минимум, 'a_max' - максимум

if a1 == a2:
    print("Числа равны! Пожалуйста, введите различные числа.")
else:
    maximum = max(a1, a2)
    minimum = min(a1, a2)
    print(f"Максимальное значение: {maximum}")
    print(f"Минимальное значение: {minimum}")
```

---



```
1
2
Максимальное значение: 2
Минимальное значение: 1
```

Рис. 2. Результат нахождения maximum, minimum двух различных чисел

№1.2.3. Вася пытается высунуть голову в форточку размерами  $a$  и  $b$  см. Приняв условно, что его голова - круглая диаметром  $d$  см, определите, сможет ли Вася сделать это. Для прохождения головы в форточку необходим зазор в 1 см. с каждой стороны. Все величины - целые числа.

Решение:

```
# Задание task_01_2_03.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

a = int(input("Введите длину форточки: "))
b = int(input("Введите ширину форточки: "))
d = int(input("Введите диаметр головы: "))

# Введенные числа должны быть положительными, если так - осуществляем
# расчет, иначе выводим "Проверьте ввод"

fix = 1 # Зазор
if min(a, b) - 2 * fix >= d:

    # Вывести "Да" или "Нет"
    print("Да")
    # Удалите комментарий и допишите код
else:
    print("Нет")
    # Удалите комментарий и допишите код
```

```
➡ Введите длину форточки: 5
Введите ширину форточки: 6
Введите диаметр головы: 6
Нет
```

Рис. 3. Результат выполнения программы

№1.2.4. Известны год и номер месяца сегодняшнего дня, а также год и номер месяца рождения человека (нумерация месяцев с 1: январь - 1 и т.д.). Определите возраст человека (число полных лет).

Решение:

```
# Задание task_01_2_04.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

year_today = int(input("Введите год сегодняшнего дня: "))
month_today = int(input("Введите месяц сегодняшнего дня: "))

year = int(input("Введите год рождения человека: "))
month = int(input("Введите месяц рождения человека: "))

# Считается, введенные значения находятся в допустимых пределах, и
# что 'month_today' >= 'month' (проверять значения не нужно)

if month_today >= month:

    # Результат необходимо записать в переменную 'age'
    age = (year_today - year, month_today - month)
    # Удалите комментарий и допишите код

print("Число полных лет: ", age)
```

---

 Введите год сегодняшнего дня: 2025  
Введите месяц сегодняшнего дня: 9  
Введите год рождения человека: 2005  
Введите месяц рождения человека: 9  
Число полных лет: (20, 0)

Рис. 4. Результат нахождения возраста человека

№1.2.5. Дана точка с целыми ненулевыми координатами (x;y). Определить номер четверти координатной плоскости, которой она принадлежит.

Решение:

```

# Задание task_01_2_05.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

x = int(input("Введите координату x: "))
y = int(input("Введите координату y: "))

if x > 0 and y > 0:
    print("1-я четверть")

elif x > 0 and y < 0:
    print("4-я четверть")

elif x < 0 and y > 0:
    print("2-я четверть")

elif x < 0 and y < 0:
    print("3-я четверть")

```

---

```

Введите координату x: 5
Введите координату y: 7
1-я четверть

```

Рис. 5. Результат нахождения четверти координатной плоскости

№1.2.6. Даны вещественные числа  $a$ ,  $b$ ,  $c$  ( $a \neq 0$ ). Решите уравнение  $ax^2 + bx + c = 0$ . При выводе значений оставьте 1 знак после запятой.

Решение:

```

# Задание task_01_2_06.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

import math

a = int(input("Введите число a: "))
b = int(input("Введите число b: "))
c = int(input("Введите число c: "))

if a == 0:
    print("Ошибка: коэффициент a не может быть равен 0")

# Расчет дискриминанта

else:
    D = b ** 2 - 4 * a * c

```

```

# Вывод решения в зависимости от значения дискриминанта
if D > 0:
    # Два различных корня
    x1 = (-b + math.sqrt(D)) / (2*a)
    x2 = (-b - math.sqrt(D)) / (2*a)
    print("x1 =", round(x1, 1))
    print("x2 =", round(x2, 1))
elif D == 0:
    # Один корень (двукратный)
    x = -b / (2*a)
    print("x =", round(x, 1))
else:
    # Действительных корней нет
    print("Нет действительных корней")

```

---

```

Введите число a: 1
Введите число b: 2
Введите число c: 1
x = -1.0

```

Рис. 6. Результат нахождения корней уравнения

№1.2.7. Дана непустая последовательность целых чисел, оканчивающаяся нулем. Найти сумму и количество введенных чисел.

Решение:

```

# Задание task_01_2_07.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

nums_sum = 0 # сумма
nums_count = 0 # количество

x = int(input("Введите число(0 для завершения): "))
while x != 0:
    nums_sum += x
    nums_count += 1

    x = int(input("Введите число(0 для завершения): "))

print("Сумма введенных чисел: ", nums_sum)

print("Количество введенных чисел: ", nums_count)

```

---

```

Введите число(0 для завершения): 1
Введите число(0 для завершения): 2
Введите число(0 для завершения): 3
Введите число(0 для завершения): 4
Введите число(0 для завершения): 0
Сумма введенных чисел: 10
Количество введенных чисел: 4

```

Рис. 7. Результат нахождения суммы и количества введенных чисел последовательности

№1.2.8. Дано число  $n$ . Из чисел 0,5,10,15,20,25,... напечатать те, которые не превышают  $n$ .

Решение:

```
# Задание task_01_2_08.  
#  
# Выполнил: Губайдуллина А. И.  
# Группа: АБП-231  
  
n = int(input("Введите число: "))  
  
# Первый вариант: с помощью цикла while  
print("Числа из последовательности 0,5,10,15,... которые не превышают", n)  
current = 0  
while current <= n:  
    print(current)  
    current += 5
```

```
➡ Введите число: 18  
Числа из последовательности 0,5,10,15,... которые не превышают 18  
0  
5  
10  
15
```

Рис. 8. Результат выполнения программы

№1.2.9. Дано вещественное число  $a$ . Найдите наименьшее натуральное  $n$ , для которого верно

$$1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} > a$$

Решение:



```

# Задание task_01_2_09.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

a = float(input("Введите число a: "))

n = 1
x_sum = 0.0

while x_sum <= a:
    x_sum += 1 / n
    n += 1

print("Наименьшее натуральное n:", n - 1)

```

---

⇒ Введите число a: 1.5  
Наименьшее натуральное n: 3

Рис. 9. Результат нахождения наименьшего натурального n

№1.2.10. Дано натуральное число. Определите сумму и количество его цифр.

Решение:

```

# Задание task_01_2_10.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

n = int(input("Введите число: "))

n_sum = 0
n_count = 0

temp = n
if temp == 0:
    n_sum = 0
    n_count = 1
else:
    while temp > 0:
        pos1 = temp % 10
        n_sum += pos1
        n_count += 1

        temp //= 10

print(f"Число: {n}")
print(f"Сумма цифр: {n_sum}")
print(f"Количество цифр: {n_count}")

```

---

⇒ Введите число: 12345  
Число: 12345  
Сумма цифр: 15  
Количество цифр: 5

Рис. 10. Результат нахождения суммы и количества цифр натурального числа

№1.2.11. Вывести в строку 10 первых натуральных чисел, оканчивающихся на цифру k, кратных числу s и находящихся в интервале, левая граница которого равна start.

Решение:

```
# Задание task_01_2_11.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

start = int(input("Введите левую границу интервала: "))
k = int(input("Введите окончание натурального числа k: "))
s = int(input("Введите кратность натурального числа s: "))
n_count = 0

# Удалите комментарий и допишите код

current = start
result = [] # список для хранения найденных чисел

while n_count < 10:
    # Проверяем условия: число оканчивается на k, кратно s,
    # и находится в интервале [start, ...]
    if current % 10 == k and current % s == 0:
        result.append(str(current))
        n_count += 1
        current += 1

# Выводим числа в строку через пробел
print(" ".join(result))
```

---

Введите левую границу интервала: 100  
Введите окончание натурального числа k: 7  
Введите кратность натурального числа s: 9  
117 207 297 387 477 567 657 747 837 927

Рис. 11. Результат выполнения программы

№1.2.12. Даны целые числа a и b (a может быть больше b). Напечатайте:

- числа от минимального до максимального в строчку (разделяя пробелом);
- числа от максимального до минимального «столбиком».

Решение:

```

# Задание task_01_2_12.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

a = int(input("Введите число a"))
b = int(input("Введите число b"))

if a == b:
    print("Числа равны! Пожалуйста, введите различные числа.")
else:
    maximum = max(a, b)
    minimum = min(a, b)
    numbers_asc = []
    for i in range(minimum, maximum + 1):
        numbers_asc.append(str(i))
    print(" ".join(numbers_asc))

# 2. Числа от максимального до минимального столбиком
print("\nЧисла от максимального до минимального:")
for i in range(maximum, minimum - 1, -1):
    print(i)

# Удалите комментарий и допишите код

```

```

➡ Введите число a1
Введите число b5
1 2 3 4 5

Числа от максимального до минимального:
5
4
3
2
1

```

Рис. 12. Результат выполнения программы

№1.2.13. Для введенных с клавиатуры положительных целых чисел  $a$  и  $b$  ( $a \leq b$ ) определите:

- сумму всех целых чисел от  $a$  до  $b$ ;
- произведение всех целых чисел от  $a$  до  $b$ ;
- среднее арифметическое всех целых чисел от  $a$  до  $b$ ;
- среднее геометрическое нечетных чисел от  $a$  до  $b$ . Отрезок поиска включает сами числа  $a$  и  $b$ . При выводе вещественных результатов оставьте два знака после запятой.

Решение:

```
# Задание task_01_2_13.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

a = int(input("Введите число a: "))
b = int(input("Введите число b: "))

# Проверяем условие  $a \leq b$ 
if a > b:
    print("Ошибка: a должно быть меньше или равно b")
else:
    n_sum = 0
    n_mult = 1
    count = 0
    odd_count = 0
    odd_mult = 1

    # Проходим по всем числам от a до b включительно
    for num in range(a, b + 1):
        n_sum += num
        n_mult *= num
        count += 1

        # Для нечетных чисел считаем произведение для среднего геометрического
        if num % 2 != 0:
            odd_mult *= num
            odd_count += 1

    # Среднее арифметическое
    n_avg = n_sum / count if count > 0 else 0

    # Среднее геометрическое нечетных чисел
    if odd_count > 0:
        n_avg_geom = odd_mult ** (1 / odd_count)
    else:
        n_avg_geom = 0

    print("Сумма =", n_sum)
    print("Произведение =", n_mult)
    print("Среднее арифметическое = {:.2f}".format(n_avg))
    print("Среднее геометрическое нечетных чисел = {:.2f}".format(n_avg_geom))
```

```
Введите число a: 1
Введите число b: 5
Сумма = 15
Произведение = 120
Среднее арифметическое = 3.00
Среднее геометрическое нечетных чисел = 2.47
```

Рис. 13. Результат выполнения программы

№1.2.14. Начав тренировки, лыжник в первый день пробежал  $s$  км. ( $s > 0$ , вещественное число). Каждый следующий день он увеличивал пробег на  $p$  % ( $0 < p \leq 100$ , вещественное число) от пробега предыдущего дня. Определите:

- пробег лыжника за второй, третий, ..., десятый день тренировок;
- какой суммарный путь он пробежал за первые 10 дней тренировок.

При выводе вещественных результатов оставьте один день после запятой.

Решение:

```
# Задание task_01_2_14.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

s = float(input("Введите пробег в первый день (км): "))
p = float(input("Введите процент увеличения ( $0 < p \leq 100$ ): "))

total = s # суммарный пробег за 10 дней
current = s # пробег текущего дня

print("Пробег по дням:")
print("День 1: {:.1f} км".format(current))

# Рассчитываем пробег за 2-10 дни
for day in range(2, 11):
    current = current * (1 + p / 100) # увеличиваем пробег на p%
    total += current # добавляем к суммарному пробегу
    print("День {}: {:.1f} км".format(day, current))

print("Суммарный пробег: {:.1f} км.".format(total))
```

```
➡ Введите пробег в первый день (км): 5
Введите процент увеличения ( $0 < p \leq 100$ ): 10
Пробег по дням:
День 1: 5.0 км
День 2: 5.5 км
День 3: 6.1 км
День 4: 6.7 км
День 5: 7.3 км
День 6: 8.1 км
День 7: 8.9 км
День 8: 9.7 км
День 9: 10.7 км
День 10: 11.8 км
Суммарный пробег: 79.7 км.
```

Рис. 14. Результат выполнения программы

№1.2.15. Известна масса каждого предмета в кг., загружаемого в грузовик.

Определить, возможна ли перевозка груза, если грузоподъемность грузовика равна  $p$  кг.

Решение:

```
# Задание task_01_2_15.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

p = float(input("Введите грузоподъемность грузовика (кг): "))
n = int(input("Введите количество предметов: "))

total = 0

# Вводим массу каждого предмета и суммируем
for i in range(n):
    mass = float(input("Введите массу предмета {} (кг): ".format(i + 1)))
    total += mass

# Проверяем возможность перевозки
if total <= p:
    print("Перевозка возможна. Общая масса груза: {:.1f} кг".format(total))
else:
    print("Перевозка невозможна. Общая масса груза: {:.1f} кг превышает грузоподъемность {:.1f} кг".format(total, p))
```

```
Введите грузоподъемность грузовика (кг): 10
Введите количество предметов: 2
Введите массу предмета 1 (кг): 3
Введите массу предмета 2 (кг): 3
Перевозка возможна. Общая масса груза: 6.0 кг
```

Рис. 15. Результат выполнения программы

№1.2.16. В области несколько районов. Заданы площади, засеваемые пшеницей (га.), и средняя урожайность (ц/га) в каждом районе. Определите количество пшеницы, собранное по области. При выводе вещественных результатов оставьте один знак после запятой.

Решение:

```

# Задание task_01_2_16.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

n = int(input("Введите количество районов: "))

total = 0

# Вводим данные по каждому району и считаем общий урожай
for i in range(n):
    print(f"Район {i + 1}:")
    area = float(input("    Площадь (га): "))
    yield_per_hectare = float(input("    Урожайность (ц/га): "))

    # Считаем урожай с текущего района и добавляем к общему
    district_harvest = area * yield_per_hectare
    total += district_harvest

print("Общее количество пшеницы по области: {:.1f} ц".format(total))

```

---

```

➡ Введите количество районов: 3
Район 1:
    Площадь (га): 4
    Урожайность (ц/га): 2
Район 2:
    Площадь (га): 7
    Урожайность (ц/га): 3
Район 3:
    Площадь (га): 9
    Урожайность (ц/га): 4
Общее количество пшеницы по области: 65.0 ц

```

Рис. 16. Результат выполнения программы

№1.2.17. Решите задачу № 2.7, организовав бесконечный цикл, который бы прерывался при выполнении условия, используя оператор `break` .

Решение:

```

# Задание task_01_2_17.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

nums_sum = 0 # сумма
nums_count = 0 # количество

print("Вводите числа (0 для завершения):")


while True: # бесконечный цикл
    x = int(input("Введите число: "))

    if x == 0: # условие прерывания цикла
        break

    nums_sum += x
    nums_count += 1

```

```
print("Сумма чисел:", nums_sum)
print("Количество чисел:", nums_count)
```



```
Вводите числа (0 для завершения):
Введите число: 1
Введите число: 2
Введите число: 3
Введите число: 4
Введите число: 5
Введите число: 6
Введите число: 0
Сумма чисел: 21
Количество чисел: 6
```

Рис. 17. Результат выполнения программы

№1.2.18. Предложение, введенное с клавиатуры, содержит слова из гласных и согласных букв кириллицы (регистр может быть различный), а также пробелы. Определите количество гласных и согласных букв в предложении. Для пропуска пробелов используйте оператор `continue`.

Решение:

```
# Задание task_01_2_18.
#
# Выполнил: Губайдулина А. И.
# Группа: АБП-231

sentence = input("Введите предложение: ")

count_gl = 0 # Кол-во гласных
count_sogl = 0 # Кол-во согласных

# Списки гласных и согласных букв кириллицы (в нижнем регистре)
vowels = 'аеёиоуыэюя'
consonants = 'бвгджзйклмнпрстфхцщш'

# Приводим предложение к нижнему регистру для удобства проверки
sentence_lower = sentence.lower()

for char in sentence_lower:
    # Пропускаем пробелы
    if char == ' ':
        continue

    # Проверяем, является ли символ гласной
    if char in vowels:
        count_gl += 1
    # Проверяем, является ли символ согласной
    elif char in consonants:
        count_sogl += 1

print("Количество гласных букв:", count_gl)
print("Количество согласных букв:", count_sogl)
print("Всего букв:", count_gl + count_sogl)
```



```
✓ Введите предложение: Программирование
Количество гласных букв: 7
Количество согласных букв: 9
Всего букв: 16
```

Рис. 18. Результат выполнения программы

№1.2.19. Выведите на экран (в строку) все целые числа от  $a$  до  $b$ , кратные некоторому числу  $c$ .

Решение:

```
# Задание task_01_2_19.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

a = int(input("Введите число a: "))
b = int(input("Введите число b: "))
c = int(input("Введите число c: "))

# Определяем начальное число (первое кратное c, не меньшее a)
start = a
if a % c != 0:
    start = a + (c - a % c)

# Выводим числа в строку через пробел
result = []
for num in range(start, b + 1, c):
    result.append(str(num))

print(" ".join(result))
```

```
➡ Введите число a: 1
Введите число b: 10
Введите число c: 2
2 4 6 8 10
```

Рис. 19. Результат нахождения чисел, кратных числу  $c$

№1.2.20. Выведите на экран (в строку) все трехзначные натуральные числа, сумма цифр которых равна целому числу  $n$  ( $0 < n \leq 27$ ).

Решение:

```

# Задание task_01_2_20.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

n = int(input("Введите число n ( $0 < n \leq 27$ ): "))

result = []
# Перебираем все трехзначные числа от 100 до 999
for num in range(100, 1000):
    # Разбиваем число на цифры
    digit1 = num // 100      # первая цифра
    digit2 = (num // 10) % 10 # вторая цифра
    digit3 = num % 10        # третья цифра

    # Проверяем сумму цифр
    if digit1 + digit2 + digit3 == n:
        result.append(str(num))

# Выводим результат в строку через пробел
if result:
    print(" ".join(result))
else:
    print(f"Нет трехзначных чисел с суммой цифр {n}")

```

---

```

➞ Введите число n ( $0 < n \leq 27$ ): 3
102 111 120 201 210 300

```

Рис. 20. Результат выполнения программы

№1.2.21. Известно количество учеников в классе и их рост (см.); рост мальчиков условно задан отрицательными числами. Определите средний рост мальчиков и средний рост девочек. При выводе вещественных результатов оставьте один знак после запятой.

Решение:

```

# Задание task_01_2_21.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

n = int(input("Введите количество учеников в классе: "))

r_sr_m = 0.0 # средний рост мальчиков
r_sr_d = 0.0 # средний рост девочек

count_m = 0 # количество мальчиков
count_d = 0 # количество девочек
sum_m = 0 # сумма роста мальчиков
sum_d = 0 # сумма роста девочек

# Вводим рост каждого ученика и разделяем на мальчиков и девочек
for i in range(n):
    height = int(input(f"Введите рост ученика {i+1} (см): "))

    if height < 0:
        # Мальчик - отрицательный рост, берем модуль
        sum_m += abs(height)
        count_m += 1
    else:
        # Девочка - положительный рост
        sum_d += height
        count_d += 1

# Вычисляем средний рост (избегаем деления на ноль)
if count_m > 0:
    r_sr_m = sum_m / count_m
if count_d > 0:
    r_sr_d = sum_d / count_d

print("Средний рост мальчиков: {:.1f}".format(r_sr_m))
print("Средний рост девочек: {:.1f}".format(r_sr_d))

```

```

> Введите количество учеников в классе: 3
Введите рост ученика 1 (см): 180
Введите рост ученика 2 (см): 150
Введите рост ученика 3 (см): -170
Средний рост мальчиков: 170.0
Средний рост девочек: 165.0

```

Рис. 21. Результат нахождения среднего роста мальчиков и девочек

№1.2.22. Даны  $n$  вещественных чисел. Определите максимальное и минимальное из них. При выводе вещественных результатов оставьте два знака после запятой.

Решение:

```

# Задание task_01_2_22.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

n = int(input("Введите количество чисел: "))

a_max = None
a_min = None

# Вводим числа и находим максимум и минимум
for i in range(n):
    num = float(input(f"Введите число {i+1}: "))

    # Инициализируем максимум и минимум первым числом
    if a_max is None:
        a_max = num
        a_min = num
    else:
        # Обновляем максимум и минимум
        if num > a_max:
            a_max = num
        if num < a_min:
            a_min = num

print("Максимум: {:.2f}".format(a_max))
print("Минимум: {:.2f}".format(a_min))

```

```

➤ Введите количество чисел: 5
Введите число 1: 1
Введите число 2: 2
Введите число 3: 3
Введите число 4: 4
Введите число 5: 5
Максимум: 5.00
Минимум: 1.00

```

Рис. 22. Результат определения максимального и минимального вещественных чисел

№1.2.23. Дано натуральное число  $n$ . Определите, является ли оно членом последовательности Фибоначчи.

Решение:

```

# Задание task_01_2_23.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

n = int(input("Введите натуральное число n: "))

# Первые два числа Фибоначчи
a, b = 0, 1

# Проверяем, является ли n одним из первых двух чисел
if n == 0 or n == 1:
    print("Является")
else:
    # Генерируем последовательность Фибоначчи, пока не превысим n
    is_fibonacci = False
    while b <= n:
        if b == n:
            is_fibonacci = True
            break
        a, b = b, a + b

    print("Является" if is_fibonacci else "Не является")

```

---

```

Введите натуральное число n: 7
Не является

```

Рис. 23. Результат определения числа последовательности Фибоначчи.

№1.2.24. Дано  $n$  вещественных чисел. Определите, является ли последовательность упорядоченной по возрастанию. В случае отрицательного ответа выведите порядковый номер числа, нарушающего такую упорядоченность.

Решение:

```

# Задание task_01_2_24.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

n = int(input("Введите количество чисел: "))

index = -1

prev = float(input("1-е число = "))
for i in range(2, n + 1):
    current = float(input(f"{i}-е число = "))

```

```

# Проверяем, нарушает ли текущее число порядок возрастания
if current <= prev:
    index = i # сохраняем индекс нарушающего элемента
    break

prev = current # обновляем предыдущее число

if index == -1:
    print("Последовательность упорядочена по возрастанию")
else:
    print(f"Последовательность не упорядочена по возрастанию")
    print(f"Элемент №{index} нарушает порядок")

```

---

Введите количество чисел: 4  
 1-е число = 1  
 2-е число = 6  
 3-е число = 7  
 4-е число = 4  
 Последовательность не упорядочена по возрастанию  
 Элемент №4 нарушает порядок

Рис. 24. Результат выполнения программы

№1.2.25. Выведите на экран таблицу умножения на  $n$  ( $2 < n \leq 9$ )

Решение:

```

# Задание task_01_2_25.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

n = int(input("Введите число n (2 < n ≤ 9): "))

# Проверяем условие 2 < n ≤ 9
if n <= 2 or n > 9:
    print("Число должно быть в диапазоне: 2 < n ≤ 9")
else:
    # Внешний цикл для второго множителя (столбцы)
    for j in range(1, 10):
        # Внутренний цикл для первого множителя (строки)
        for i in range(1, n + 1):
            result = i * j
            print(f"{i} × {j} = {result}", end="    ")
        print() # Переход на новую строку после каждого столбца
    print() # Пустая строка между блоками

```

```

Введите число n (2 < n ≤ 9): 5
1 x 1 = 1   2 x 1 = 2   3 x 1 = 3   4 x 1 = 4   5 x 1 = 5

1 x 2 = 2   2 x 2 = 4   3 x 2 = 6   4 x 2 = 8   5 x 2 = 10

1 x 3 = 3   2 x 3 = 6   3 x 3 = 9   4 x 3 = 12   5 x 3 = 15

1 x 4 = 4   2 x 4 = 8   3 x 4 = 12   4 x 4 = 16   5 x 4 = 20

1 x 5 = 5   2 x 5 = 10   3 x 5 = 15   4 x 5 = 20   5 x 5 = 25

1 x 6 = 6   2 x 6 = 12   3 x 6 = 18   4 x 6 = 24   5 x 6 = 30

1 x 7 = 7   2 x 7 = 14   3 x 7 = 21   4 x 7 = 28   5 x 7 = 35

1 x 8 = 8   2 x 8 = 16   3 x 8 = 24   4 x 8 = 32   5 x 8 = 40

1 x 9 = 9   2 x 9 = 18   3 x 9 = 27   4 x 9 = 36   5 x 9 = 45

```

Рис. 25. Результат выполнения программы

№1.2.26. Выведите графическое изображения делимости чисел от 1 до n (значение n вводится с клавиатуры) - в каждой строке напечатайте очередное число и столько символов \*, сколько делителей у этого числа.

Решение:

```

# Задание task_01_2_26.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231
n = int(input("Введите число n: "))

# Внешний цикл для чисел от 1 до n
for number in range(1, n + 1):
    count = 0 # счетчик делителей

    # Внутренний цикл для поиска делителей
    for divisor in range(1, number + 1):
        if number % divisor == 0:
            count += 1

    # Выводим число и соответствующее количество звездочек
    print(f"{number} {'*' * count}")

```

```

➤ Введите число n: 10
1 *
2 **
3 **
4 ***
5 **
6 ****
7 **
8 ****
9 ***
10 ****

```

Рис. 26. Результат выполнения программы

№1.2.27. Выведите на экран (в строку) n первых простых чисел.

Решение:

```
# Задание task_01_2_27.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

n = int(input("Введите количество простых чисел: "))

count = 0 # счетчик найденных простых чисел
number = 2 # первое простое число

print("Первые", n, "простых чисел:")

while count < n:
    is_prime = True # флаг простоты числа

    # Проверяем, является ли число простым
    for divisor in range(2, int(number**0.5) + 1):
        if number % divisor == 0:
            is_prime = False
            break

    # Если число простое, добавляем его и увеличиваем счетчик
    if is_prime:
        print(number, end=" ")
        count += 1

    number += 1
```

---

Введите количество простых чисел: 3  
Первые 3 простых чисел:  
2 3 5

Рис. 27. Результат вывода n первых простых чисел

№1.2.28. Составьте программу для нахождения всех натуральных решений уравнения  $x^2 + y^2 + z^2 = k^2$ , где  $x, y, z \in [1, 30]$ , а  $k$  вводится с клавиатуры.

Решение:



```

# Задание task_01_2_28.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

k = int(input("Введите число k: "))

solutions = [] # список для хранения решений

# Перебираем все возможные значения x, y, z от 1 до 30
for x in range(1, 31):
    for y in range(1, 31):
        for z in range(1, 31):
            # Проверяем уравнение  $x^2 + y^2 + z^2 = k^2$ 
            if x*x + y*y + z*z == k*k:
                solutions.append((x, y, z))

# Выводим результаты
if solutions:
    print(f"Найдено {len(solutions)} решений уравнения  $x^2 + y^2 + z^2 = {k}^2$ :")
    for solution in solutions:
        print(f"x={solution[0]}, y={solution[1]}, z={solution[2]}")
else:
    print(f"Решений уравнения  $x^2 + y^2 + z^2 = {k}^2$  в диапазоне [1,30] не найдено")

```

```

Введите число k: 30
Найдено 15 решений уравнения  $x^2 + y^2 + z^2 = 30^2$ :
x=4, y=10, z=28
x=4, y=20, z=22
x=4, y=22, z=20
x=4, y=28, z=10
x=10, y=4, z=28
x=10, y=20, z=20
x=10, y=28, z=4
x=20, y=4, z=22
x=20, y=10, z=20
x=20, y=20, z=10
x=20, y=22, z=4
x=22, y=4, z=20
x=22, y=20, z=4
x=28, y=4, z=10
x=28, y=10, z=4

```

Рис. 28. Результат выполнения программы

№1.2.29. Дан список из  $n$  вещественных чисел, введенных с клавиатуры (среди чисел есть по крайней мере одно положительное и отрицательное число). Сформируйте из него 2 списка:

- положительных чисел, используя списковые включения;

- отрицательных чисел, не используя списковые включения. Выведите на экран:
- исходный список;
- получившиеся списки;
- среднее арифметическое первого списка и среднее геометрическое второго списка. При выводе вещественных результатов оставьте два знака после запятой.

Решение:

```
# Задание task_01_2_29.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

n = int(input("n = "))

nums = [] # исходный список
for i in range(n):
    num = float(input(f"Введите число {i+1}: "))
    nums.append(num)

# Списковое включение для положительных чисел
nums_pos = [num for num in nums if num > 0]

# Без спискового включения для отрицательных чисел
nums_neg = []
for num in nums:
    if num < 0:
        nums_neg.append(num)

# Среднее арифметическое положительных чисел
if nums_pos:
    sr_ar = sum(nums_pos) / len(nums_pos)
else:
    sr_ar = 0

# Среднее геометрическое отрицательных чисел
if nums_neg:
    product = 1
    for num in nums_neg:
        product *= abs(num) # берем модуль, т.к. отрицательные числа
    sr_geom = product ** (1 / len(nums_neg))
```

```

else:
    sr_geom = 0

# Вывод результатов
print("Исходный список:", [f"{x:.2f}" for x in nums])
print("Положительные числа:", [f"{x:.2f}" for x in nums_pos])
print("Отрицательные числа:", [f"{x:.2f}" for x in nums_neg])
print("Среднее арифметическое положительных: {:.2f}".format(sr_ar))
print("Среднее геометрическое отрицательных: {:.2f}".format(sr_geom))

```

---

```

n = 4
Введите число 1: 1
Введите число 2: 6
Введите число 3: -10
Введите число 4: -3
Исходный список: ['1.00', '6.00', '-10.00', '-3.00']
Положительные числа: ['1.00', '6.00']
Отрицательные числа: ['-10.00', '-3.00']
Среднее арифметическое положительных: 3.50
Среднее геометрическое отрицательных: 5.48

```

Рис. 29. Результат выполнения программы

№1.2.30. Дан список целых чисел, введенных с клавиатуры (длина неизвестна). Ответьте на вопросы:

- являются ли все элементы положительными числами?
- есть ли хотя бы один нулевой элемент в списке?
- являются ли все элементы четными числами?
- есть ли хотя бы один нечетный элемент в списке? Каждый из пунктов выполните дважды: используя стандартный проход в цикле (например, через алгоритм с флажком), и используя функции `any()` и/или `all()` .

Решение:

```

# Задание task_01_2_30.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

# Все разделенные пробелом элементы будут преобразованы в список целых чисел
nums = [int(item) for item in input().split()]

# 1. Все положительные
all_pos_1 = True
for item in nums:
    if item <= 0:
        all_pos_1 = False
        break

all_pos_2 = all([item > 0 for item in nums])

# 2. Хотя бы 1 нулевой элемент
any_zero_1 = False
for item in nums:
    if item == 0:
        any_zero_1 = True
        break

any_zero_2 = any([item == 0 for item in nums])

# 3. Все четные
all_even_1 = True
for item in nums:
    if item % 2 != 0:
        all_even_1 = False
        break

all_even_2 = all([item % 2 == 0 for item in nums])

# 4. Хотя бы 1 нечетный элемент
any_odd_1 = False
for item in nums:
    if item % 2 != 0:
        any_odd_1 = True
        break

any_odd_2 = any([item % 2 != 0 for item in nums])

print("Все положительные:", all_pos_1, all_pos_2)
print("Хотя бы 1 нулевой элемент:", any_zero_1, any_zero_2)
print("Все четные:", all_even_1, all_even_2)
print("Хотя бы 1 нечетный элемент:", any_odd_1, any_odd_2)

```

---

```

-1 1 100 0
Все положительные: False False
Хотя бы 1 нулевой элемент: True True
Все четные: False False
Хотя бы 1 нечетный элемент: True True

```

Рис. 30. Результат выполнения программы

№1.2.31. Дано предложение. Выведите его на экран, удалив из него все слова, содержащие произвольную букву (вводится с клавиатуры).

Решение:

```
# Задание task_01_2_31.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

s = input("Введите предложение: ")
k = input("Введите букву для удаления: ")

# Разбиваем предложение на слова
words = s.split()

# Создаем новый список слов, которые НЕ содержат букву k
filtered_words = []
for word in words:
    if k.lower() not in word.lower(): # проверяем без учета регистра
        filtered_words.append(word)

# Собираем отфильтрованное предложение обратно
result = ' '.join(filtered_words)

print("Результат:", result)
```

---

```
Введите предложение: МАМА мыла РаМу
Введите букву для удаления: Р
Результат: МАМА мыла
```

Рис. 31. Результат выполнения программы

№1.2.32. В зрительном зале кинотеатра  $n$  рядов, количество мест в которых может меняться. Разработчик смоделировал занятость мест как двумерный массив (список из списков), где каждый вложенный список содержит информацию о проданных местах в соответствующем ряду (1 - занято, 0 - свободно). Напишите программу, которая позволит пользователю увидеть количество свободных мест, а также, введя номер ряда и места, получить информацию - свободно оно или нет. Данные о занятости мест вводятся с клавиатуры (набор из 0 и 1 для каждого ряда).

Решение:

```

# Задание task_01_2_32.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

n = int(input("Введите количество рядов: "))

# 1. Заполнение мест
seats = []
for i in range(n):
    row = input(f"Введите занятость мест для ряда {i+1} (0 - свободно, 1 - занято, через пробел): ").split()
    row = [int(seat) for seat in row]
    seats.append(row)

# 2. Всего свободных мест
count = 0
for row in seats:
    for seat in row:
        if seat == 0:
            count += 1

print(f"Всего свободных мест: {count}")

# 3. Ввод значений для поиска
n_p, m_p = [int(item) for item in input("Введите ряд и место через пробел: ").split()]

# Корректируем индексы (пользователь считает с 1, а Python с 0)
row_index = n_p - 1
seat_index = m_p - 1

# Проверяем границы
if 0 <= row_index < len(seats) and 0 <= seat_index < len(seats[row_index]):
    if seats[row_index][seat_index] == 0:
        print(f"Место {m_p} в ряду {n_p} свободно")
    else:
        print(f"Место {m_p} в ряду {n_p} занято")
else:
    print("Такого места не существует")

```

```

Введите количество рядов: 3
Введите занятость мест для ряда 1 (0 - свободно, 1 - занято, через пробел): 0
Введите занятость мест для ряда 2 (0 - свободно, 1 - занято, через пробел): 0
Введите занятость мест для ряда 3 (0 - свободно, 1 - занято, через пробел): 0
Всего свободных мест: 3
Введите ряд и место через пробел: 1 0
Такого места не существует

```

Рис. 32. Результат выполнения программы

№1.2.33. Вводится список из n сотрудников в формате: Фамилия Имя

Отчество Пол Стаж где:

- все значения разделены пробелом и сами не содержат пробелов;
- Пол : "М" или "Ж" ;

- Стаж : количество полных лет, отработанных в компании.

Далее:

- определите самого «молодого» и самого «старого» сотрудника, используя функцию sorted() ;
- сформируйте 2 отдельных списка: мужчин и женщин и ответьте, в каком из списков больше имен, начинающихся на букву k (вводится с клавиатуры).

Решение:

```
# Задание task_01_2_33.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

# 1. Заполнение списка
n = int(input("Введите количество сотрудников: "))

employees = []
for i in range(n):
    data = input(f"Введите данные сотрудника {i+1} (Фамилия Имя Отчество Пол Стаж): ").split()
    # Преобразуем стаж в число
    data[4] = int(data[4])
    employees.append(data)

# 2. Самый «молодой» и самый «старый» сотрудник
# Сортируем по стажу (по возрастанию)
sorted_by_exp = sorted(employees, key=lambda x: x[4])
employee_min = sorted_by_exp[0] # самый молодой (меньший стаж)
employee_max = sorted_by_exp[-1] # самый старый (большой стаж)

print("Самый \молодой\: {}".format(" ".join(employee_min[:3])))
print("Самый \старый\: {}".format(" ".join(employee_max[:3])))

# 3. Отдельные списки мужчин и женщин
men = [emp for emp in employees if emp[3] == "М"]
women = [emp for emp in employees if emp[3] == "Ж"]

k = input("Введите букву k: ")

# Подсчет мужчин с именем на букву k
men_k_count = 0
for man in men:
    if man[1].startswith(k.upper()) or man[1].startswith(k.lower()):
```

```
men_k_count += 1

# Подсчет женщин с именем на букву k
women_k_count = 0
for woman in women:
    if woman[1].startswith(k.upper()) or woman[1].startswith(k.lower()):
        women_k_count += 1

# Вывод результатов
print(f"Мужчин: {len(men)}, из них с именем на '{k}': {men_k_count}")
print(f"Женщин: {len(women)}, из них с именем на '{k}': {women_k_count}")

if men_k_count > women_k_count:
    print(f"Больше имен на '{k}' среди мужчин")
elif women_k_count > men_k_count:
    print(f"Больше имен на '{k}' среди женщин")
else:
    print(f"Одинаковое количество имен на '{k}'")
```

Введите количество сотрудников: 3  
Введите данные сотрудника 1 (Фамилия Имя Отчество Пол Стаж): Иванов Иван Иванович М 5  
Введите данные сотрудника 2 (Фамилия Имя Отчество Пол Стаж): Петров Алексей Семенович М 8  
Введите данные сотрудника 3 (Фамилия Имя Отчество Пол Стаж): Михалкова Анна Сергеевна Ж 7  
Самый "молодой": Иванов Иван Иванович  
Самый "старый": Петров Алексей Семенович  
Введите букву k: а  
Мужчин: 2, из них с именем на 'а': 1  
Женщин: 1, из них с именем на 'а': 1  
Одинаковое количество имен на 'а'

Рис. 33. Результат выполнения программы

№1.2.34. Вводится список из  $n$  годовых вкладов, предлагаемых банками, в формате: Банк Сумма Процент где:

- все значения разделены пробелом и сами не содержат пробелов;
- наименование банка уникально;
- Сумма : сумма для открытия вклада в руб. (целое число,  $>0$ );
- Процент : годовой процент по вкладу (вещественное число,  $(0,100]$ ).

Далее определите (гарантируется, что искомый банк - один):

- самый доступный банк (с наименьшей первоначальной суммой);



- самый выгодный банк, принимая, что за год прибыль = сумма \* процент / 100 . При выводе финансовых значений оставьте два знака после запятой.

Решение:

```
# Задание task_01_2_34.
#
# Выполнил: Губайдуллина А. И.
# Группа: АБП-231

# 1. Заполнение списка
n = int(input("Введите количество банков: "))

deposits = []
for i in range(n):
    data = input(f"Введите данные банка {i+1} (Банк Сумма Процент): ").split()
    # Создаем словарь для каждого банка
    bank_data = {
        "name": data[0],
        "initial_sum": int(data[1]),
        "rate": float(data[2])
    }
    deposits.append(bank_data)

# 2. Самый доступный банк (с наименьшей первоначальной суммой)
min_sum_bank = min(deposits, key=lambda x: x["initial_sum"])
print(f"Самый доступный банк: {min_sum_bank['name']} (сумма: {min_sum_bank['initial_sum']} руб.)")

# 3. Самый выгодный банк (максимальная прибыль = сумма * процент / 100)
max_profit_bank = max(deposits, key=lambda x: x["initial_sum"] * x["rate"] / 100)
profit = max_profit_bank["initial_sum"] * max_profit_bank["rate"] / 100
print(f"Самый выгодный банк: {max_profit_bank['name']} (прибыль: {profit:.2f} руб.)")
```

Введите количество банков: 3  
Введите данные банка 1 (Банк Сумма Процент): Сбербанк 50000 5.2  
Введите данные банка 2 (Банк Сумма Процент): ВТБ 60000 7.6  
Введите данные банка 3 (Банк Сумма Процент): Альфа 30000 3.7  
Самый доступный банк: Альфа (сумма: 30000 руб.)  
Самый выгодный банк: ВТБ (прибыль: 4560.00 руб.)

Рис. 34. Результат выполнения программы

**Вывод:** в ходе выполнения лабораторной работы изучила основные управляющие конструкции языка Python: условный оператор и циклы, научилась использовать управляющие структуры для решения задач различной сложности, закрепила навыки обработки данных с использованием коллекций и итераторов, развила умение комбинировать условия и циклы для оптимизации алгоритмов.