Коротко о Питон

Создание **Python** было начато Гвидо ван Россумом (Guido van Rossum) в 1991 году. **Python** является простым и, в то же время, мощным интерпретируемым объектно-ориентированным языком программирования. Он предоставляет структуры данных высокого уровня, имеет изящный синтаксис и использует динамический контроль типов, что делает его идеальным языком для быстрого написания различных приложений. Python поддерживает модули и пакеты, поощряя модульность и повторное использование кода. Интерпретатор Python и большая стандартная библиотека доступны бесплатно в виде исходных и исполняемых кодов для большинства распространенных платформ и могут свободно распространяться.

Структура программы. Встроенные типы

В языке Python важны отступы, поэтому инструкции (операторы), входящие в последовательность действий, должны иметь одинаковый отступ, а блоки определяются с использованием отступов: увеличение отступа указывает на начало, уменьшение – конец блока. Инструкции, которые предполагают наличие отступов, завершаются символом двоеточия (**:**).

Подпрограммы в Python оформляются только как функции. Функции объединяются в **модули** (библиотеки функций). **Модули** (также функции из модулей) при необходимости подключаются к пользовательским программам с помощью команды:

`import имя_модуля1`, ..., а отдельные функции - командой:

`from имя_модуля import функция1`, ...

Однострочные комментарии в Python начинаются с символа решетки (**#**), многострочные используют тройные кавычки (**`"""`** или **`'''`**) в начале и в конце комментария.

Основные встроенные типы:

- числовые – целые: **`int`** (обычное целое), **`long`** (целое произвольной точности), **`bool`** (логический); действительные: **`float`** (с плавающей точкой).
- последовательности – неизменчивые: **`str`** (строка), **`unicode`** (Unicode-строка), **`tuple`** (кортеж); изменчивые: **`list`** (список).
- отображения – **`dict`** (словарь).
- вызываемые объекты – **функции** (пользовательские и встроенные); функции-генераторы; методы (пользовательские и встроенные); модули; классы; экземпляры классов.
- файлы **`file`**.
- специальные и вспомогательные типы: **`None`**, **`NotImplemented`** и **`Ellipsis`**; **`buffer`**, **`slice`**.

Числовые типы

Диапазон значений типа **`int`** – от **-2147483648** до **2147483647**, а точность целых произвольной точности (**`long`**) зависит от объема доступной памяти. Числа могут быть представлены в системах счисления с основаниями **10**, **2**, **8**, или **16.** Примеры: **`15`**, **`0b1111`**, **`0o17`**, **`0xf`** (или **`0xF`**).

Числа с плавающей точкой (**`float`**). записываются либо через точку, либо в нотации с экспонентой.

Тип **`bool`** (подтип целочисленного типа для "канонического" обозначения логических величин) – принимает одно из двух значений: **`True`** (**`истина`**) или **`False`** (**`ложь`**).

Тип string и тип unicode

В Python строки бывают двух типов: **обычные** и **Unicode**-строки. Строки-константы в программе можно задать как с помощью апострофов (**`'`**), так и обычных двойных кавычек (**`"`**). Набор операций над строками включает конкатенацию "**`+`**", повтор "**`*`**", форматирование "**`%`**".

Имена (переменные) и преобразование типов

В Python не нужно объявлять тип переменной вручную, объявление происходит автоматически, когда вы присваиваете значение переменной.

Имена (идентификаторы) в Python могут начинаться с латинской буквы любого регистра (*регистр имеет значение*) или подчеркивания, а дальше допустимо использование цифр. В качестве идентификаторов нельзя использовать ключевые слова (при необходимости использования их в качестве имени, нужно добавить одиночное подчеркивание в конце таких слов). Имена, начинающиеся с подчеркивания или двух подчеркиваний, имеют особый смысл. Одиночное подчеркивание говорит программисту о том, что имя имеет местное применение, и не должно использоваться за пределами модуля.

Иногда целое число необходимо представить как строку, а строку, состоящую из цифр – в виде числа, чтобы с данными можно было выполнять необходимые (строковые или арифметические) операции. Для этого используются функции, одноименные с именем типа, то есть, **`int`** – для целых, **`float`** – для вещественных чисел, **`str`** – для строк. Например, **`int('123')`** вернет целое число **123**, а **`str(123)`** вернет строку '**123**'.

Присвоение, математические операторы и функции

- для присвоения значения переменным используется **оператор присваивания («=»).** Примеры записей:

```
n = 123
s = 'Python'
```

- можно использовать также **множественное присвоение**, где значения присваиваются сразу нескольким переменным. Пример записи:
  `a, b, c = 1, 2, 3`
- к **математическим операторам** относятся: **+** (сложение), **-** (вычитание), ***** (умножение), **/** (деление), **//** (целочисленное деление), **%** (остаток от деления), ** (возведение в степень), **-x** (смена знака). Значение переменных можно увеличить с помощью оператора «**+=**», уменьшить с «**-=**», указав в левой части переменную, а в правой – значение, на которую произойдет изменение.

  Последовательность выполнения действий: возведения в степень – ***справа налево***, унарные минусы (отрицания), умножения и деления, сложения и вычитания.
- функция **int(n[, b])** используется для получения десятичного числа [из строки, представляющей число с основанием системы счисления **b]**, **abs(x)** – для получения модуля числа, функция **round(n[, z])** округляет число **x** до заданного знака после (или до) точки;
- функция **sum(range([a,] b))** – позволяет получить сумму элементов [от **a]** до (**b - 1**), [x, y =] **divmod(a, b)** – возвращает результат (**x, y**), равный (**a / b, a % b**), **pow(x, y)** – **x** в степени **y**;
- функция **range([a,] b[, s])** создаёт список как арифметическую прогрессию от **a** до **b-1** шагом **s**.

Функции ввода-вывода

- функция **print** служит для вывода информации. Примеры записи:

  ```
  print(a + b, "O'key"),
  print(5, '"Отлично"', sep = ' - ', end = ''), где:
  ```

  параметр "**sep**" является разделителем между выводимыми значениями (параметр **sep = '\n'** осуществляет переход на новую строку. Если параметр **sep** опустить, значения разделяются одним пробелом);

  параметр **end** указывает, что выводится после вывода всех значений, перечисленных в функции **print** (по умолчанию параметр **end** означает **'\n'**, то есть, что следующий вывод будет происходить с новой строки, если **end = ''**, то перехода на новую строку не будет).
- для ввода данных используется функция **input("сообщение-подсказка")**. Примеры:

  a = input() – считывает строку с клавиатуры и присваивает переменной **a**;
  a = int(input()) – считывает строку с клавиатуры, преобразовывает в целое и присваивает переменной **a**.

- для ввода значений нескольких переменных в одной строке можно использовать инструкцию **input().split()**:

  a, b, c = input().split() – считывает значения 3-х строковых переменных **a**, **b** и **c**, вводимых с клавиатуры через пробел;

  a, b, c = map(int, input().split()) – считывает значения 3-х переменных, которые вводятся в одной строке, преобразовывает их в целые числа и присваивает переменным **a**, **b** и **c**.

Задания 1-16:

1. Напишите программу, которая вычисляет и выводит на экран сумму чисел **1**, **2**, **3**, **5**, **8**.

   решение тут
2. Напишите программу, которая вычисляет и выводит на экран значение выражения:

   $$(3 + 2) \times 6$$

   решение тут
3. Напишите программу, которая вычисляет и выводит на экран сумму чисел:

   $$\frac{1}{2}, \frac{1}{3}, \frac{1}{5}, \frac{1}{8}.$$

   решение тут
4. Напишите программу, которая вычисляет и выводит на экран значение выражения:

   $$2^1 + 2^2 + 2^3 + 2^4$$

   решение тут
5. Напишите программу, которая вычисляет и выводит на экран значение выражения:

   $$\sqrt{3 \times 3 + 4^2}$$

   решение тут
6. Напишите программу, которая выводит на экран число **42153.121232**:
   - округляя его до целого значения;
   - округляя его до **0,001**;
   - округляя его до **100**.

   Фрагмент экрана:

   ```
   42153
   42153.121
   42200
   ```

   решение тут
7. Напишите программу, которая вычисляет и выводит на экран сумму **s** чисел **a** и **b**. Значения переменным **a** и **b** присваиваются в тексте программы. Фрагмент экрана:

```
Сумма чисел 11 и 7 равна 18
```

решение тут

8. Напишите программу, которая вычисляет и выводит на экран произведение **m** чисел **a** и **b**. Значения переменным **a** и **b** присваиваются в тексте программы. Фрагмент экрана:

```
Произведение чисел 5 и 8 равно 40
```

решение тут

9. Напишите программу, которая вычисляет и выводит на экран площадь прямоугольника. Длина и ширина прямоугольника присваиваются в тексте программы. Фрагмент экрана:

```
Площадь прямоугольника со сторонами 6 и 5 равна 30
```

решение тут

10. Напишите программу, которая вычисляет и выводит на экран длину гипотенузы прямоугольного треугольника. Длины катетов присваиваются в тексте программы. Фрагмент экрана:

```
Гипотенуза прямоугольного треугольника с катетами 12 и 5 равна 13.0
```

решение тут

11. Напишите программу, которая вычисляет и выводит на экран разность **r** целых чисел **a** и **b**. Значения переменных **a** и **b** вводятся с клавиатуры. Фрагмент экрана:

```
Вычисление разности
Введите число "a": 15
Введите число "b": 6
Разность чисел 15 и 6 равна 9
```

решение тут

12. Напишите программу, которая вычисляет и выводит на экран частное **d** от деления действительных чисел **a** на **b** (***b** не равно **0!***) с точностью **0,01**. Значения переменных **a** и **b** вводятся с клавиатуры. Фрагмент экрана:

```
Вычисление частного
Введите число "a": 91.45
Введите число "b": 17.3
Частное от деления числа 91.45 на 17.3 равно 5.29
```

решение тут

13. Напишите программу, которая вычисляет и выводит на экран площадь треугольника с точностью **0,1**. Длина основания и высота треугольника – действительные числа и вводятся с клавиатуры. Фрагмент экрана:

```
Вычисление площади треугольника
Введите длину основания: 14.8
Введите высоту: 9.2
Площадь треугольника равна 68.1
```

решение тут

14. Напишите программу, которая вычисляет и выводит на экран высоту прямоугольного треугольника с точностью **0,1**. Длины катетов треугольника вводятся с клавиатуры. Фрагмент экрана:

```
Вычисление высоты прямоугольного треугольника
Введите длину катета 1: 10.5
Введите длину катета 2: 5.6
Высота треугольника равна 4.9
```

решение тут

15. Напишите программу вычисления площади треугольника по трём его сторонам с точностью **0,1** (площадь треугольника по известным трём его сторонам можно определить по формуле Герона: $s = \sqrt{p * (p - a) * (p - b) * (p - c)}$, где **p** – полупериметр треугольника). Длины сторон треугольника вводятся с клавиатуры. Фрагмент экрана:

```
Вычисление площади треугольника
Введите через пробел длины 3-х сторон: 7.5 8.1 4.5
Площадь треугольника равна 16.7
```

решение тут

16. Напишите программу, которая преобразует введенное с клавиатуры дробное число в денежный формат. Фрагмент экрана:

```
Преобразование числа в денежный формат
Введите дробное число: 145.2567
145.26 руб. – это 145 руб. 26 коп.
```

решение тут

Функции math и random

- функция **pi** возвращает значение числа «**пи**», **ceil(x)** – наименьшее целое, большее или равное **x**, **floor(x)** – наибольшее целое, меньшее или равное **x**;
- функция **sqrt(x)** – возвращает квадратный корень числа **x**, **log10(x)** – десятичный логарифм числа **x**, **log(x)** – натуральный логарифм числа **x**;
- функции **sin(x)**, **cos(x)**, **tan(x)**, **asin(x)**, **acos(x)**, **atan(x)** используются для получения синуса, косинуса, тангенса, арксинуса, арккосинуса или арктангенса числа **x** соответственно;
- функция **random()** возвращает случайное вещественное число в диапазоне от **0** до **1**, **randrange([start,] stop[, step])** – целое число из диапазона **start** – (**stop - 1**) шаг: **step**, **choice(s)** – случайный элемент из последовательности **s**.

  Примеры записей:

```
import math, random
print(math.pi)
print(math.ceil(x))
print(math.sin(math.pi / 2))
print(random.randrange(1, 8, 2))
```

Задания 17-21:

17. Напишите программу, которая выводит на экран число **пи** с точностью **0,001**. Фрагмент экрана:

```
3.142
```

решение тут

18. Напишите программу, которая выводит на экран квадратный корень десятичного логарифма числа **10000** и натуральный логарифм числа **25** с точностью **0,01**. Фрагмент экрана:

```
2.0
3.22
```

решение тут

19. Напишите программу, которая выводит на экран синус, косинус и котангенс угла **30°** с точностью **0,001**. Фрагмент экрана:

```
синус угла 30 градусов - 0.5
косинус 30 градусов - 0.866
котангенс угла 30 градусов - 1.732
```

решение тут

20. Напишите программу, которая вычисляет и выводит арктангенс числа **2,5** в градусах с точностью **1**. Фрагмент экрана:

```
арктангенс числа 2,5 - 68 градусов.
```

решение тут

21. Напишите программу, которая выводит случайное нечётное число от 2 до 9. Фрагмент экрана:

```
7 - случайное нечётное число от 2 до 9.
```

решение тут

Оператор условия и выбора (инструкция if)

- инструкция **if** используется для выбора одного из двух (или нескольких) серий инструкций, выполняемых во время работы программы в зависимости от соблюдения заданных условий. Пример записи:

```
if a > b:
    c = a
else:
    c = b
```

- в Python нет инструкции **case**, вместо неё можно использовать конструкцию **if-elif-else**:

```
if a == 1:
    print('один')
elif a == 2:
    print('два')
elif a == 3:
    print('три')
...
else:
    print('не знаю')
```

- в условиях могут использоваться операторы сравнения: **<** (меньше), **<=** (меньше или равно), **>** (больше), **>=** (больше или равно), **!=** (не равно), **==** (равно); логические операторы: **or** (ИЛИ), **and** (И), **not** (НЕ); проверка принадлежности: **in** (принадлежит), **not in** (не принадлежит); проверка идентичности: **is** (идентично), **is not** (не идентично);
- цепочка сравнений вида **a < b < c ... < y < z** фактически равносильна: **(a < b) and (b < c) and ... and (y < z)**.

Задания 22-33:

22. Напишите программу, которая вычисляет частное от деления двух чисел. Программа проверяет правильность введенных пользователем данных и, если они неверные (делитель равен нулю), выдаёт сообщение об ошибке. Фрагмент экрана:

```
Вычисление частного
Введите делимое: 15
Введите делитель: 0
Ошибка! Делитель не может равняться нулю.
```

решение тут

23. Напишите программу вычисления площади кольца. Программа должна проверять правильность вводимых данных. В случае ошибки должно выводиться сообщение:

```
Ошибка! Радиус отверстия не может быть больше радиуса кольца.
```

решение тут

24. Напишите программу вычисления сопротивления электрической цепи, состоящей из двух сопротивлений. Сопротивления могут быть соединены последовательно или параллельно:

```
Вычисление сопротивления электрической цепи
Введите исходные данные:
Величина первого сопротивления (Ом): 15
Величина второго сопротивления (Ом): 27.3
Тип соединения (1 - последовательное, 2 - параллельное): 2
Сопротивление цепи: 9.68 Ом.
```

решение тут

25. Напишите программу вычисления стоимости покупки с учётом скидки. Скидка в 10% предоставляется, если сумма покупки больше 1000 руб. Фрагмент экрана:

```
Вычисление стоимости покупки с учетом скидки
Введите сумму покупки: 1200
Вам предоставляется скидка 10%
Сумма покупки с учетом скидки: 1080.00 руб.
```

решение тут

26. Напишите программу вычисления стоимости покупки с учётом скидки. Скидка в 3% предоставляется в случае, если сумма покупки больше 500 руб., в 5% – если сумма больше 1000 руб. Фрагмент экрана:

```
Вычисление стоимости покупки с учетом скидки
Введите сумму покупки: 1100
Вам предоставляется скидка 5%.
Сумма покупки с учетом скидки: 1045.00 руб.
```

решение тут

27. Напишите программу, которая запрашивает ввод номера месяца и выводит соответствующее название времени года. При вводе некорректных данных, программа должна вывести сообщение «Должно быть число от 1 до 12». Фрагмент экрана:

```
Введите номер месяца (число от 1 до 12): 11
11-й месяц – это зима.
```

решение тут

28. Напишите программу, которая запрашивает у пользователя номер дня недели и выводит одно из сообщений: «Рабочий день», «Суббота» или «Воскресенье». Фрагмент экрана:

```
Введите номер дня недели (число от 1 до 7): 3
Рабочий день.
```

решение тут

29. Напишите программу, которая «называет» числа от 1 до 10. Фрагмент экрана:

```
Введите число от 1 до 10: 9
девять
```

решение тут

30. Напишите программу, которая после введенного с клавиатуры числа (в диапазоне от 1 до 20), обозначающего денежную единицу, дописывает слово «рубль» в правильной форме. Например, «12 рублей», «3 рубля» и т.д. Фрагмент экрана:

```
Введите количество рублей (от 1 до 20): 12
12 рублей
```

решение тут

31. Напишите программу, которая после введенного с клавиатуры числа (в диапазоне от 0 до 99), дописывает слово «копейка» в правильной форме. Например, «99 копеек», «33 копейки», «61 копейка», ... Фрагмент экрана:

```
Введите количество копеек (от 0 до 99): 83
83 копейки
```

решение тут

32. Напишите программу, вычисляющую стоимость междугороднего телефонного разговора. Исходными данными для программы являются код города и продолжительность разговора. Цена одной минуты зависит от расстояния до города, в котором находится абонент:

Город	Код	Цена минуты, руб.
Владивосток	423	10,0
Москва	095	5,0
Мурманск	815	8,0
Ростов	863	2,5
Самара	846	5,5

Фрагмент экрана:

```
Вычисление стоимости разговора по телефону
Введите исходные данные: Код города: 423
Длительность (целое количество минут): 3
Город: Владивосток. Цена минуты: 10.0 руб.
Стоимость разговора: 30.0 руб.
```

решение тут

33. Напишите программу, которая вычисляет и сравнивает две суммы: шестнадцатеричного и двоичного, десятичного и восьмеричного чисел. Числа вводятся с клавиатуры. Фрагмент экрана:

```
Введите двоичное число: 111
Введите шестнадцатеричное число: A
Введите восьмеричное число: 6
Введите десятичное число: 15
1-я сумма меньше 2-й на 4
```

решение тут

Строки

- **строки** в языке Python – тип данных, предназначенный для обработки текстовой информации. Строки могут быть заключены в одинарные (`''`) или двойные (`""`) кавычки. Длинные строки могут быть разбиты на несколько строк с помощью обратной косой черты (`'\'`), или заключены в группы из трёх одинарных или двойных кавычек;
- в Python имеются два типа строк: **обычные** и **unicode-строки** (в Python 3 по умолчанию используются unicode-строки). Чтобы литералы воспринимались интерпретатором правильно, необходимо указать кодировку в начале программы. Например:

```
# -*- coding: utf-8 -*-
```

- строки Unicode создаются просто, например: `u'Hello World!'`. Управляющие последовательности позволяют включать в строку специальные символы (например: `'\u00a7'` – это символ `'§'`).

Строковые операции, функции и срезы

- над строками возможны операции: "`+`" («склеивание» – конкатенация строк), "`*n`" (повторение строки). Пример записи: `"A" + "B"`, `"A"*5`

- для работы со строками используются функции: **len(s)** – длина строки, **ord(s)** – код символа **s**, **chr(n)** – символ с кодом **n**, **bin(n)**, **oct(n)**, **hex(n)** – двоичное, восьмеричное или шестнадцатеричное представление целого числа **n**.
- строка – последовательность символов, где можно получить любой символ по его индексу (первый символ имеет индекс **0**). Подстрока может быть определена с помощью среза – двух индексов, разделенных двоеточием. Опущенный индекс считается равным **0**. Отрицательные значения индексов используются для отсчета с конца (последний символ имеет индекс **-1**).

  Примеры:

  word = 'Python'

  word[:3] – первые 3 символа (**'Pyt'**)

  word[2:] – строка, кроме первых 2-х символов (**'thon'**)

  print(word[:-2]) – строка, кроме последних 2-х символов (**'Pyth'**)

  print(word[-3:-1]) – из последних 3-х 2 символа, кроме последнего (**'ho'**)

Методы строк

- **s.count(sub[, start[, end]])** – количество вхождений подстроки **sub** в **s[start:end]**;
- **s.find(sub[, start[, end]])**, **s.rfind(sub[, start[, end]])** – наименьший или наибольший индекс в строке **s** начала вхождения подстроки **sub** в **s[start:end]**. Если подстрока не найдена, возвращает **-1**;
- **s.endswith(suffix[, start[, end]])** – возвращает **1**, если строка **s[start:end]** заканчивается на **suffix**, иначе возвращает **0**;
- **s.lstrip()**, **s.rstrip()**, **s.strip()** – строка, с удаленными в начале, в конце или в начале и конце строки пробелами;
- **s.replace(old, new[, maxcount])** – строка, в которой все вхождения подстроки **old** заменены на **new**;
- **s.lower()**, **s.upper()** – строка, все символы которой приведены к нижнему или к верхнему регистру;
- **s.isdecimal()** – возвращает **1**, если строка не содержит ничего, кроме десятичных цифр, иначе возвращает **0**.

Операция форматирования

- в операции форматирования левый операнд строки является строкой формата. Описание формата в порядке следования спецификаторов:
  '%[ключ][флаг][*мин_ширина][.точность][длина_типа]спецификатор_типа'
- ключ (опционально), определяет, какой аргумент значения будет подставляться;
- флаги преобразования: **"#"** – для форматов **'o'**, **'x'** и **'X'** результат будет начинаться с **'0'** (**'0o'**, **'0x'**, **'0X'**), **"0"** – результат заполняется нулями слева, **"-"** – результат выравнивается влево, **"+"** – перед числом всегда ставится знак;
- литералы для спецификаторов: **мин_ширина** – «**0**» ... «**9**», «*», **точность** (для чисел с плавающей запятой) – «**1**» ... «**9**», «*»;
- спецификаторы типа: **'d'**, **'i'**, **'u'** – десятичное число, **'o'** – число в 8-ричной системе счисления, **'x'**, **'X'** – число в 16-ричной системе счисления (буквы в нижнем или верхнем регистре), **'f'**, **'F'** – число с плавающей точкой (обычный формат), **'e'**, **'E'**, **'g'**, **'G'** – число с плавающей точкой с экспонентой, **'c'** – символ, **'s'**, **'r'**, **'a'** – строка;
- для отображения, в строке формата могут использоваться буквы и другие символы, **'%%'** – отображает **'%'**.

Примеры форматирования строк:

Запись строки	Вывод	Запись строки	Вывод
`"%+08i" % 15`	`'+0000015'`	`'%-9.3f' % 12.1236`	`'12.124  '`
`'%#o' % 22`	`'0o26'`	`"%#X" % 28`	`'0X1C'`
`'%6i%%' % 15`	`'    15%'`	`'%e' % 0.000000024`	`'2.400000e-08'`
`'%7s!' % 'Ура'`	`'   Ура!'`	`'%rABC' % 'pascal'`	`"'pascal'ABC"`

Задания 34-48:

34. Напишите программу, которая используя строку **'двадцать'** с помощью строковых операторов и срезов получает и выводит строку **'двенадцать'**.

    решение тут

35. Напишите программу, которая используя строку **'тринадцатое октября'** с помощью строковых операторов и срезов получает и выводит строку **'тридцатое ноября'**.

    решение тут

36. Напишите программу, которая используя только строковые операторы и срезы слова **'алгол'**, получает и выводит строки **'аллоалло'** (2 слова **'алло'**) и **'гоооооооооооол'** (12 букв **'о'**).

    решение тут

37. Напишите программу, которая используя только срезы слова **'бардак'** и операторы конкатенации, получает и выводит строку **'абракадабра'**.

    решение тут

38. Напишите программу, которая используя только срезы слова **'стоматолог'** и операторы конкатенации, получает и выводит через пробел слова **'сто'**, **'том'**, **'томат'**, **'сом'**, **'тол'**, **'соло'**, **'гол'**, **'лого'**, **'мост'**, **'молот'**, **'голос'**.

    решение тут

Основы программирования на языке Python

39. Напишите программу, которая сравнивает две введённые с клавиатуры строки, определяет и выводит какая из строк длиннее и на сколько. Фрагмент экрана:

```
Введите 1-ю строку: Монитор
Введите 2-ю строку: Клавиатура
2-я строка длиннее 1-й на 3 символ(а,ов)
```

решение тут

40. Напишите программу, которая выводит символ по его коду, введённому с клавиатуры. Фрагмент экрана:

```
Введите код символа (целое число от 1040 до 2003): 1063
Символ с (Unicode-)кодом 1063 - Ч
```

решение тут

41. Напишите программу, которая определяет и выводит код введённого с клавиатуры символа (или 1-го символа строки). Фрагмент экрана:

```
Введите символ (при вводе строки берётся 1-й символ): Ёжик
Код символа "Ё" строки "Ёжик" - 1025
```

решение тут

42. Напишите программу, которая десятичное (целое) число представляет в виде двоичного, восьмеричного и шестнадцатеричного чисел. Десятичное число вводятся с клавиатуры. Фрагмент экрана:

```
Введите десятичное число: 45
Число 45 в двоичном виде: 101101
Число 45 в восьмеричном виде: 55
Число 45 в шестнадцатеричном виде: 2D
```

решение тут

43. Напишите программу, которая определяет, являются ли введённая с клавиатуры строка натуральным числом, и в том случае, если число превышает 9999, то выводит произведение последних четырёх, иначе – произведение всех цифр числа. Фрагмент экрана:

```
Введите строку символов: 426872758
Строка являются натуральным числом.
Произведение последних 4-х цифр числа: 560
```

решение тут

44. Напишите программу, которая обрабатывает введённую с клавиатуры строку. Строка преобразовывается следующим образом: если в конце строки «1», символы с 1 по 5 выводятся в нижнем, с 6 по 10 – в верхнем, с 11 по 15 – в нижнем регистре и т.д., если в конце введённой строки «2», символы с 1 по 5 – в верхнем, с 6 по 10 – в нижнем, с 11 по 15 – в верхнем регистре и т.д. Фрагмент экрана:

```
Введите строку с "1" или "2" в конце, длиной от 10 до 25 символов:
asDFGHjklУКЕнгпростСМИТQweRTОлДщыяч2
Строка слишком длинная, символы после 25-го будут отброшены
ASDFGhjklyКЕНГПростсМИТQW
```

решение тут

45. Напишите программу, которая заменяет во введённой с клавиатуры строке первое и последнее вхождения буквы «ы» на номер вхождения. Фрагмент экрана:

```
Введите строку, в которой содержится хотя бы 1 буква "ы":
операцыя "ы" и другие прыключения Шурыка
операц1я "ы" и другие прыключения Шур4ка
```

решение тут

46. Напишите программу, которая после ввода с клавиатуры года, месяца и числа используя операции форматирования выводит дату в формате «ДД.ММ.ГГГГг.». Фрагмент экрана:

```
Введите год - число от 1900 до 2020: 1961
Введите месяц - число от 1 до 12: 4
Введите день - число от 1 до 31: 12
Дата: 12.04.1961г.
```

решение тут

47. Напишите программу, которая целое число отображает без дробной части, а дробное – с указанной точностью и соответствующим знаком («+» или «-»). Число (целое или дробное) вводятся с клавиатуры. Фрагмент экрана:

```
Введите число (целое или дробное): 12
+12
```

или

```
Введите число (целое или дробное): 358.126541
Введите точность (количество знаков после разделительной "."),
число от 1 до 6: 3
+358.127
```

решение тут

48. Напишите программу, которая суммирует два числа в шестнадцатеричной, восьмеричной и двоичной системах счисления. Фрагмент экрана:

```
Введите (десятичное) число 1: 28
Введите (десятичное) число 2: 11
Сумма чисел 28 и 11 в 16-чной системе счисления:
1C + B = 27
Сумма чисел 28 и 11 в 8-чной системе счисления:
34 + 13 = 47
Сумма чисел 28 и 11 в 2-чной системе счисления:
11100 + 1011 = 100111
```

решение тут

Циклы For

- Цикл **for** выполняет тело цикла для каждого элемента последовательности. Для прерывания выполнение цикла можно использовать **break**. Примеры записи:

```
for i in range(1, 10):
    a = input()
    s = s + a
print(s)
```

с инструкцией **break**:

```
for s in '120532':
    if not ('0' <= s < '5'):
        print('Строка не является 5-чным числом')
        break
```

Задания 49-68:

49. Напишите программу, которая 10 раз выводит на экран строку **'Циклы for на языке Python'**. Фрагмент экрана:

```
Циклы for на языке Python
Циклы for на языке Python
...
Циклы for на языке Python
```

решение тут

50. Напишите программу, которая выводит последовательность целых чисел от **3** до **15** через пробел. Фрагмент экрана:

```
3 4 5 6 7 8 9 10 11 12 13 14 15
```

решение тут

51. Напишите программу, которая выводит через пробел: в 1-й строке ряд чётных, а во 2-й – ряд нечётных чисел от **0** до **20**. Фрагмент экрана:

```
0 2 4 6 8 10 12 14 16 18 20
1 3 5 7 9 11 13 15 17 19
```

решение тут

52. Напишите программу, которая выводит через пробел: в 1-й строке каждое третье число из ряда от **1** до **25**, а во 2-й – каждое четвёртое число из ряда от **-1** до **-25**. Фрагмент экрана:

```
1 4 7 10 13 16 19 22 25
-1 -5 -9 -13 -17 -21 -25
```

решение тут

53. Напишите программу, которая выводит таблицу квадратов первых десяти целых положительных чисел. Фрагмент экрана:

```
Таблица квадратов
-----------------
Число   Квадрат
-----------------
  1         1
  2         4
...
 10       100
-----------------
```

решение тут

54. Напишите программу, которая выводит таблицу квадратов первых пяти нечётных чисел. Фрагмент экрана:


```
Таблица квадратов нечетных чисел
-----------------
Число   Квадрат
-----------------
  1         1
  3         9
...
  9        81
-----------------
```

решение тут

55. Напишите программу, которая выводит таблицу степеней двойки (от нулевой до десятой):

```
Таблица степеней двойки
   0         1
   1         2
   2         4
...
   9       512
  10      1024
```

решение тут

56. Напишите программу, которая вычисляет сумму первых **n** положительных целых чисел. Количество суммируемых чисел вводится во время работы программы. Фрагмент экрана:

```
Вычисление суммы положительных целых чисел
Введите количество суммируемых чисел: 18
Сумма первых 18 положительных целых чисел равна 171
```

решение тут

57. Напишите программу, для вычисления суммы первых **n** положительных нечётных чисел. Фрагмент экрана:

```
Вычисление суммы положительных нечетных чисел
Введите количество суммируемых чисел: 15
Сумма первых 15 нечетных чисел равна 225
```

решение тут

58. Напишите программу, вычисления суммы **n** (вводится с клавиатуры) членов ряда **1, 1/2, 1/3, 1/4, ...** . Фрагмент экрана:

```
Вычисление суммы членов ряда 1, 1/2, 1/3, 1/4, ...
Введите количество суммируемых чисел: 5
Сумма первых 5 членов ряда 1, 1/2, 1/3, 1/4, ... 2.283333333333333
```

решение тут

59. Напишите программу, которая выводит через пробел 16 членов ряда чисел Фибоначчи, начиная с 1-го (значения членов ряда Фибоначчи – **0**, **1**, **1**, **2**, **3**, **5**, **8**, **13**, **21**, ..., т.е.: **n(0) = 0**, **n(1) = 1**, **n(i) = n(i - 2) + n(i - 1)**, ...). Фрагмент экрана:

```
Члены ряда чисел Фибоначчи с 1-го по 16-й:
1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```

решение тут

60. Напишите программу, которая вычисляет факториал числа **n**, введенного с клавиатуры (факториал – произведение целых чисел от **1** до **n**). Фрагмент экрана:

```
Вычисление факториала числа
Введите число: 8
Факториал числа 8 равен 40320
```

решение тут

61. Напишите программу, которая выводит таблицу значений функции $y = -2{,}4x^2 + 5x - 3$ в диапазоне от **–2** до **+2** с шагом **0,5**. Фрагмент экрана:

```
Таблица значений функции y = -2,4 * x^2 + 5 * x - 3
------------------------------------------------------------
  -2.00  -1.50  -1.00  -0.50  +0.00  +0.50  +1.00  +1.50  +2.00
 -22.60 -15.90 -10.40  -6.10  -3.00  -1.10  -0.40  -0.90  -2.60
------------------------------------------------------------
```

решение тут

62. Напишите программу, которая проверяет, является ли введенная с клавиатуры строка двоичным числом. Фрагмент экрана:

```
Введите число: 1011201211220
Строка не является 2-чным числом
```

решение тут

63. Напишите программу, которая проверяет, является ли введенная с клавиатуры строка шестнадцатеричным числом. Фрагмент экрана:

```
Введите число: 10983ACF5
Строка является 16-чным числом
```

или:

```
Введите число: 10983acf5
Строка является 16-чным числом
```

решение тут

64. Напишите программу, для вычисления среднего арифметического введённых с клавиатуры дробных чисел. Количество чисел вводится с клавиатуры. Фрагмент экрана:

```
Введите количество чисел: 4
Введите очередное число: 11.7
Введите очередное число: 15
Введите очередное число: 7.8
Введите очередное число: 2.1
Введено чисел: 4 Сумма: 36.6
Среднее арифметическое: 9.15
```

решение тут

65. Напишите программу, которая вычисляет среднее арифметическое последовательности дробных чисел (количество чисел и числа вводятся с клавиатуры).

Программа также должна вывести максимальное и минимальное числа последовательности. Фрагмент экрана:

```
Введите количество чисел: 3
Введите первое число: 15.12
Введите очередное число: 5.111
Введите очередное число: 31.6
Введено чисел: 3 Среднее арифметическое: 12.237
Максимальное число: 31.6 Минимальное число: 5.111
```

решение тут

66. Напишите программу, которая определяет, является ли введённая с клавиатуры строка палиндромом (перевёртышем) типа «**казак**», «**ABBA**». Фрагмент экрана:

```
Введите строку: шалаш
Введенная строка "шалаш" является палиндромом
```

решение тут

67. Напишите программу проверки знания таблицы умножения чисел от **1** до **10** на случайное число **m** (от **2** до **9**). Программа должна вывести 10 примеров и выставить оценку: за 10 и 9 правильных ответов – **"отлично"**, за 8 и 7 – **"хорошо"**, за 6 и 5 – **"удовлетворительно"**, за 4 и менее – **"плохо"**. Фрагмент экрана:

```
Проверка знания таблицы умножения на 7
Введите ответы на нижеприведенные примеры:
    1 x 7 = 7
    2 x 7 = 14
    3 x 7 = 22
...
У Вас 8 правильных ответов. Ваша оценка - "хорошо"
```

решение тут

68. Напишите программу проверки умения складывать и вычитать числа в пределах **100**. Программа должна вывести 10 случайных примеров, причём в примерах уменьшаемое не должно быть меньше вычитаемого, т.е. не допускается предлагать испытуемому примеры с отрицательным результатом. Фрагмент экрана:

```
Проверка умения складывать и вычитать числа
Введите ответы на нижеприведенные примеры:
   14 + 74 = 88
    3 + 93 = 96
   91 -  5 = 86
...
   90 - 79 = 21
У Вас 7 правильных ответов. Ваша оценка - "хорошо"
```

решение тут

Циклы While

- цикл **while** – это инструкции, выполняющие одну и ту же последовательность действий, пока заданное после «**while**» условие истинно. Пример записи цикла:

```
while i < 15:
```

```
print(i)
i = i + 2
```

Задания 69-78:

69. Напишите программу, вычисляющую сумму и среднее арифметическое последовательности положительных чисел, вводимых с клавиатуры (длина последовательности не известна, окончанием ввода является число **0**). Фрагмент экрана:

```
Вычисление суммы и среднего арифметического
последовательности положительных чисел
Числа вводите после стрелки
Для завершения ввода введите ноль
-> 45
-> 23.1
-> 15.2
-> 0
Введено чисел: 3
Сумма чисел: 83.3
Среднее арифметическое: 27.77
```

решение тут

70. Напишите программу, которая определяет минимальное число из введенной с клавиатуры последовательности положительных чисел (длина последовательности не известна, окончанием ввода является число **0**). Фрагмент экрана:

```
Определение минимального числа
последовательности положительных чисел
Числа вводите после стрелки
Для завершения ввода введите ноль
-> 35.4
-> 80
-> 24.8
-> 0
Минимальное число последовательности: 24.8
```

решение тут

71. Напишите программу для определения минимального количества (**n**) натуральных чисел, сумма которых не менее числа **m** (**m** вводится с клавиатуры). Фрагмент экрана:

```
Определение минимального количества натуральных чисел,
сумма которых не менее (введите число): 100
Сумма первых 14 натуральных чисел не менее 100 и составляет 105
```

решение тут

72. Напишите программу для определения минимального количества (**n**) нечётных чисел, сумма которых не менее числа **m** (**m** вводится с клавиатуры). Фрагмент экрана:


```
Определение минимального количества нечетных чисел,
сумма которых не менее (введите число): 101
Сумма первых 11 нечетных чисел не менее 101 и составляет 121
```

решение тут

73. Напишите программу для определения минимального количества (**n**) членов ряда **1**, **1/2**, **1/3**, **1/4**, **...**, сумма которых составляет не менее **5**. Фрагмент экрана:

```
Определение минимального количества членов ряда
1, 1/2, 1/3, ..., сумма которых не менее 5
Сумма первых 83 членов ряда 1, 1/2, 1/3, ...
не менее 5 и составляет 5.002
```

решение тут

74. Напишите программу, которая определяет сумму положительных и сумму отрицательных чисел в последовательности (длина последовательности не известна, окончанием ввода является число **0**). Фрагмент экрана:

```
Введите целое число (0 - завершает ввод): 5
Введите очередное число: 3
Введите очередное число: -7
Введите очередное число: 0
Сумма положительных чисел: +8
Сумма отрицательных чисел: -7
```

решение тут

75. Напишите программу, которая "задумывает" число в диапазоне от 1 до 10 и предлагает пользователю угадать число за 5 попыток. Фрагмент экрана:

```
Компьютер "задумал" число от 1 до 10
Угадайте его за 5 попыток
Введите число: 7
Введите число: 4
Введите число: 6
Поздравляем! Вы угадали число за 3 попытки!
```

решение тут

76. Напишите программу, которая проверяет, является ли целое число, введенное пользователем, простым. Фрагмент экрана:

```
Введите целое число: 49
49 - не простое число
```

решение тут

77. Напишите программу, которая преобразует введенное пользователем десятичное число в число в указанной системе счисления (от 3-х до 9-и). Фрагмент экрана:

```
Введите целое десятичное число: 23
Введите основание системы счисления: 4
Десятичное число 23 в системе с основанием 4 - 113
```

решение тут ещё решение

78. Напишите программу, которая вычисляет наибольший общий делитель двух целых чисел. Фрагмент экрана:

```
Вычисление наибольшего общего делителя для двух целых чисел
Введите 1-е число: 20
Введите 2-е число: 45
Наибольший общий делитель чисел 20 и 45 равен 5
```

решение тут ещё решение

Массивы

- *в Python отсутствуют массивы в традиционном понимании этого термина, вместо них используются списки. Будем считать, что* **массивы** – это ***совокупность однотипных данных***, которые идентифицируются ***именем*** и ***индексами*** массива (начальный индекс в Python равен **0**). В зависимости от количества индексов массивы бывают ***одномерные***, ***двухмерные***, и т.д.
- пустой одномерный массив можно получить инструкцией: **`m1 = []`**, а заполнять операциями: **`m1 = m1 + [x]`** или **`m1 += [x]`** или методом **`m1.append(x)`**.
- получение трёхмерного массива и заполнение его элементами со значениями **a**:
  `m = [[[a for k in range(z)]for j in range(y)]for i in range(x)]`
- получение и заполнение с клавиатуры двумерного массива данными целого типа:

```
m =[[int(input('m['+str(i)+','+str(j)+']: '))for j in range(c)]for i in range(r)]
```

- функция **`sum(m)`** позволяет получить сумму элементов массива **m**.
- при работе с массивами можно использовать срезы (**`m[нач:кон:шаг]`**).

Методы массивов

`m.append(x)` – добавление элемента **x** в конец массива **m**;
`m1.extend(m2)` – добавление элементов массива **m2** в конец массива **m1**;
`m.insert(i, x)` – вставка элемента **x** в массив **m** перед элементом с индексом **i**.
`m.pop(i)` – удаление из массива **m** элемента с индексом **i** (отрицательные индексы позволяют вести отсчет с конца. По умолчанию **i** равен **-1**, т.е. соответствует последнему элементу);
`m.count(x)` – возвращение числа элементов массива **m** равных **x**;
`m.index(x)` – возвращение индекса первого элемента массива **m** равного **x**;
`m.remove(x)` – удаление из массива **m** первого элемента равного **x**;
`m.reverse()` – изменение порядка следования элементов массива **m** на обратный.

Задания 79-100:

79. Напишите программу, выводящую максимальный элемент одномерного массива из **5** положительных чисел. Числа вводятся с клавиатуры. Фрагмент экрана:

```
Определение максимального элемента массива из 5 чисел
Введите 0-й элемент массива: 1.8
Введите 1-й элемент массива: 7.2
...
Максимальный элемент массива 7.2
```

решение тут

80. Напишите программу, которая выводит минимальный элемент одномерного массива из **5** положительных чисел. Числа вводятся с клавиатуры. Фрагмент экрана:

```
Определение минимального элемента массива из 5 чисел
Введите 0-й элемент массива: 17.406
Введите 1-й элемент массива: 2.6
Введите 2-й элемент массива: 4.52
...
Минимальный элемент массива 2.6
```

решение тут

81. Напишите программу, которая сортирует одномерный массив по возрастанию методом пузырька. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Сортировка массива методом пузырька
Введите количество элементов массива: 7
Введите 0-й элемент массива: 8.9
Введите 1-й элемент массива: 2.5
Введите 2-й элемент массива: 9
...
Введите 5-й элемент массива: 1.2
Введите 6-й элемент массива: 7.4
Отсортированный массив: 1.2 2.5 5.2 6.0 7.4 8.9 9.0
```

решение тут

82. Напишите программу, которая в одномерном массиве меняет местами элементы, стоящие на чётных и нечётных местах (количество элементов – чётное). Данные вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива (четное число): 4
Введите 0-й элемент массива: 8.7
Введите 1-й элемент массива: 2.1
Введите 2-й элемент массива: 5.2
Введите 3-й элемент массива: 3.3
Массив после обработки: 2.1 8.7 3.3 5.2
```

решение тут

83. Напишите программу, которая перемещает элементы массива по кругу влево, т.е. **m[1] → m[0]**; **m[2] → m[1]**; ..., **m[0] → m[n]**. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 4
Введите 0-й элемент массива: 1.1
Введите 1-й элемент массива: 5.4
Введите 2-й элемент массива: 8
Введите 3-й элемент массива: 4.7
Массив после обработки: 5.4 8.0 4.7 1.1
```

решение тут

84. Напишите программу, которая меняет порядок следования элементов массива на обратный. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 5
Введите 0-й элемент массива: 8.4
Введите 1-й элемент массива: 5.3
Введите 2-й элемент массива: 1.8
Введите 3-й элемент массива: 9.9
Введите 4-й элемент массива: 4.7
Массив с обратным порядком следования элементов:
4.7 9.9 1.8 5.3 8.4
```

решение тут ещё решение

85. Напишите программу, которая в одномерном массиве меняет местами первую и вторую половины массива (количество элементов массива – чётное). Данные вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива (четное число): 6
Введите 0-й элемент массива: 6.4
Введите 1-й элемент массива: 1.9
Введите 2-й элемент массива: 8.3
Введите 3-й элемент массива: 5.6
Введите 4-й элемент массива: 4.2
Введите 5-й элемент массива: 8
Массив после обработки: 5.6 4.2 8.0 6.4 1.9 8.3
```

решение тут

86. Напишите программу, которая выводит начальный, средний и последний элементы одномерного массива (количество элементов массива – нечётное). Данные вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива (нечетное число): 7
Введите 0-й элемент массива: 1
Введите 1-й элемент массива: 2
Введите 2-й элемент массива: 3
...
Введите 6-й элемент массива: 7
Первый, средний и последний элементы массива: 1.0 4.0 7.0
```

решение тут другое решение ещё решение

87. Напишите программу, которая в одномерном массиве исключает начальный, средний и последний его элементы (количество элементов массива – нечётное). Данные вводятся с клавиатуры. Фрагмент экрана:


```
Введите количество элементов массива (нечетное число): 7
Введите 0-й элемент массива: 1
Введите 1-й элемент массива: 2
Введите 2-й элемент массива: 3
...
Введите 6-й элемент массива: 7
Массив без первого, среднего и последнего элементов:
[2.0, 3.0, 5.0, 6.0]
```

решение тут ещё решение

88. Напишите программу, которая выводит элементы одномерного массива с нечётными индексами. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 7
Введите 0-й элемент массива: 1
Введите 1-й элемент массива: 2
Введите 2-й элемент массива: 3
...
Введите 6-й элемент массива: 7
Массив без элементов с четными индексами: [2.0, 4.0, 6.0]
```

решение тут другое решение ещё одно решение ещё решение

89. Напишите программу, которая выводит элементы одномерного массива с нечётными индексами. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 7
Введите 0-й элемент массива: 1
Введите 1-й элемент массива: 2
Введите 2-й элемент массива: 3
...
Введите 5-й элемент массива: 6
Массив без элементов с нечетными индексами: [1.0, 3.0, 5.0]
```

решение тут другое решение ещё одно решение ещё решение

90. Напишите программу, которая в одномерном массиве исключает целочисленные его элементы. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 6
Введите 0-й элемент массива: 5.0
Введите 1-й элемент массива: 2.8
Введите 2-й элемент массива: 199.
Введите 3-й элемент массива: 256.25
Введите 4-й элемент массива: 100.001
Введите 5-й элемент массива: 196
Массив без целочисленных элементов: [2.8, 256.25, 100.001]
```

решение тут ещё решение

91. Напишите программу, которая в одномерном массиве исключает нецелочисленные его элементы. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 6
Введите 0-й элемент массива: 5.0
Введите 1-й элемент массива: 2.8
Введите 2-й элемент массива: 199.
Введите 3-й элемент массива: 256.25
Введите 4-й элемент массива: 100.001
Введите 5-й элемент массива: 196
Массив без нецелочисленных элементов: [5.0, 199.0, 196.0]
```

решение тут ещё решение

92. Напишите программу, которая в одномерном массиве исключает отрицательные его элементы. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 6
Введите 0-й элемент массива: 0.1
Введите 1-й элемент массива: -99
Введите 2-й элемент массива: +1.289
Введите 3-й элемент массива: 0.0
Введите 4-й элемент массива: 58.11
Введите 5-й элемент массива: -123
Массив без отрицательных элементов: [0.1, 1.289, 0.0, 58.11]
```

решение тут ещё решение

93. Напишите программу, которая в одномерный массив **m1** вставляет все элементы массива **m2**, начиная с позиции **k**. Количество элементов (**n1**, **n2**), элементы массивов и значение позиции **k** (**0 < k < n1**) вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива 1: 6
Введите 0-й элемент массива 1: 47.5
Введите 1-й элемент массива 1: 28.6
...
Введите количество элементов массива 2: 4
Введите 0-й элемент массива 2: 31
Введите 1-й элемент массива 2: 12.8
...
Введите k (значение позиции вставки): 2
Массив 1 с элементами массива 2: [47.5, 28.9, 31.0, ...]
```

решение тут ещё решение

94. Напишите программу для определения значения элемента одномерного массива, после добавления которого, сумма всех элементов массива равняется **0**. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:


```
Введите количество элементов массива: 4
Введите 0-й элемент массива: 15.2
Введите 1-й элемент массива: 12.3
Введите 2-й элемент массива: -16.5
Введите 3-й элемент массива: 5.1
После добавления в массив [15.2, 12.3, -16.5, 5.1]
числа -16.1, сумма всех элементов массива равняется 0.
```

решение тут ещё решение

95. Напишите программу, которая исключает из одномерного массива **q** элементов, начиная с позиции **k**. Количество элементов, элементы массива, значения **q** и **k** вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 5
Введите 0-й элемент массива: 19.4
Введите 1-й элемент массива: 14.3
...
Введите 4-й элемент массива: 12.8
Введите q (количество исключаемых элементов): 2
Введите k (позиция начала исключения элементов): 3
Массив после исключения элементов: [19.4, 14.3, 12.8]
```

решение тут ещё решение

96. Напишите программу, которая после каждого элемента одномерного массива вставляет новый элемент, равный его квадрату. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 3
Введите 0-й элемент массива: 16
Введите 1-й элемент массива: 5.5
Введите 2-й элемент массива: 2.5
Массив после добавления квадратов элементов:
[16.0, 256.0, 5.5, 30.25, 2.5, 6.25]
```

решение тут ещё решение

97. Напишите программу, которая определяет, образуют ли элементы одномерного массива убывающую последовательность. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 5
Введите 0-й элемент массива: 9
Введите 1-й элемент массива: 7
...
Элементы одномерного массива [9.0, 7.0, 6.0, 2.0, 2.0]
не образуют убывающую последовательность
```

решение тут ещё решение

98. Напишите программу, которая из одномерного массива с целочисленными элементами, строит два новых массива: **m1** – с нечётными, **m2** – с чётными элементами. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 7
Введите натуральное число (0-й элемент массива): 8
Введите натуральное число (1-й элемент массива): 3
...
Введите натуральное число (6-й элемент массива): 148
Массив с нечетными элементами: [3, 15, 27]
Массив с четными элементами: [8, 16, 22, 148]
```

решение тут ещё решение

99. Напишите программу, которая определяет, имеются ли в одномерном массиве два подряд идущих одинаковых числа. Количество элементов и элементы массива вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество элементов массива: 5
Введите натуральное число (0-й элемент массива): 5
Введите натуральное число (1-й элемент массива): 71
...
Массив [5, 71, 5, 16, 8]
не имеет два подряд идущих одинаковых числа
```

решение тут

100. Дан двумерный массив, в который записываются значения катетов **k** прямоугольных треугольников. Напишите программу, которая для каждого треугольника после значений катетов добавляет в массив элемент, равный его площади. Количество треугольников и размеры катетов вводятся с клавиатуры. Фрагмент экрана:

```
Введите количество треугольников: 4
Треугольник 1 - размер катета 1: 1.4
Треугольник 1 - размер катета 2: 2.5
Треугольник 2 - размер катета 1: 2.2
Треугольник 2 - размер катета 2: 1.7
...
Полученный массив [[катет1, катет2, площадь]...]:
[[1.4, 2.5, 1.75], [2.2, 1.7, 1.87], [3.6, 1.8, 3.24], [3.3, 1.6, 2.64]]
```

решение тут ещё решение

Основы программирования на языке Python