**Лабораторная работа №1**

**Тема:** Арифметические выражения, управляющие конструкции и массивы в Python

**Цель:** научиться создавать простейшие программы на Python, используя операторы выбора и циклов, арифметические выражения и массивы

**теоретическая часть**

**Python** - высокоуровневый язык программирования общего назначения, ориентированная на повышение производительности разработчика и чтения кода.

Python поддерживает несколько парадигм программирования, в том числе структурную, объектно-ориентированную, функциональную, императивную и аспектно-ориентированную.

В настоящее время существует две ветви Python - 2.х и 3.х. Мы будем работать преимущественно с веткой 3.х.

Существуют различные интерпретаторы для языка Python. Официальный интерпретатор можно скачать на сайте https://www.python.org

Также Вы можете использовать один из вариантов многочисленных сборников Python, например: Anaconda, Python (x, y), EnthoughtCanopy, WinPython тому подобное.

**Основные принципы синтаксиса языка Python:**

1. Конец строки является концом инструкции (точка с запятой не нужна).

х = 5

print (2 + х)

2. Вложенные инструкции объединяются в блоки по величине отступлений. Отступление может быть любым, главное, чтобы в пределах одного вложенного блока отступление был одинаков (рекомендуется делать отступление 4 пробелы).

if х == 10:

print ( 'yes')

3. Вложенные инструкции в Python записываются в соответствии с одного и того же шаблона, когда основная инструкция завершается двоеточием, вслед за чем располагается вложенный блок кода, обычно с отступлением под строкой основной инструкции.

Рассмотрим на примерах основы работы с Python.

**# Базовые типы данных (указываются неявно)**

a = 5 # int

b = 7.0 # float

c = 2> 4 # boolean

d = "World" # string

e = 1.5 + 0.5j # complex

print (type (a))

print (type (b))

print (type (c))

print (type (d))

print (type (e))

print (a, b, c, d, e.real, e.imag)

**# Преобразование типов данных**

int (True) # == 1

float (True) # == 1.0

str (True) # = 'True'

bool (0) # == False

bool (0.0) # == False

bool (1) # == True

bool (10) # == True

**В Python есть особый способ обмена переменных значениями:**

(a, b) = (b, a)

Он используется очень часто. Данный метод работает всегда, даже если переменные различных типов (в этом случае они обмениваются не только значениями, но и типами). Круглые скобки в этой записи можно опустить:

a, b = b, a

**# Основные арифметические операции**

a = 5 # int

b = 7.0 # float

c = 1 + 2 # 3

d = 5 - 3 # 2

e = a \* b # 35.0

f = 3.0 / 2 # 1.5

g = 3/2 # 1

h = 5% 3 # 2

j = 10 \*\* 7.3 # 19952623.1497

print (type (a))

print (type (b))

print (type (c))

print (type (d))

print (type (e))

print (type (f))

print (type (g))

print (type (h))

print (type (j))

print (a, b, c, d, e, f, g, h, j)

**# Встроенные математические функции (необходимо подключение библиотеки math)**

from math import \*

a = 1

b = 2

x = sqrt (a \* b) / (exp (a) \* b) + a \* exp ((2 \* a) / b)

print (x)

Таблица 1 - Функции в библиотеке math

|  |  |
| --- | --- |
| **Название функции** | **назначение функции** |
| math.ceil (x) | Возвращает округленное x как ближайшее целое значение типа float, равным или превышающим x (округление "вверх"). |
| math.copysign (x, y) | Возвращает число x со знаком числа y. На платформе, которая поддерживает знак нуля copysign (1.0, -0.0) даст -1.0. |
| math.fabs (x) | Возвращает абсолютное значение (модуль) числа x. В Python есть встроенная функция abs, но она возвращает модуль числа с тем же типом, что число, fabsже всегда возвращает значение типа float. |
| math.factorial (x) | Возвращает факториал целого числа x, если x не является целым возникает ошибка ValueError. |
| math.floor (x) | В противоположность ceil (x) возвращает округленное x как ближайшее целое значение типа float, меньше или равно x (округление "вниз"). |
| math.fmod (x, y) | Аналогична функции fmod (x, y) библиотеки C. Отметим, что это не то же самое, что выражение Python x% y.Желательно использовать при работе с объектами float, в то время как x% y больше подходит для int. |
| math.frexp (x) | Представляет число в экспоненциальном записи x = m \* 2e и возвращает мантиссу m (действительное число, модуль которого лежит в интервале от 0,5 до 1) и порядок e (целое число) как пару чисел (m, e). Если x = 0, то возвращает (0.0, 0) |
| math.fsum (iterable) | Возвращает float сумму от числовых элементов итеруемого объекта. |
| math.isinf (x) | Проверяет, есть ли float объект x плюс или минус бесконечностью, результат соответственно True или False. |
| math.isnan (x) | Проверяет, есть ли float объект x объектом NaN (not a number). |
| math.ldexp (x, i) | Возвращает x \* 2i, то есть совершает действие, обратное функции math.frexp (x). |
| math.modf (x) | Возвращает часть, идущая после запятой и целую часть от float числа. Оба результата сохраняют знак начального числа x и представлены типом float. |
| math.trunc (x) | Возвращает целую часть числа x в виде int объекта. |
| **Степенные и логарифмические функции** | |
| math.exp (x) | Возвращает exp. |
| math.log (x [, base]) | При передаче функции одного аргумента x, возвращает натуральный логарифм x. При передаче двух аргументов, второй берется как основа логарифма. |
| math.log1p (x) | Возвращает натуральный логарифм от x + 1. |
| math.log10 (x) | Возвращает десятичный логарифм x. |
| math.pow (x, y) | Возвращает x в степени y. |
| math.sqrt (x) | Квадратный корень (square root) с x. |
| **тригонометрические функции** | |
| math.acos (x) | Возвращает арккосинус x, в радианах. |
| math.asin (x) | Возвращает арксинус x, в радианах. |
| math.atan (x) | Возвращает арктангенс x, в радианах. |
| math.atan2 (y, x) | Возвращает atan (y / x), в радианах. Результат лежит в интервале [-π, π]. Вектор, конец, которого задается точкой (x, y) образует угол с положительным направлением оси x. Поэтому эта функция более общее назначение, чем предыдущая. Например и atan (1), и atan2 (1, 1) дадут в результате pi / 4, но atan2 (1, -1) это уже 3 \* pi / 4. |
| math.cos (x) | Возвращает косинус x, где x выражен в радианах. |
| math.hyp (x, y) | Возвращает евклидову норму, то есть sqrt (x \*\* 2 + y \*\* 2). Удобно для вычисления гипотенузы (hyp) и длины вектора. |
| math.sin (x) | Возвращает синус x, где x выражен в радианах. |
| math.tan (x) | Возвращает тангенс x, где x выражен в радианах. |
| **Радианы в градусы и наоборот** | |
| math.degrees (x) | Конвертирует значение угла x с радиан в градусы. |
| math.radians (x) | Конвертирует значение угла x из градусов в радианы. |

**# Ввод данных**

x = input ( 'Введите х \ n') # вводимые имеют тип строки

y = input ( 'Введите y \ n')

x = int (x) # осуществляем преобразование типов

y = int (y)

print (x + y) # добавляем два числа, которые были введены пользователем

**# Вывод данных по формату**

for i in range (10):

A = i \* 18

print ( "% 02i \ t% .1f"% (i, A))

Таблица 2 - Основные методы форматирования данных для вывода

|  |  |
| --- | --- |
| % s | строка |
| % d | целое число |
| % f | десятичное представление с шестью знаками после запятой |
| % e | "научное" представление |
| % g | компактное представление десятичного числа |
| % xz | вывода в формате z в поле ширины x, выравнивание по правой стороне |
| % -xz | вывода в формате z в поле ширины x, выравнивание по левой стороне |
| % .yz | вывода в формате z с y знаками после запятой |
| % x.yz | вывода в формате z с y знаками после запятой в поле ширины x |
| %% | вывода знака процента |
| \ n | переход на новую строку |

**# Оператор условия if**

for i in range (-50,51,10):

if i <0:

print ( "% i - холодно"% (i))

elif i> 0:

print ( "% i - тепло"% (i))

else:

print ( "% i - нормально"% (i))

**# Цикл for**

for i in range (10):

print (i)

A = []

for i in range (10):

A.append (i \*\* i)

B = zeros (10 dtype = int)

for i in range (10):

B [i] = i \*\* i

print (A)

print (B)

**# Цикл while**

i = 0

while i <10:

print (i)

i = i + 1

i = 0

A = []

while i <10:

A.append (i \*\* i)

i = i + 1

print (A)

В Python вместо массивов используются списки, словари, кортежи. Многомерный массив реализуется с помощью списка списков (вложенного списка). При необходимости использовать именно массивы, например., При обработке большого объема данных, следует применять библиотеку NumPy, которая позволяет с ними работать.

**# Списки.** **пример 1**

a = [1, 2, 3]

b = [4, 5]

c = a + b

d = b \* 3

print (a) # Будет выведено [1, 2, 3]

print (b) # будет выведено [4, 5]

print (d) # будет выведено [4, 5, 4, 5, 4, 5]

print ([7, 8] + [9]) # будет выведено [7, 8, 9]

print ([0, 1] \* 3) # будет выведено [0, 1, 0, 1, 0, 1]

a2 = [1, 2, 3, 4, 5]

for i in range (len (a2)):

print (a2 [i])

for elem in a:

print (elem, end = '')

**# Списки.** **пример 2**

A = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

print (A [0: 9: 2]) # будет выведено [0, 2, 4, 6, 8]

print (A [0: 9: 3]) # будет выведено [0, 3, 6]

print (A [2: 4]) # будет выведено [2, 3]

print (A [-8: -6]) # будет выведено [2, 3]

print (A [-3: -1]) # будет выведено [7, 8]

b = list ( 'список')

print (b) # будет выведено [ 'с', 'п', 'и', 'с', 'в', 'к']

**# Списки.** **пример 3**

clouds = [ "St", "Sc", "Ns", "As", "Ac"]

print (type (clouds)) # будет выведено <class 'list'>

print (clouds) # будет выведено [ 'St', 'Sc', 'Ns', 'As "," Ac']

clouds.append ( "Cu")

print (clouds) # будет выведено [ 'St', 'Sc', 'Ns', 'As "," Ac "," Cu']

clouds.append ( "Cb")

print (clouds) # будет выведено [ 'St', 'Sc', 'Ns', 'As "," Ac "," Cu "," Cb']

clouds.sort ()

print (clouds) # будет выведено [ 'Ac', 'As', 'Cb', 'Cu "," Ns "," Sc "," St']

print (clouds [2]) # будет выведено Cb

print (clouds [2: 5]) # будет выведено [ 'Cb', 'Cu "," Ns']

print (clouds [2:]) # будет выведено [ 'Cb', 'Cu "," Ns "," Sc "," St']

print (clouds [5]) # будет выведено [ 'Ac', 'As', 'Cb', 'Cu "," Ns']

print (clouds [2: 5: 2]) # будет выведено [ 'Cb', 'Ns']

print (clouds [2 :: 2]) # будет выведено [ 'Cb', 'Ns', 'St']

print (clouds [:: 2]) # будет выведено [ 'Ac', 'Cb', 'Ns', 'St']

**# Списки списков (можно использовать, например., Вместо двумерного массива)**

a = [[1, 2, 3], [4, 5, 6]]

print (a) # будет выведено [[1, 2, 3], [4, 5, 6]]

print (a [0]) # будет выведено [1, 2, 3]

print (a [1]) # будет выведено [4, 5, 6]

print (a [0] [0]) # будет выведено 1

print (a [0] [1]) # будет выведено 2

print (a [1] [0]) # будет выведено 4

**# Списки списков.** **Пример создания матрицы:**

**# 1 0 0 0**

**# 2 1 0 0**

**# 2 2 1 0**

**# 2 февраля 2 января**

n = 4

a = [[0] \* n for i in range (n)]

for i in range (n):

for j in range (n):

if i <j:

a [i] [j] = 0

elif i> j:

a [i] [j] = 2

else:

a [i] [j] = 1

for row in a:

print ( '' .join ([str (elem) for elem in row]))

**Срезы для списков**

**example\_list [i: j]** # выберет все элементы списка с i-го (включительно) по j-й (исключительно)

**example\_list [i:]** # выберет все элементы списка с i-го (включительно) до конца,

**example\_list [: j]** # выберет все элементы списка с января по j-й (исключительно).

**example\_list [i: j: k]** # выберет каждый k-й элемент списка с i-го (включительно) по j-й (исключительно)

**example\_list [i :: k]** # выберет каждый k-й элемент списка с i-го (включительно) до конца

**example\_list [: j: k]** # выберет каждый k-й элемент списка с января по j-й (исключительно),

**example\_list [:: k]** # выберет каждый k-й элемент списка.

#k также может быть отрицательным, что приведет к формированию нового списка с обратным порядком элементов. Проверьте сами: **example\_list [:: 2],** **example\_list [4: 3],** **example\_list [1 :: - 1],** **example\_list [3: 0: -1].**

**Операции со списками**

**x in A** - проверить, содержится ли элемент в списке. Возвращает True или False.

**x not in A** - то же самое, что not (x in A).

**min (A)** - наименьший элемент списка.

**max (A)** - наибольший элемент списка.

**методы списков**

|  |  |
| --- | --- |
| **метод** | **назначение** |
| **list.append** (x) | Добавляет элемент в конец списка |
| **list.extend** (L) | Расширяет список list, добавляя в конец все элементы списка L |
| **list.insert** (i, x) | Вставляет на i-ый элемент значение x |
| **list.remove** (x) | Удаляет первый элемент в списке, который имеет значение x. ValueError, если такого элемента не существует |
| **list.pop** ([i]) | Удаляет i-й элемент и возвращает его. Если индекс не указан, удаляется последний элемент |
| **list.index** (x)  **list.index** (x [start [, end]]) | Возвращает положение первого элемента со значением x. Если указанные параметры start и end - поиск ведется от start до end |
| **list.count** (x) | Возвращает количество элементов со значением x |
| **list.sort** ([key = функция]) | Сортирует список на основе функции |
| **list.reverse** () | Разворачивает список в обратном направлении |
| **list.copy** () | Поверхностная копия списка |
| **list.clear** () | очищает список |

**Задание:**

В задании (1) каждого варианта необходимо вычислить значение z и вывести его на экран.

В задании (2) каждого варианта для его реализации следует применить разветвления и циклы.

В задании (3) каждого варианта одномерные массивы следует реализовать с помощью списков, а матрицы - с помощью вложенных списков.

**Варианты заданий:**

**вариант 1**

1)

Число вводится пользователем в консоли Python.

2) Вывести n-ый элемент последовательности Фибоначчи (в последовательности Фибоначчи каждое последующее число равно сумме двух предыдущих: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377 . F0 = 1, F1 = 1, F2 = 1, Fn = Fn-1 + Fn-2, n> = 2)

3) Дано матрицу размером 5х4. Поменять местами первую строку и третью строчку.

**вариант 2**

1)

Число вводится пользователем в консоли Python.

2) Найти сумму всех чисел от 1 до n, кратных числу k.

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти максимальный элемент.
  + Вычислить среднее арифметическое отрицательных элементов массива.
  + вывести положительные элементы на экран в обратном порядке.

**вариант 3**

1)

Числа m и n вводятся пользователем в консоли Python.

2) Компьютер «загадал» число от 1 до 100 (с помощью функции random). Пользователь вводит с клавиатуры некоторое число и получает один из ответов: "Мое число больше", "Мое число меньше", "Вы угадали". Игра повторяется до тех пор пока число не угадано.

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* + Найти минимальный элемент.
  + Вычислить сумму положительных нечетных элементов массива.
  + вывести положительные элементы на экран.

**вариант 4**

1)

Число m вводится пользователем в консоли Python.

2) Найти произведение всех нечетных чисел от х до у

3) среднее арифметическое элементов каждой строки матрицы Q (l, m) и вычесть его из элементов этой строки.

**вариант 5**

1)

Числа х и у вводятся пользователем в консоли Python.

2) Найти наибольший общий делитель чисел x и y

3) Дано двумерный массив размерностью 4х6, заполненный целыми числами. Сформировать одномерный массив, каждый элемент которого равен наибольшему элементу соответствующей строки.

**вариант 6**

1)

Число вводится пользователем в консоли Python.

2) Дано натуральное число n, вычислить у = 1  3  5 ... (2  n-1).

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* + Найти максимальный положительный элемент.
  + Вычислить сумму додантих парных элементов массива.
  + вывести отрицательные элементы на экран в обратном порядке.

**вариант 7**

1)

Число m вводится пользователем в консоли Python.

2) Начав тренировки, спортсмен в первый день пробежал 10 км. Ежедневно он увеличивал дневную норму на 10% нормы предыдущего дня.Какой суммарный путь пробежит спортсмен за n дней?

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти минимальный положительный элемент.
  + Вычислить среднее арифметическое положительных элементов массива.
  + Вывести ненулевые элементы на экран в обратном порядке.

**вариант 8**

Числа х и у вводятся пользователем в консоли Python.

1. Вычислить факториал, используя цикл. число n вводит пользователь.
2. Дано двумерный массив размерностью 4х6, заполненный целыми числами. Сформировать одномерный массив, каждый элемент которого равен сумме элементов соответствующей строки.

**вариант 9**

1)

Число вводится пользователем в консоли Python.

2) *Вычислить, используя цикл, числа* x и n вводятся пользователем.

3) Определить, есть ли в двумерном массиве столбец, состоящий только из положительных или нулевых элементов.

**вариант 10**

1)

Число вводится пользователем в консоли Python.

2) Ежемесячная стипендия студента составляет A грн., А расходы на проживание превышают стипендию и составляют В грн. на месяц. Рост цен ежемесячно увеличивает расходы на 5%. Составьте программу расчета суммы денег, которую необходимо единовременно попросить у родителей, чтобы можно было прожить учебный год (10 месяцев), используя только эти деньги и стипендию.

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти минимальный положительный элемент.
  + Вычислить произведение нечетных элементов массива.
  + вывести отрицательные элементы на экран.

**вариант 11**

Число m вводится пользователем в консоли Python.

1. Определить, является ли n простым числом.

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти минимальный положительный элемент.
  + Вычислить сумму четных элементов массива.
  + Вывести массив на экран в обратном порядке.

**вариант 12**

1)

Число вводится пользователем в консоли Python.

2) Одноклеточная амеба делится каждые 3:00 на 2 клетки. Определить сколько амеб через n часов.

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти максимальный отрицательный элемент.
  + Вычислить среднее арифметическое нечетных элементов массива.
  + вывести отрицательные элементы на экран.

**вариант 13**

1)

Числа и вводится пользователем в консоли Python.

2) Определить, является ли число n избыточным числом. Избыточное число - положительное целое число n, сумма положительных делителей (отличных от n) которого превышает n. Число 48, например, является избыточным, поскольку 1 + 2 + 3 + 4 + 6 + 8 + 12 + 16 + 24 = 76, 76> 48.

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти минимальный отрицательный элемент.
  + Вычислить среднее арифметическое положительных элементов массива.
  + вывести положительные элементы на экран.

**вариант 14**

1)

Числа и вводится пользователем в консоли Python.

2) Спортсмен пробегает за 1-й день М км, каждый последующий день он увеличивает норму пробега на К%. Определите через сколько дней норма пробега может стать более 50 км.

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* + Найти максимальный отрицательный элемент.
  + Вычислить произведение отрицательных элементов массива.
  + Вывести ненулевые элементы на экран в обратном порядке.

**вариант 15**

1)

Числа и вводится пользователем в консоли Python.

2) Найти наименьшее общее кратное чисел x и y.

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* + Найти минимальный элемент.
  + Вычислить произведение ненулевых нечетных элементов массива.
  + Вывести массив на экран в обратном порядке.

**вариант 16**

1)

Число m вводится пользователем в консоли Python.

2) Дано натуральное число n, вычислить у = 2  4  6 ... (2  n).

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* + Найти максимальный отрицательный элемент.
  + Вычислить среднее арифметическое четных элементов массива.
  + Вывести ненулевые элементы на экран в обратном порядке.

**вариант 17**

1)

Числа a и b вводятся пользователем в консоли Python.

2) Дано натуральное число n, вычислить *×××.*

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти максимальный элемент.
  + Вычислить сумму четных элементов массива.
  + вывести отрицательные элементы на экран в обратном порядке.

**вариант 18**

1)

Числа и вводятся пользователем в консоли Python.

2) Найти сумму S квадратов чисел от 1 до N

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* + Найти максимальный положительный элемент.
  + Вычислить произведение элементов массива.
  + вывести положительные элементы на экран.

**вариант 19**

1)

Число вводится пользователем в консоли Python.

2) Найти сумму всех четных чисел от x до y.

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* + Найти минимальный положительный элемент.
  + Вычислить сумму положительных элементов массива, кратных 3.
  + Вывести НЕ нулевые элементы на экран.

**вариант 20**

1)

Числа m и n вводятся пользователем в консоли Python.

2) Определить, является ли число n совершенным. Совершенное число - натуральное число, равное сумме всех своих делителей, например., 6 (1 + 2 + 3 = 6), 28 (1 + 2 + 4 + 7 + 14 = 28).

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти минимальный элемент.
  + Вычислить произведение не нулевой элементов массива.
  + вывести положительные элементы на экран в обратном порядке.

**вариант 21**

1)

Число х вводится пользователем в консоли Python.

2) Найти первое число Фибоначчи, что будет больше заданного числа р (в последовательности Фибоначчи каждое последующее число равно сумме двух предыдущих: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144 , 233, 377. F0 = 1, F1 = 1, F2 = 1, Fn = Fn-1 + Fn-2, n> = 2).

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти максимальный элемент.
  + Вычислить среднее арифметическое отрицательных элементов массива.
  + Вывести массив на экран в обратном порядке.

**вариант 22**

1)

Число х вводится пользователем в консоли Python.

2) Определить, является ли число n недостаточным числом. Недостаточное число - натуральное число, сумма собственных делителей которого меньше именно число. Напр., 15 - недостаточное число, его делителями являются 1, 3 и 5, их сумма равна 9, что меньше 15.

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* + Найти максимальный положительный элемент .
  + Вычислить сумму элементов массива.
  + Вывести ненулевые элементы на экран в обратном порядке.

**вариант 23**

1)

Число х вводится пользователем в консоли Python.

2) Поменять порядок цифр числа n на возвратной.

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* + Найти максимальный отрицательный элемент .
  + Вычислить сумму отрицательных элементов массива .
  + Вывести положительные элементы на экран .

**вариант 24**

1)

Число х вводится пользователем в консоли Python.

2) Дано натуральное число N. Определить наибольшую цифру и ее позицию в числе (например., N = 573863, самой цифра 8, ее позиция - четвертая слева).

Для выполнения задания использовать операции целочисленного деления и нахождения остатка от деления.

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* + Найти максимальный элемент.
  + Вычислить среднее арифметическое нечетных элементов массива.
  + Вывести отрицательные элементы на экран .

**вариант 25**

1)

Число х вводится пользователем в консоли Python.

2) Дано целое число М. Нужно найти наименьшее целое отрицательное число k, при котором 3 k> M.

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти минимальный отрицательный элемент .
  + Вычислить произведение ненулевых элементов массива, кратных 3.
  + Вывести отрицательные элементы на экран в обратном порядке .

**вариант 26**

1)

Число х вводится пользователем в консоли Python.

2) Найти сумму всех чисел от х до у, кратных числу 3.

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти минимальный положительный элемент .
  + Вычислить произведение не нулевой элементов массива.
  + Вывести ненулевые элементы на экран в обратном порядке.

**вариант 27**

1)

Число х вводится пользователем в консоли Python.

2) Найти сумму цифр числа n.

3) Дано одномерный массив, состоящий из N действительных элементов.

* + Найти максимальный элемент.
  + Вычислить среднее арифметическое положительных элементов массива .
  + Вывести отрицательные элементы на экран в обратном порядке .

**вариант 28**

1)

Числа m и n вводятся пользователем в консоли Python.

2) в интервале от A до B найти сумму и количество всех целых чисел, делящихся на 3 без остатка и не делятся на 9 без остатка.

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* Найти минимальный элемент.
* Вычислить сумму элементов массива.
* Вывести положительные элементы на экран .

**вариант 29**

1)

Число х вводится пользователем в консоли Python.

2) Дано целое неотрицательное число N. Если N - нечетное, то вывести произведение 1 · 3 · ... · N; если N - четное, то вывести произведение 2 \* 4 \* ... · N.

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* Найти максимальный элемент массива.
* Вычислить среднее арифметическое элементов массива.
* Вывести массив на экран в обратном порядке.

**вариант 30**

1)

Число х вводится пользователем в консоли Python.

2) Дано натуральное число *n* . Получить все простые делители этого числа.

3) Дано одномерный массив, состоящий из N целочисленных элементов.

* + Найти минимальный отрицательный элемент .
  + Вычислить сумму отрицательных элементов массива .
  + Вывести положительные элементы на экран .