ЛАБОРАТОРНАЯ РАБОТА №6.

Работа со списками в Python.

Цели работы:

1. изучить особенности использования списков в Python;
2. создать приложение, демонстрирующее различные способы работы со списками в Python.

**Краткие теоретические сведения**

Одна из ключевых особенностей Python, благодаря которой он является таким популярным – это простота. Особенно подкупает простота работы с различными структурами данных – списками, кортежами, словарями и множествами. Сегодня мы рассмотрим **работу со списками**.

**Что такое список (list) в Python?**

Список (list) – это структура данных для хранения объектов различных типов. Если вы использовали другие языки программирования, то вам должно быть знакомо понятие массива. Так вот, список очень похож на массив, только, как было уже сказано выше, в нем можно хранить объекты различных типов. Размер списка не статичен, его можно изменять. Список по своей природе является изменяемым типом данных. Переменная, определяемая как список, содержит ссылку на структуру в памяти, которая в свою очередь хранит ссылки на какие-либо другие объекты или структуры.

**Как списки хранятся в памяти?**

Как уже было сказано выше, список является изменяемым типом данных. При его создании в памяти резервируется область, которую можно условно назвать некоторым “контейнером”, в котором хранятся ссылки на другие элементы данных в памяти. В отличии от таких типов данных как число или строка, содержимое “контейнера” списка можно менять.

**Создание, изменение, удаление списков и работа с его элементами**

Создать список можно одним из следующих способов.

>>> a = []

>>> type(a)

<**class** 'list'>

>>> b = list()

>>> type(b)

<**class** 'list'>

Также можно создать список с заранее заданным набором данных.

>>> a = [1, 2, 3]

>>> type(a)

<**class** 'list'>

Если у вас уже есть список и вы хотите создать его копию, то можно воспользоваться следующим способом:

>>> a = [1, 3, 5, 7]

>>> b = a[:]

>>> print(a)

[1, 3, 5, 7]

>>> print(b)

[1, 3, 5, 7]

или сделать это так:

>>> a = [1, 3, 5, 7]

>>> b = list(a)

>>> print(a)

[1, 3, 5, 7]

>>> print(b)

[1, 3, 5, 7]

В случае, если вы выполните простое присвоение списков друг другу, то переменной *b* будет присвоена ссылка на тот же элемент данных в памяти, на который ссылается *a*, а не копия списка *а*. Т.е. если вы будете изменять список *a*, то и *b* тоже будет меняться.

>>> a = [1, 3, 5, 7]

>>> b = a

>>> print(a)

[1, 3, 5, 7]

>>> print(b)

[1, 3, 5, 7]

>>> a[1] = 10

>>> print(a)

[1, 10, 5, 7]

>>> print(b)

[1, 10, 5, 7]

Добавление элемента в список осуществляется с помощью метода append().

>>> a = []

>>> a.append(3)

>>> a.append("hello")

>>> print(a)

[3, 'hello']

Для удаления элемента из списка, в случае, если вы знаете его значение, используйте метод remove(x), при этом будет удалена первая ссылка на данный элемент.

>>> b = [2, 3, 5]

>>> print(b)

[2, 3, 5]

>>> b.remove(3)

>>> print(b)

[2, 5]

Если необходимо удалить элемент по его индексу, воспользуйтесь командой *del имя\_списка[индекс]*.

>>> c = [3, 5, 1, 9, 6]

>>> print(c)

[3, 5, 1, 9, 6]

>>> del c[2]

>>> print(c)

[3, 5, 9, 6]

Изменить значение элемента списка, зная его индекс, можно напрямую к нему обратившись.

>>> d = [2, 4, 9]

>>> print(d)

[2, 4, 9]

>>> d[1] = 17

>>> print(d)

[2, 17, 9]

Очистить список можно просто заново его проинициализировав, так как будто вы его вновь создаете. Для получения доступа к элементу списка укажите индекс этого элемента в квадратных скобках.

>>> a = [3, 5, 7, 10, 3, 2, 6, 0]

>>> a[2]

7

Можно использовать отрицательные индексы, в таком случае счет будет идти с конца, например для доступа к последнему элементу списка можно использовать вот такую команду:

>>> a[-1]

0

Для получения из списка некоторого подсписка в определенном диапазоне индексов, укажите начальный и конечный индекс в квадратных скобках, разделив их двоеточием.

>>> a[1:4]

[5, 7, 10]

**Методы списков**

***list.append(x)***

Добавляет элемент в конец списка. Ту же операцию можно сделать так *a[len(a):] = [x]*.

>>> a = [1, 2]

>>> a.append(3)

>>> print(a)

[1, 2, 3]

***list.extend(L)***

Расширяет существующий список за счет добавления всех элементов из списка *L*. Эквивалентно команде *a[len(a):] = L*.

>>> a = [1, 2]

>>> b = [3, 4]

>>> a.extend(b)

>>> print(a)

[1, 2, 3, 4]

***list.insert(i, x)***

Вставить элемент *x* в позицию *i*.  Первый аргумент – индекс элемента после которого будет вставлен элемент *x*.

>>> a = [1, 2]

>>> a.insert(0, 5)

>>> print(a)

[5, 1, 2]

>>> a.insert(len(a), 9)

>>> print(a)

[5, 1, 2, 9]

***list.remove(x)***

Удаляет первое вхождение элемента *x* из списка.

>>> a = [1, 2, 3]

>>> a.remove(1)

>>> print(a)

[2, 3]

***list.pop([i])***

Удаляет элемент из позиции *i* и возвращает его. Если использовать метод без аргумента, то будет удален последний элемент из списка.

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

>>> print(a.pop(2))

3

>>> print(a.pop())

5

>>> print(a)

[1, 2, 4]

***list.clear()***

Удаляет все элементы из списка. Эквивалентно *del a[:]*.

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

>>> print(a)

[1, 2, 3, 4, 5]

>>> a.clear()

>>> print(a)

[]

***list.index(x[, start[, end]])***

Возвращает индекс элемента.

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

>>> a.index(4)

3

***list.count(x)***

Возвращает количество вхождений элемента *x* в список.

>>> a=[1, 2, 2, 3, 3]

>>> print(a.count(2))

2

***list.sort(key=None, reverse=False)***

Сортирует элементы в списке по возрастанию. Для сортировки в обратном порядке используйте флаг *reverse=True*. Дополнительные возможности открывает параметр *key*, за более подробной информацией обратитесь к документации.

>>> a = [1, 4, 2, 8, 1]

>>> a.sort()

>>> print(a)

[1, 1, 2, 4, 8]

***list.reverse()***

Изменяет порядок расположения элементов в списке на обратный.

>>> a = [1, 3, 5, 7]

>>> a.reverse()

>>> print(a)

[7, 5, 3, 1]

***list.copy()***

Возвращает копию списка. Эквивалентно *a[:]*.

>>> a = [1, 7, 9]

>>> b = a.copy()

>>> print(a)

[1, 7, 9]

>>> print(b)

[1, 7, 9]

>>> b[0] = 8

>>> print(a)

[1, 7, 9]

>>> print(b)

[8, 7, 9]

**List Comprehensions**

*List Comprehensions* чаще всего на русский язык переводят как  абстракция списков или списковое включение, является частью синтаксиса языка, которая предоставляет простой способ построения списков. Проще всего работу *list comprehensions* показать на примере. Допустим вам необходимо создать список целых чисел от *0* до *n*, где *n* предварительно задается. Классический способ решения данной задачи выглядел бы так:

>>> n = int(input())

7

>>> a=[]

>>> **for** i **in** range(n):

a.append(i)

>>> print(a)

[0, 1, 2, 3, 4, 5, 6]

Использование *list comprehensions* позволяет сделать это значительно проще:

>>> n = int(input())

7

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

>>> print(a)

[0, 1, 2, 3, 4, 5, 6]

или вообще вот так, в случае если вам не нужно больше использовать *n:*

>>> a = [i **for** i **in** range(int(input()))]

7

>>> print(a)

[0, 1, 2, 3, 4, 5, 6]

***List Comprehensions* как обработчик списков**

В языке *Python*есть две очень мощные функции для работы с коллекциями: *map*и *filter*. Они позволяют использовать функциональный стиль программирования, не прибегая к помощи циклов, для работы с такими типами как *list*, *tuple*, *set*, *dict*и т.п. Списковое включение позволяет обойтись без этих функций. Приведем несколько примеров для того, чтобы понять о чем идет речь.

Пример с заменой функции *map*.

Пусть у нас есть список и нужно получить на базе него новый, который содержит элементы первого, возведенные в квадрат. Решим эту задачу с использованием циклов:

>>> a = [1, 2, 3, 4, 5, 6, 7]

>>> b = []

>>> **for** i **in** a:

b.append(i\*\*2)

>>> print('a = {}\nb = {}'.format(a, b))

a = [1, 2, 3, 4, 5, 6, 7]

b = [1, 4, 9, 16, 25, 36, 49]

Та же задача, решенная с использованием *map*, будет выглядеть так:

>>> a = [1, 2, 3, 4, 5, 6, 7]

>>> b = list(map(**lambda** x: x\*\*2, a))

>>> print('a = {}\nb = {}'.format(a, b))

a = [1, 2, 3, 4, 5, 6, 7]

b = [1, 4, 9, 16, 25, 36, 49]

В данном случае применена *lambda*-функция, о том, что это такое и как ее использовать можете прочитать [здесь](https://devpractice.ru/python-lesson-10-functions-in-python/).

Через списковое включение эта задача будет решена так:

>>> a = [1, 2, 3, 4, 5, 6, 7]

>>> b = [i\*\*2 **for** i **in** a]

>>> print('a = {}\nb = {}'.format(a, b))

a = [1, 2, 3, 4, 5, 6, 7]

b = [1, 4, 9, 16, 25, 36, 49]

Пример с заменой функции *filter*.

Построим на базе существующего списка новый, состоящий только из четных чисел:

>>> a = [1, 2, 3, 4, 5, 6, 7]

>>> b = []

>>> **for** i **in** a:

**if** i%2 == 0:

b.append(i)

>>> print('a = {}\nb = {}'.format(a, b))

a = [1, 2, 3, 4, 5, 6, 7]

b = [2, 4, 6]

Решим эту задачу с использованием *filter*:

>>> a = [1, 2, 3, 4, 5, 6, 7]

>>> b = list(filter(**lambda** x: x % 2 == 0, a))

>>> print('a = {}\nb = {}'.format(a, b))

a = [1, 2, 3, 4, 5, 6, 7]

b = [2, 4, 6]

Решение через списковое включение:

>>> a = [1, 2, 3, 4, 5, 6, 7]

>>> b = [i **for** i **in** a **if** i % 2 == 0]

>>> print('a = {}\nb = {}'.format(a, b))

a = [1, 2, 3, 4, 5, 6, 7]

b = [2, 4, 6]

**Слайсы / Срезы**

Слайсы (срезы) являются очень мощной составляющей *Python*, которая позволяет быстро и лаконично решать задачи выборки элементов из списка. Выше уже был пример использования слайсов, здесь разберем более подробно работу с ними. Создадим список для экспериментов:

>>> a = [i **for** i **in** range(10)]

Слайс задается тройкой чисел, разделенных запятой: *start:stop:step*. *Start*– позиция с которой нужно начать выборку, *stop*– конечная позиция, *step*– шаг. При этом необходимо помнить, что выборка не включает элемент определяемый *stop*.

Рассмотрим примеры:

>>> # Получить копию списка

>>> a[:]

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

>>> # Получить первые пять элементов списка

>>> a[0:5]

[0, 1, 2, 3, 4]

>>> # Получить элементы с 3-го по 7-ой

>>> a[2:7]

[2, 3, 4, 5, 6]

>>> # Взять из списка элементы с шагом 2

>>> a[::2]

[0, 2, 4, 6, 8]

>>> # Взять из списка элементы со 2-го по 8-ой с шагом 2

>>> a[1:8:2]

[1, 3, 5, 7]

Слайсы можно сконструировать заранее, а потом уже использовать по мере необходимости. Это возможно сделать, в виду того, что слайс – это объект класса *slice*. Ниже приведен пример, демонстрирующий эту функциональность:

>>> s = slice(0, 5, 1)

>>> a[s]

[0, 1, 2, 3, 4]

>>> s = slice(1, 8, 2)

>>> a[s]

[1, 3, 5, 7]

**Тип “*List Comprehensions*”… в генераторном режиме**

Есть ещё одни способ создания списков, который похож на списковое включение, но результатом работы является не объект класса *list*, а генератор.

Предварительно импортируем модуль *sys*, он нам понадобится:

>>> **import** sys

Создадим список, используя списковое включение :

>>> a = [i **for** i **in** range(10)]

проверим тип переменной *a*:

>>> type(a)

<**class** 'list'>

и посмотрим сколько она занимает памяти в байтах:

>>> sys.getsizeof(a)

192

Для создания объекта-генератора, используется синтаксис такой же как и для спискового включения, только вместо квадратных скобок используются круглые:

>>> b = (i **for** i **in** range(10))

>>> type(b)

<**class** 'generator'>

>>> sys.getsizeof(b)

120

Обратите внимание, что тип этого объекта *‘generator’*, и в памяти он занимает места меньше, чем список, это объясняется тем, что в первом случае в памяти хранится весь набор чисел от 0 до 9, а во втором функция, которая будет нам генерировать числа от 0 до 9. Для наших примеров разница в размере не существенна, рассмотрим вариант с 10000 элементами:

>>> c = [i **for** i **in** range(10000)]

>>> sys.getsizeof(c)

87624

>>> d = (i **for** i **in** range(10000))

>>> sys.getsizeof(d)

120

Сейчас уже разница существенна, как вы уже поняли, размер генератора в данном случае не будет зависеть от количества чисел, которые он должен создать.

Если вы решаете задачу обхода списка, то  принципиальной разницы между списком и генератором не будет:

>>> **for** val **in** a:

print(val, end=' ')

0 1 2 3 4 5 6 7 8 9

>>> **for** val **in** b:

print(val, end=' ')

0 1 2 3 4 5 6 7 8 9

Но с генератором нельзя работать также как и со списком: нельзя обратиться к элементу по индексу и т.п.

## ****Что такое кортеж (tuple) в Python?****

Кортеж (tuple) – это неизменяемая структура данных, которая по своему подобию очень похожа на список. Как вы наверное знаете, список – это изменяемый тип данных. Т.е. если у нас есть список a = [1, 2, 3] и мы хотим заменить второй элемент с 2 на 15, то мы может это сделать, напрямую обратившись к элементу списка.

>>> a = [**1**, **2**, **3**]

>>> print(a)

[**1**, **2**, **3**]

>>> a[**1**] = **15**

>>> print(a)

[**1**, **15**, **3**]

С кортежем мы не можем производить такие операции, т.к. элементы его изменять нельзя.

>>> b = (**1**, **2**, **3**)

>>> print(b)

(**1**, **2**, **3**)

>>> b[**1**] = **15**

Traceback (most recent call last):

File "<pyshell#6>", line **1**, in <module>

b[**1**] = **15**

TypeError: 'tuple' object does not support item assignment

## ****Зачем нужны кортежи в Python?****

Существует несколько причин, по которым стоит использовать кортежи вместо списков. Одна из них – это обезопасить данные от случайного изменения. Если мы получили откуда-то массив данных, и у нас есть желание поработать с ним, но при этом непосредственно менять данные мы не собираемся, тогда, это как раз тот случай, когда кортежи придутся как нельзя кстати. Используя их в данной задаче, мы дополнительно получаем сразу несколько бонусов – во-первых, это экономия места. Дело в том, что кортежи в памяти занимают меньший объем по сравнению со списками.

>>> lst = [**10**, **20**, **30**]

>>> tpl = (**10**, **20**, **30**)

>>> print(lst.\_\_sizeof\_\_())

**32**

>>> print(tpl.\_\_sizeof\_\_())

**24**

Во-вторых – прирост производительности, который связан с тем, что кортежи работают быстрее, чем списки (т.е. на операции перебора элементов и т.п. будет тратиться меньше времени). Важно также отметить, что кортежи можно использовать в качестве ключа у словаря.

## ****Создание, удаление кортежей и работа с его элементами****

### ****Создание кортежей****

Для создания пустого кортежа можно воспользоваться одной из следующих команд.

>>> a = ()

>>> print(type(a))

<class 'tuple'>

>>> b = tuple()

>>> print(type(b))

<class 'tuple'>

Кортеж с заданным содержанием создается также как список, только вместо квадратных скобок используются круглые.

>>> a = (**1**, **2**, **3**, **4**, **5**)

>>> print(type(a))

<class 'tuple'>

>>> print(a)

(1, 2, 3, 4, 5)

При желании можно воспользоваться функцией tuple().

>>> a = tuple((**1**, **2**, **3**, **4**))

>>> print(a)

(**1**, **2**, **3**, **4**)

### ****Доступ к элементам кортежа****

Доступ к элементам кортежа осуществляется также как к элементам списка – через указание индекса. Но, как уже было сказано – изменять элементы кортежа нельзя!

>>> a = (**1**, **2**, **3**, **4**, **5**)

>>> print(a[**0**])

**1**

>>> print(a[**1**:**3**])

(**2**, **3**)

>>> a[**1**] = **3**

Traceback (most recent call last):

File "<pyshell#24>", line **1**, in <module>

a[**1**] = **3**

TypeError: 'tuple' object does not support item assignment

### ****Удаление кортежей****

Удалить отдельные элементы из кортежа невозможно.

>>> a = (**1**, **2**, **3**, **4**, **5**)

>>> del a[**0**]

Traceback (most recent call last):

File "<pyshell#26>", line **1**, in <module>

del a[**0**]

TypeError: 'tuple' object doesn't support item deletion

Но можно удалить кортеж целиком.

>>> del a

>>> print(a)

Traceback (most recent call last):

File "<pyshell#28>", line **1**, in <module>

print(a)

NameError: name 'a' is not defined

### ****Преобразование кортежа в список и обратно****

На базе кортежа можно создать список, верно и обратное утверждение. Для превращения списка в кортеж достаточно передать его в качестве аргумента функции tuple().

>>> lst = [**1**, **2**, **3**, **4**, **5**]

>>> print(type(lst))

<class 'list'>

>>> print(lst)

[**1**, **2**, **3**, **4**, **5**]

>>> tpl = tuple(lst)

>>> print(type(tpl))

<class 'tuple'>

>>> print(tpl)

(**1**, **2**, **3**, **4**, **5**)

Обратная операция также является корректной.

>>> tpl = (**2**, **4**, **6**, **8**, **10**)

>>> print(type(tpl))

<class 'tuple'>

>>> print(tpl)

(**2**, **4**, **6**, **8**, **10**)

>>> lst = list(tpl)

>>> print(type(lst))

<class 'list'>

>>> print(lst)

[**2**, **4**, **6**, **8**, **10**]

**Примечания к выполнению лабораторной работы:**

1. В первом задании необходимо использовать кортежи (tuple);
2. Во втором и третьем заданиях использовать списки (list);
3. В четвертом задании использовать множество (set);

4. Реализовать три способа заполнения массива:

* Заполнение с клавиатуры (с помощью рекурсивной функции);
* Заполнение в цикле с использованием random;
* Заполнение с помощью генератора.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Номер варианта** | **№№ заданий** | | | | **Номер варианта** | **№№ заданий** | | | |
| ***1*** | ***2*** | ***3*** | ***4*** | ***1*** | ***2*** | ***3*** | ***4*** |
| **1** | 1 | 6 | **11** | **16** | **16** | 16 | 21 | **26** | **1** |
| **2** | 2 | 7 | **12** | **17** | **17** | 17 | 22 | **27** | **2** |
| **3** | 3 | 8 | **13** | **18** | **18** | 18 | 23 | **28** | **3** |
| **4** | 4 | 9 | **14** | **19** | **19** | 19 | 24 | **29** | **4** |
| **5** | 5 | 10 | **15** | **20** | **20** | 20 | 25 | **30** | **5** |
| **6** | 6 | 11 | **16** | **21** | **21** | 21 | 26 | **1** | **6** |
| **7** | 7 | 12 | **17** | **22** | **22** | 22 | 27 | **2** | **7** |
| **8** | 8 | 13 | **18** | **23** | **23** | 23 | 28 | **3** | **8** |
| **9** | 9 | 14 | **19** | **24** | **24** | 24 | 29 | **4** | **9** |
| **10** | 10 | 15 | **20** | **25** | **25** | 25 | 30 | **5** | **10** |
| **11** | 11 | 16 | **21** | **26** | **26** | 26 | 1 | **6** | **11** |
| **12** | 12 | 17 | **22** | **27** | **27** | 27 | 2 | **7** | **12** |
| **13** | 13 | 18 | **23** | **28** | **28** | 28 | 3 | **8** | **13** |
| **14** | 14 | 19 | **24** | **29** | **29** | 29 | 4 | **9** | **14** |
| **15** | 15 | 20 | **25** | **30** | **30** | 30 | 5 | **10** | **15** |

***Задание 1*. Обработка элементов одномерного массива**

1. Даны натуральное число *n* и вещественный массивиз *n* элементов*.* Выдать те элементы массива, индексы которых являются степенями двойки (1, 2, 4, 8, …).
2. Даны натуральное число *n* и вещественный массивиз *n* элементов*.* Выдать те элементы массива, индексы которых являются полными квадратами ( 1, 4, 9, 16, …).
3. Даны натуральное число *n* и вещественный массивиз *n* элементов*.* Выдать те элементы массива, индексы которых являются числами Фибоначчи ( 1, 2, 3, 5, 8, 13, …).
4. Даны натуральное число *n* и вещественный массив *x* из *n* элементов*.* Вычислить **
5. Даны натуральное число *n* и вещественный массивиз *n* элементов*.* Вычислить произведение положительных элементов массива, расположенных после минимального элемента. Если таких элементов нет, выдать соответствующее сообщение.
6. Даны натуральное число *n* и вещественный массивиз *n* элементов*.* Вычислить среднее арифметическое элементов массива, расположенных между максимальным и минимальным элементами, включая максимальный и минимальный элементы.
7. Даны натуральное число *n* и вещественный массив из *n* элементов*.* Найти отношение суммы положительных элементов, расположенных до максимального элемента, к сумме отрицательных элементов, расположенных после максимального элемента в массиве.   
   Если положительные элементы, расположенные перед максимальным, или отрицательные элементы, расположенные после максимального, отсутствуют, выдать соответствующее сообщение.
8. Даны натуральное число *n* и вещественный массив из *n* элементов*.* Вычислить среднее арифметическое элементов массива, найти максимальный элемент массива и величину отклонения среднего арифметического от максимального элемента.
9. Даны натуральное число *n* и вещественный массив *x* из *n* элементов*.* Вычислить *y =* *x*1 + *x*1 *x*2 + *x*1 *x*2 *x*3 +...+ *x*1 *x*2 ... *xm*, где *m* – номер первого отрицательного элемента массива, либо число *n*, если в массиве нет отрицательных элементов.
10. Даны натуральное число *n* и вещественный массив *x* из *n* элементов*.* Найти номер минимального элемента и проверить оставшуюся часть массива на симметричность.
11. Даны натуральное число *n* и вещественный массив *x* из *n* элементов. Вычислить среднее значение *xср* и дисперсию *d* для заданного массива наблюдений: 
12. Даны натуральное число *n* и вещественный массив *x* из *n* элементов*.* Вычислить среднее значение и дисперсию для заданного массива наблюдений более эффективным (по сравнению с предыдущей задачей) способом, за один проход массива: 
13. Даны натуральное число *n*, вещественное число *x* и вещественный массив *a* из *n* элементов*.* Вычислить значение многочлена 

а также его первую и вторую производные при заданном значении *x*.

1. Даны натуральное число *n* и вещественный массив *a* из *n* элементов*.* Подсчитать количество элементов массива, удовлетворяющих условию а также проверить, удовлетворяет ли этому условию максимальный элемент массива.
2. Даны натуральное число *n* и массив целых чисел из *n* элементов. Проверить, расположены ли в массиве элементы в порядке возрастания или в порядке убывания.
3. Даны натуральное число *n* и массив вещественных чисел *x* из *n* элементов (*n*≥2). Вычислить 
4. Даны натуральное число *n* и вещественный массив *x* из *n* элементов*.* Вычислить 
5. Даны натуральное число *n* и массив вещественных чисел *x* из *n* элементов (*n≥*3). Вычислить произведение . Среди сомножителей этого произведения найти максимальное значение.
6. Даны натуральное число *n* и массив целых положительных чисел из *n* элементов*.* Найти те элементы массива, в состав которых входит заданная цифра. Если такие элементы отсутствуют, выдать об этом сообщение.
7. Даны натуральное число *n* и массив *a* из *n* вещественных чисел. Найти y=ma*x*{*a*1 , (*a*1 ∙ *a*2 ), ... , (*a*1 ∙*a*2 ∙ ... ∙*an* ) }.
8. Даны натуральное число *n* и массив *x* вещественных чисел.(*n≥*4, *n* – кратно четырём). Вычислить сумму **Среди слагаемых этой суммы найти минимальное слагаемое.
9. Вычислить емкость батареи, составленной из *n* параллельно соединённых конденсаторов, по формуле
   * 1. *c*  *c*1  *c*2 ... *cn*
   1. и ёмкость батареи, составленной из последовательно соединённых конденсаторов, по формуле
      1. 1 1 1 1
   2.   ... . *c c*1 *c*2 *cn*
   3. Вычислить максимальную ёмкость конденсаторов.
10. На прямой расположены *n* точек: M1(*x*1), M2 (*x*2), ..., Mn(*xn*) .
11. Вычислить взаимные расстояния между всевозможными парами точек. Найти сумму этих расстояний и наименьшее расстояние.
12. Даны натуральное число *n (n*4*)* и массив *x* вещественных чисел из *n* элементов*.* Вычислить

*n k*

1

1

*k*

*i*

*i*

*x*









* 1. y= max{(*x*1  *x*2  *x*3  *x*4 ), (x1  *x*2  *x*3  *x*4  *x*5 ), . . . , (x1  *x*2 ... *xn* )}

1. Даны натуральное число *n*, вещественное число *c* и логический массив *x* из *n* элементов*.* Вычислить *y**ck* , где *k* – число элементов массива *x*, предшествующих первому по порядку элементу, имеющему значение *false* (если все элементы имеют значение *true*, то принять *k=n ).*
2. Даны натуральное число *n* и массив целых чисел из *n* элементов*.*
3. Найти номер первого нулевого элемента массива и проверить на убывание оставшуюся часть массива.
4. Найти среднее арифметическое элементов, предшествующих первому нулевому элементу. Если нулевой элемент в массиве отсутствует, выдать об этом сообщение.
5. Даны натуральное число *n* и массив целых чисел из *n* элементов*.* Известно, что среди элементов массива имеется по крайней мере один элемент со значением нуль.
6. Проверить на симметричность часть массива, расположенную между максимальным элементом и первым нулевым элементами.
7. Даны натуральное число *n* и массив целых чисел из *n* элементов*.*
8. Найти в массиве минимальный и максимальный элементы, а также их наибольший общий делитель.
9. Даны натуральное число *n*, вещественное число *r* и массиввещественных чисел из *n* элементов.
10. Проверить, имеется ли в массиве пара чисел, абсолютная величина разности между которыми меньше заданного значения *r*.
11. Даны натуральное нечетное число *n* (*n* ≥3) и массив из *n* элементов, не содержащий одинаковых элементов. Найти *средний по величине* элемент, то есть такой, что в массиве ровно *n* *div* 2 элементов меньше его и столько же элементов больше его*.* Массивсохранить, не сортировать, дополнительных массивов не использовать.
12. Например, в массиве (8, 3, 4, 7, 6, 9, 15) из 7 элементов *средний по величине* элемент – это число 7. Элементы, меньшие числа 7 : 3, 4, 6. Элементы, большие числа 7: 8, 9, 15. Количество элементов, меньших *среднего*, и количество элементов, больших *среднего*, равно 3.

***Задание 2*.** **Преобразование одномерного массива**

1. Даны натуральное число *n* и вещественный массив *a* из *n* элементов. Преобразовать массив по правилу: если *ai* >0, то увеличить *ai* на квадрат наименьшего элемента массива, иначе увеличить *ai* на квадрат наибольшего элемента массива.
2. Даны натуральное число *n* и вещественный массив из *n* элементов. Все элементы массива, принадлежащие отрезку [*a*, *b*], заменить средним арифметическим элементов массива. Элементы массива, меньшие *a*, заменить нулём. Элементы массива, большие *b*, заменить единицей. 3) Даны натуральное число *n* и вещественный массив из *n* элементов. Каждый элемент массива (кроме двух последних элементов) заменить большим из двух следующих за ним элементами.
3. Даны натуральное число *n* и вещественный массив из *n* элементов. Все элементы с нечётными номерами, предшествующие первому по порядку наибольшему элементу, умножить на этот наибольший элемент.
4. Даны натуральное число *n* и целочисленный массив из *n* элементов. В массиве поменять местами минимальный элемент и элемент, имеющий заданное значение. При отсутствии в массиве элемента с заданным значением выдать соответствующее сообщение.
5. Даны натуральное число *n* и целочисленный массив из *n* элементов. В массиве поменять местами элементы, имеющие заданные значения *x* и *y*. При отсутствии таких элементов выдать соответствующее сообщение. 7) Даны натуральное число *n* и вещественный массив из *n* элементов. Переставить элементы массива таким образом, чтобы вначале размещались все числа, меньшие единицы, а затем все остальные. Дополнительный массив не использовать. 8) Даны натуральное число *n* и вещественный массив из *n* элементов. Каждый третий элемент массива заменить полусуммой двух предыдущих элементов, а стоящий перед ним – полусуммой соседних с ним элементов. Дополнительный массив не использовать.
6. Даны натуральное число *n* и логический массив из *n* элементов. Каждый элемент логического массива, кроме первого и последнего, заменить логической суммой соседних с ним элементов.
7. Даны натуральное число *n* и вещественный массив *x* из *n* элементов. Все элементы массива, начиная с первого положительного, уменьшить в два раза.
8. Даны натуральное число *n* и целочисленный массив из *n* элементов. Найти минимальный, максимальный и первый нулевой (если он имеется) элементы. Поменять местами минимальный и максимальный элементы. Первый нулевой элемент заменить средним арифметическим положительных элементов. При отсутствии нулевого элемента выдать соответствующее сообщение.
9. Даны натуральные числа *n*, *p*, *q* и массив целых чисел из *n* элементов (*p*>*q*0). Заменить в массиве нулями те элементы массива, модули которых при делении на *p* дают в остатке число *q*. При отсутствии таких элементов выдать соответствующее сообщение.
10. Даны натуральное число *n* и массив вещественных чисел из *n* элементов. Все неотрицательные числа, не принадлежащие отрезку [1, 2], заменить единицей. Все остальные (отрицательные и принадлежащие заданному отрезку) – разделить на максимальное значение среди них.
11. Даны натуральное число *n* и вещественный массив *x* из *n* элементов.

Преобразовать массив по следующему правилу:

*xk*'  max{x 1, *x*2 , . . . , *xk* }, *k* = 1, 2, …, *n,*

где *xk*′ значение *к*го элемента после преобразования.

1. Даны натуральное число *n* и вещественный массив *x* из *n э*лементов. Преобразовать массив *x* по следующему правилу: *xk*'  (*xk*1  *xk*  *xk*1 )/3, *k* = 2, 3, …, *n-*1, где *xk*' – значение *к*-го элемента после преобразования.

Первый и последний элементы массива оставить без изменения. Вспомогательный массив не использовать.

1. Даны натуральное число *n* и вещественный массив *a* из *n* чисел.

Каждый элемент массива заменить значениями выражений *ai*'  *ai*1  2*ai*  *ai*1 , *i*  2,3,...,*n*1, *a*1' *a*1 *a*2 ; *an*' *an*1 *an* .

4 2 2

Вспомогательный массив не использовать.

1. Даны натуральное число *n* и вещественный массив *a* из *n* чисел. Каждый *j*-ый элемент массива, кроме первого заменить средним арифметическим значением первых *j* элементов.

Вспомогательный массив не использовать.

1. Даны натуральное число *n* и вещественный массив *a* из *n* чисел. Каждый *i*-ый элемент массива, кроме двух первых, увеличить на значение (*i-*2)го элемента. Вспомогательный массив не использовать.
2. Даны натуральное число *n* и вещественный массив из *n* чисел. Из каждого положительного элемента массива вычесть предыдущий элемент. Вспомогательный массив не использовать. При отсутствии положительных элементов выдать соответствующее сообщение.
3. Даны натуральное число *n*, вещественные числа *а*, *b* и вещественный массив *x* из *n* чисел. Поместить в начало массива все элементы, значения которых принадлежат отрезку [*а, b*]*,* затем – все остальные элементы, сохранив исходный порядок следования элементов.

Вспомогательный массив не использовать.

1. Даны натуральное число *n* и вещественный массив из *n* чисел. Элементы, расположенные в массиве между минимальным и максимальным элементами (включая эти элементы), расположить в обратном порядке. При отсутствии элементов между минимальным и максимальным значениями выдать соответствующее сообщение.
2. Даны натуральное число *n* и вещественный массив из *n* чисел. Наименьший элемент массива заменить средним арифметическим отрицательных элементов, а наибольший элемент возвести в степень, равную его номеру в массиве. При отсутствии отрицательных элементов в массиве выдать об этом сообщение.
3. Даны натуральное число *n* и вещественный массив из *n* чисел. Если массив упорядочен по неубыванию, то изменить в нём порядок следования элементов на обратный. В противном случае изменить порядок следования элементов на обратный в части массива, начиная с первого элемента массива до элемента, указывающего на первое нарушение упорядоченности. Вспомогательный массив не использовать.
4. Даны натуральное число *n* и вещественный массив из *n* чисел. Преобразовать массив по следующему правилу: все отрицательные элементы массива перенести в его начало, а все остальные перенести в конец, сохраняя исходное взаимное расположение как среди отрицательных, так и среди всех остальных. Вспомогательный массив не использовать.
5. Даны натуральное число *n* и вещественный массив из *n* чисел. Преобразовать массив по следующему правилу: все отрицательные элементы массива перенести в его конец, все положительные – в начало, все равные нулю – в середину, сохраняя исходное взаимное расположение, как среди отрицательных, так и среди всех остальных.

Вспомогательный массив не использовать.

1. Даны натуральное число *n* и вещественный массив из *n* чисел.

Поместить нулевые элементы в начало массива, затем разместить все остальные элементы, изменив порядок следования у ненулевых элементов на обратный.

Вспомогательный массив не использовать. 27) Даны натуральное число *n* и целочисленный массив из *n* чисел. Изменить порядок следования элементов с нечётными значениями на обратный.

Например, дан массив элементов (4, 5, 3, 8, 1, 2, 7), изменённый массив

– (4, 7, 1, 8, 3, 2, 5).

Если элементы с нечётными значениями отсутствуют, выдать об этом сообщение.

Вспомогательный массив не использовать. 28) Даны натуральное чётное число *n* и вещественный массив из *n* чисел. Наименьший элемент массива поместить на первое место, наименьший из оставшихся – на последнее место, следующий по величине – на второе место, следующий – на предпоследнее и так далее – до середины массива. Вспомогательный массив не использовать.

1. Даны натуральное число *n* и вещественный массив из *n* чисел.

Возвести каждый элемент в степень, показателем которой определяется номер элемента.

1. Даны натуральное число *n* и целочисленный массив из *n* чисел. Каждый элемент массива, являющийся простым числом, увеличить в два раза. Если массив не содержит простых чисел, выдать об этом сообщение.

***Задание 3.* Сдвиги, удаление, вставка**

1. Даны натуральное число *n* и вещественный массив из *n* элементов. Сдвинуть элементы массива на *k* позиций влево.

**Примечание.** При сдвиге элементов массива на *k* позиций влево освободившиеся *k* последних элемента заполнить нулями.

1. Даны натуральное число *n* и вещественный массив из *n* элементов. Сдвинуть элементы массива на *k* позиций вправо.

**Примечание.** При сдвиге элементов массива вправо на *k* позиций освободившиеся *k* первых элемента заполнить нулями.

1. Даны натуральное число *n* и вещественный массив из *n* элементов. Удалить из массива элемент, расположенный перед максимальным элементом. При отсутствии элементов перед максимальным выдать соответствующее сообщение.
2. Даны натуральное число *n* и вещественный массив из *n* элементов. Вставить в массив элемент с заданным значением после первого положительного элемента. Если массив не содержит ни одного положительного элемента, выдать об этом сообщение.
3. Даны натуральное число *n* и вещественный массив из *n* элементов. Вставить в массив элемент с заданным значением после элемента с максимальным значением и после элемента с минимальным значением. Если массив состоит из одинаковых элементов, выдать соответствующее сообщение.
4. Даны натуральное число *n* и вещественный массив из *n* элементов. Удалить из массива элемент, расположенный после элемента с максимальным значением и после элемента с минимальным значением. Если массив состоит из одинаковых элементов или какое-то удаление невозможно, выдать соответствующее сообщение.
5. Даны натуральное число *n* и целочисленный массив из *n* элементов.

Перед каждым элементом, имеющим значение *x*, вставить элемент со значением *у.*

Если массив не содержит ни одного элемента со значением *х*, выдать об этом сообщение.

1. и целочисленный массив из *n* элементов.

После каждого элемента, имеющего значение *x*, вставить элемент со значением *у*.

Если массив не содержит ни одного элемента со значением *х*, выдать об этом сообщение.

1. Даны натуральное число *n* и вещественный массив из *n* элементов. Удалить из массива элемент, расположенный перед каждым значением, равным максимальному элементу. При отсутствии таких элементов выдать соответствующее сообщение.
2. Даны натуральное число *n* и вещественный массив из *n* элементов. Удалить из массива элемент, расположенный после каждого значения, равного минимальному элементу. При отсутствии таких элементов выдать соответствующее сообщение.
3. Даны натуральное число *n* и целочисленный массив из *n* элементов. Удалить из массива максимальный элемент и вставить элемент с заданным значением *x* после первого элемента со значением *y*.

Если массив не содержит ни одного элемента со значением *у*, выдать об этом сообщение.

1. Даны натуральное число *n* и вещественный массив из *n* элементов. Удалить из массива элемент с заданным номером *k* и вставить элемент с заданным значением после первого отрицательного элемента.

Если массив не содержит ни одного отрицательного элемента или элемента с номером *k*, выдать об этом сообщение.

1. Даны натуральное число *n* и вещественный массив из *n* элементов. Вставить в массив элемент с заданным значением перед элементом с заданным номером *k* и перед первым отрицательным элементом.

Если массив не содержит ни одного отрицательного элемента или элемента с номером *k*, выдать соответствующее сообщение. 14) Даны натуральное число *n* и вещественный массив из *n* элементов. Вставить в массив элемент с заданным значением перед элементом с заданным номером *k*, а также после первого отрицательного элемента и после первого положительного элемента.

Если массив не содержит ни одного отрицательного или положительного элемента или элемента с номером *k*, выдать соответствующее сообщение. 15) Даны натуральное число *n* и целочисленный массив из *n* элементов. Удалить из массива нулевой элемент, расположенный перед первым отрицательным элементом, а также нулевой элемент, расположенный после первого положительного элемента.

Если массив не содержит ни одного такого (отрицательного, положительного, нулевого перед первым отрицательным, нулевого после первого положительного) элемента, выдать соответствующее сообщение.

1. и целочисленный массив из *n* элементов. За каждым элементом с заданным значением вставить его дубликат.

Если массив не содержит ни одного элемента с заданным значением, выдать об этом сообщение.

1. Даны натуральное число *n* и целочисленный массив из *n* элементов. Перед каждым элементом с заданным значением вставить элемент со значением нуль. Если массив не содержит ни одного элемента с заданным значением, выдать об этом сообщение.
2. Даны натуральное число *n* и вещественный массив из *n* элементов. Перед каждым элементом с отрицательным значением вставить элемент, равный абсолютной величине отрицательного элемента. Если массив не содержит ни одного отрицательного элемента, выдать об этом сообщение.
3. Даны натуральное число *n* и целочисленный массив из *n* элементов. В каждой группе подряд идущих одинаковых элементов оставить только один элемент. Если в массиве нет ни одной такой группы, выдать соответствующее сообщение.
4. Даны натуральное число *n* и целочисленный массив из *n* элементов. Удалить каждую группу подряд идущих одинаковых элементов. Если в массиве нет ни одной такой группы, выдать соответствующее сообщение.
5. Даны натуральное число *n* и вещественный массив из *n* элементов. Удалить из массива *k* элементов, расположенных перед первым отрицательным элементом. Если удаление невозможно, выдать соответствующее сообщение.
6. Даны натуральное число *n* и вещественный массив из *n* элементов. Продублировать *k* раз элемент, расположенный перед первым отрицательным элементом. Если указанная операция невозможна, выдать соответствующее сообщение.
7. Даны натуральное число *n* и целочисленный массив из *n* элементов. Удалить из массива наиболее длинную последовательность его нулевых элементов. При отсутствии в массиве последовательности нулевых элементов выдать соответствующее сообщение. 24) Даны натуральное число *n* и целочисленный массив из *n* элементов. После каждого элемента с заданным значением удалить один элемент, если он отличен от заданного значения*.* Если массив не содержит ни одной такой пары элементов, выдать об этом сообщение.
8. Даны натуральное число *n* и целочисленный массив из *n* элементов. Удалить из массива все вхождения максимального и минимального элементов. Если массив состоит из одинаковых элементов, выдать соответствующее сообщение.
9. и целочисленный массив из *n* элементов.

Удалить из массива все элементы, расположенные между первым и вторым нулевыми элементами. Если массив не содержит ни одного нулевого элемента или если нулевой элемент один, или если первый и второй нули идут подряд, выдать соответствующее сообщение.

1. Даны натуральное число *n* и целочисленный массив из *n* элементов. Вставить перед каждым элементом массива, расположенным между первым и вторым нулевыми элементами, заданное значение. Если массив не содержит ни одного нулевого элемента или если нулевой элемент один, или если первый и второй нули идут подряд, выдать об этом сообщение.
2. Даны натуральное число *n* и целочисленный массив из *n* элементов. Вставить за каждым элементом массива, значение которого кратно заданному значению, его номер в исходном массиве. При отсутствии таких элементов выдать соответствующее сообщение.
3. Даны натуральное число *n* и целочисленный массив из *n* элементов. Удалить из массива *a* “лишние” (кроме первого) элементы так, чтобы оставшиеся элементы образовывали возрастающую последовательность. При отсутствии “лишних” элементов выдать соответствующее сообщение.

30 ) Даны натуральные числа *n*, *m* и целочисленный массив*a* из *n* элементов. Массив просматривать циклически, начиная с первого элемента. За последним элементом массива просматривать первый элемент. Как только встретится *m*-ый элемент, его удалять из массива. Счёт возобновлять со следующего элемента после удалённого. Так продолжать, пока в массиве останется только один элемент. Найти номер оставшегося элемента

***Задание 4.* Работа с несколькими массивами**1) Даны натуральное число *n* и массивы *a* и *b* одного типа из *n* элементов. Получить массив *c*, элементы которого вычисляются по формулам: *c*1 *a*1 *bn*, *c*2 *a*2 *bn*1, ..., *cn* *an* *b*1 .

1. Даны натуральное число *n* и массив из *n* символов. Разбить исходный массив символов на два массива: в один поместить только буквы исходного массива, в другой массив – только цифры.
2. Даны натуральное число *n*, вещественный массив *a* из *n* элементов*.*

Выделить серии из подряд идущих положительных чисел. В массив *b*

поместить индексы первых элементов серий, в массив *c* – длины этих серий.

1. Даны натуральное число *n* и вещественные массивы *a* и *b* из *n* элементов каждый. Найти наименьшее значение среди тех элементов массива *a*, которые не являются элементами массива *b*.
2. Даны натуральные числа *n,* *m* и массивы *a* и *b* одного типа из *n* и *m* элементов соответственно. Получить массив *c* из *n*+*m* элементов, чередуя элементы массивов *a* и *b*. Оставшиеся элементы наиболее длинного массива разместить в конце массива *c*.
3. Даны натуральное число *n* и целочисленный массив *a* из *n* элементов. Все чётные элементы массива *a* поместить в массив *b*, а нечётные – в массив *c*.

Подсчитать количество тех и других.

1. Даны натуральное число *n*, вещественные числа *a* и *b* и вещественный массив *x* из *n* элементов. Переслать в массив *y* те элементы массива *x*, значения которых принадлежат заданному отрезку [*а*, *b*].
2. Даны натуральное число *n* и целочисленный массив *x* из *n* элементов. Переслать в массив *y* все элементы массива *x* с нечётными номерами, а в массив *z,* все элементы массива *x*, значения которых кратны 5.
3. Даны натуральное число *n* и вещественные массивы *a* и *b* из *n* элементов каждый. Преобразовать массивы по правилу: если элементы *a i* и *b i* оба отрицательны, то каждый из них увеличить на 0.5; если отрицательный только один из этих элементов, то отрицательный элемент заменить его квадратом, если оба элемента неотрицательны, то каждый из них заменить на среднее арифметическое исходных значений.
4. Даны натуральное число *n* и вещественный массив *a* из *n*2элементов. В массив *b* поместить сумму каждых *n* элементов массива *a*. Элементы массива *b* вычисляются по формулам:

*b*1 *a*1...*an*, *b*2 *an*1 ...*a*2*n*,

*b*3 *a*2*n*1 ...*a*3*n*

...

*bn* *a*(*n*1)*n*1 ...*an*2.

1. Даны натуральное число *n* и два вещественных вектора *a* и b из *n* элементов каждый. Найти угол между векторами, используя формулу:

*n*

(*A*,*B*) *i*1 *aibi* cos.

*A*  *B*













*n*

*i*

*i*

*n*

*i*

*i*

*b*

*a*

1

2

1

2

1. Даны натуральное число *n* и два вещественных массива *x* и y из *n* элементов каждый. Вычислить значение переменной R по формуле:

*n*

(*xi*  *s*)2 (*yi*  *d*)2

*R*  *i*1 ,















*n*

*i*

*n*

*i*

*i*

*i*

*d*

*y*

*s*

*x*

1

1

2

2

)

(

)

(

где *s* и *d* – среднее арифметическое элементов массивов *x* и *y* соответственно.

1. Даны натуральное число *n* и целочисленный массив из *n* элементов, содержащий повторяющиеся элементы. Все неповторяющиеся элементы исходного массива, поместить в другой массив.
2. Даны натуральное число *n* и целочисленный массив *a* из *n* элементов. В массив *b* поместить наиболее длинную возрастающую последовательность чисел массива *a*.
3. Даны натуральное число *n,* вещественный массив *a* из *n* элементов*.* В массив *b* поместить самую длинную убывающую последовательность чисел массива *a*.
4. Даны натуральные числа *n*, *m* и вещественные массивы *a* из *n* элементов и *b* из *m* элементов. Проверить, упорядочены ли массивы по убыванию. Если оба массива упорядочены по убыванию, то построить новый массив *c* из *n*+*m* элементов слиянием массивов *a* и *b* без нарушения упорядоченности. В противном случае в массиве *c* расположить вначале все элементы массива *a*, затем все элементы массива *b*.
5. Даны натуральные числа *n*, *m* и целочисленные массивы *a* из *n* элементов и *b* из *m* элементов. Поместить в массив *c* все элементы массива *a* и те элементы из массива *b*, которых нет в массиве *a*.
6. Даны натуральные числа *n*, *m* и целочисленные массивы *a* из *n* элементов и *b* из *m* элементов. Поместить в массив *c* по одному разу, те элементы, которые имеются как в массиве *a*, так и в массиве *b*.
7. Даны натуральные числа *n*, *m*, целое число *E* и целочисленные массивы *a* из *n* элементов и *b* из *m* элементов. Вставить в массив *a* за первым вхождением элемента *Е* все элементы массива *b*. Если значение *Е* не входит в массив *a*, то вставить максимальный элемент массива *b* за первым элементом с максимальным значением массива *a*.
8. Даны натуральные числа *n*, *m*, и целочисленные массивы *a* из *n* элементов и *b* из *m* элементов. Подсчитать, количество начальных совпадающих чисел в обоих массивах. Заполнить массив *c* числами, которые остались в массивах *a* и *b* после последнего совпадения чисел (начать числами первого массива, продолжить – числами второго массива).
9. Даны натуральное чётное число *n* и вещественный массив *a* из *n* элементов. Элементы массива *a* с чётными индексами перенести в массив *b*, а оставшиеся элементы – сдвинуть к началу массива *a*.
10. Даны натуральное число *n* и вещественный массив *a* из *n* элементов. Получить массив *b* из *n* элементов следующим образом. Выбрать из массива *a* наименьший элемент и поместить в начало массива *b*. Затем из оставшихся элементов массива *a* выбрать наибольший элемент и поместить в конец массив *b*. Продолжить выбор из оставшихся элементов массива *a* (наименьшие элементы помещать в начало массива *b*, а наибольшие – в конец массив *b*) до полного заполнения массива *b*.
11. Даны натуральное число *n* и вещественный массив *a* из *n*

неотрицательных чисел. Получить массив *b* из *n*-2 элементов следующим образом. Найти в массиве *a* элемент, для которого сумма элементов, стоящих до него, наименее отличается от суммы элементов, стоящих после него, и поместить найденный элемент в массив *b*. Продолжить выбор из оставшихся элементов массива *a,* пока в массиве *a* не останется 1 или 2 элемента.

1. Даны натуральное число *n* и вещественные массивы *a* и *b* из *n* элементов каждый. Массивы содержат значения координат точек на плоскости. Из заданного множества точек на плоскости выбрать такие три, которые составляют треугольник наибольшей площади.
2. Даны натуральное число *n* и вещественные массивы *a* и *b* из *n* элементов каждый. Массивы содержат значения координат точек на плоскости. Из заданного множества точек на плоскости выбрать такие три, которые составляют треугольник наибольшего периметра.
3. Даны натуральное число *n*, вещественное число *R* и вещественные массивы *x*, *y*, и *z* из *n* элементов каждый. Массивы содержат значения координат точек в трёхмерном пространстве. Найти такую из точек, что шар заданного радиуса *R* с центром в этой точке содержит минимальное число точек из данного множества точек.
4. Даны натуральное число *n* и вещественные массивы *x* и y из *n* элементов каждый. Массивы содержат значения координат точек на плоскости. Из заданного множества точек на плоскости выбрать такие три точки *A*, *B*, *C*, чтобы внутри треугольника *ABC* содержалось максимальное количество точек этого множества.
5. Даны натуральное число *n* и вещественные массивы *x* и *y* из *n* элементов каждый. Массивы содержат значения координат вершин выпуклого многоугольника. Вершины упорядочены. Определить, находится ли точка *a* с координатами *xa* , *ya* внутри многоугольника.
6. Даны натуральные числа *n*, *k*, вещественные числа *a*, *b* имассив вещественных чисел *x* из *n* элементов. Всечисла массива *x* расположены на интервале [*a*, *b*]. Интервал разбить на *k* равных участков. Вычислить плотности распределения чисел на отдельных участках по формуле: *p i* *mi* (1in), где *mi* - количество чисел массива *x* на *i* – ом участке. *n*

*n*

1. Рациональное алгебраическое выражение *z* *ai xki* , *ki*  *m*,

*i*1

*i*=1,2,…,*n* задано массивом коэффициентов *ai* и массивом соответствующих показателей степеней *ki* (несколько слагаемых могут иметь одинаковые показатели степени, порядок следования показателей степеней – произволен). Привести в алгебраическом выражении подобные члены и сформировать массив коэффициентов полученного многочлена *Pm* (*x*) по возрастанию степеней *x*.

**Контрольные вопросы**

1. Дайте определение массива. Перечислите особенности хранения и использования массивов?
2. Что понимают под индексом элемента массива?
3. Что называют размерностью массива?
4. Приведите отрывок кода для заполнения одномерного массива из 10 элементов с клавиатуры и случайным образом.
5. Простые алгоритмы сортировки.
6. Сортировка Шелла
7. Быстрая сортировка
8. Линейный и двоичный поиск в массивах. Их достоинства и недостатки.
9. Передача массивов как параметров в методы (подпрограммы) и как результат функции
10. Инициализация массивов
11. Цикл foreach
12. Присваивание и сравнение массивов в Java
13. Какие стандартные алгоритмы по работе с одномерными массивами Вы знаете?

Контрольные вопросы:

* 1. В чем разница между list и set?
  2. В каких случаях применяются tuple?
  3. Плюсы и минусы tuple по сравнению с list.
  4. Приведите примеры некоторых операций над списками.