ФЕДЕРАЛЬНОЕ АГЕНТСТВО ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Иркутский государственный университет путей сообщения»
Сибирский колледж транспорта и строительства

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
К ПРАКТИЧЕСКИМ РАБОТАМ

по дисциплине

ОПЦ.06 Основы алгоритмизации и программирования

по специальности

09.02.01 Компьютерные системы и комплексы

базовая подготовка
среднего профессионального образования

Иркутск 2023

РАССМОТРЕНО:
ЦМК специальности 09.02.01
Компьютерные системы и комплексы
Протокол №9 от « 26 » мая 2023 г.
Председатель ЦМК:
Арефьева Н.В.

Разработчик: Саквенко Т.В., преподаватель первой категории Сибирского колледжа транспорта и строительства ФГБОУ ВО «Иркутский государственный университет путей сообщения».

СОДЕРЖАНИЕ

Практическая работа №1

Формализация поставленной задачи. Составление блок-схем линейных и разветвляющихся алгоритмов

Цель работы: Овладеть техникой составления алгоритмов.

Задачи работы: Научиться составлять линейные и разветвляющиеся алгоритмы.

Задание: Составить алгоритмы, используя предложенные задания.

Ознакомьтесь с теоретическим материалом, необходимым для выполнения работы:

Алгоритм — совокупность последовательных шагов, схема действий, приводящих к желаемому результату.

Разработать алгоритм означает разбить задачу на определенную последовательность шагов. От разработчика алгоритма требуется знание особенностей и правил составления алгоритмов.

Основные особенности и свойства алгоритмов:

1. Наличие ввода исходных данных.
2. Наличие вывода результата выполнения алгоритма.
3. Дискретность.
4. Формальность.
5. Определенность (точность).
6. Понятность.
7. Результативность (конечность).
8. Корректность.
9. Массовость.
10. Эффективность.

Алгоритмы можно записывать по-разному. Форма записи, состав и количество операций алгоритма зависит от того, кто будет исполнителем этого алгоритма.

Способы описания алгоритма:

1. Формульная запись
2. Табличная запись
3. Развернутая словесная
4. На алгоритмическом языке
5. Графический (в виде блок схемы)
6. На языке программирования

Основным элементарным действием в вычислительных алгоритмах является присваивание значения переменной величине. Если значение константы определено видом ее записи, то переменная величина получает конкретное значение только в результате присваивания.

Присваивание – это операция, которая значение выражения, стоящее справа от символа «=» запоминает в переменной или элементе массива, стоящем слева. При присваивании происходит преобразование типов данных, если они не совпадают.

Линейными называются алгоритмы, в которых все действия осуществляются последовательно друг за другом, при этом каждая команда выполняется только один раз строго после той команды, которая ей предшествует. Линейный алгоритм составляется из команд присваивания, ввода, вывода и обращения к вспомогательным алгоритмам.

Разветвляющимся называется алгоритм, в котором действие выполняется по одной из возможных ветвей решения задачи, в зависимости от выполнения условий. Каждое из возможных направлений дальнейших действий называется ветвью. В блок-схемах разветвление реализуется специальным блоком «Решение». Этот блок предусматривает возможность двух выходов. В самом блоке «Решение» записывается логическое условие, от выполнения которого зависят дальнейшие действия.

Циклическим называется алгоритм, в котором некоторая часть операций (тело цикла – последовательность команд) выполняется многократно, т.е. более одного раза.

Различают:

1.циклы с известным числом повторений (или со счетчиком);

2.циклы с неизвестным числом повторений (циклы с предусловием и циклы с постусловием).

В любом цикле должна быть переменная, которая управляет выходом из цикла, т.е. определяет число повторений цикла.

В циклах со счетчиком известно число повторений цикла, т.е. оно является фиксированным числом. В этом случае переменная, которая считает количество повторений (шагов) цикла, называется счетчиком цикла (или параметром цикла, или управляющей переменной цикла).

Циклы с предусловием – это такие циклы, в которых до начала выполнения тела цикла проверяется условие выполнения следующего шага цикла. Если значение этого условия истинно (т.е. условие выполняется), то выполняется тело цикла.

Цикл с постусловием также используется при неизвестном заранее количестве повторений цикла, но в отличие от цикла с предусловием здесь условие на выход из цикла проверяется после того, как выполнились операторы тела цикла.

Задание 1: Составить линейные алгоритмы, используя предложенные задания.

Рассмотрите примерные упражнения

1. Составление алгоритма в табличной форме для вычисления выражения :

$$y = \frac{8x - 1}{4x + 2}$$

Исходные данные: x.
Результат: y.

Последовательность шагов и описание действий:

1.a := 8x
2.b := a -1
3.c := 4x
4.d := c+2
5.y := b/d

2. Составление алгоритма вычисления площади поверхности цилиндра с диаметром D и высотой H в табличной форме.

Исходные данные: D, H.
Результат: S.

Последовательность шагов и описание действий:

1.a:= πD2
2.b:= a / 4
3.c:= πDH
4.S:= b + c

3. Вычислить факториал числа n (c=n!), который вычисляется по формуле c=1*2*3*4*…*n. Алгоритм представить в словесной форме.

Исходные данные: n.
Результат: c.

Алгоритм:

1.Полагаем c=1 и переходим к следующему пункту.
2.Полагаем i=1 и переходим к следующему пункту.
3.Полагаем c=i*c и переходим к следующему пункту.
4.Проверяем, равно ли i числу n. Если i=n, то вычисления прекращаем. Если i больше n, то увеличиваем i на 1 и переходим к пункту 3.

Выполнить самостоятельно

Составьте алгоритмы решения следующих задач:
1.По заданным формулам составить вычислительные алгоритмы в виде таблиц:

$$S = \frac{\pi D^2}{4} + \frac{\pi DL}{2}$$

(S – площадь боковой поверхности конуса, D – диаметр основания, L - образующая);

$$f = \frac{1}{2\pi\sqrt{LC}}$$

(f – частота собственных колебаний в контуре, L – индуктивность катушки, C – емкость конденсатора).

2. Дано значение x. Получить значения -2x+3x2–4x3 и 1+2x+3x2–4x3 . Позаботиться об экономии операций.

3. Дано значение a. Не используя никаких функций и никаких операций, кроме умножения, получить значение a8 за три операции и a10 за четыре операции.

4. Составить алгоритм для вычисления пути, пройденного лодкой, если ее скорость в стоячей воде v км/ч, скорость течения реки v1 км/ч, время движения по озеру t1 ч, а против течения реки – t2 ч. Использовать словесный способ записи алгоритма.

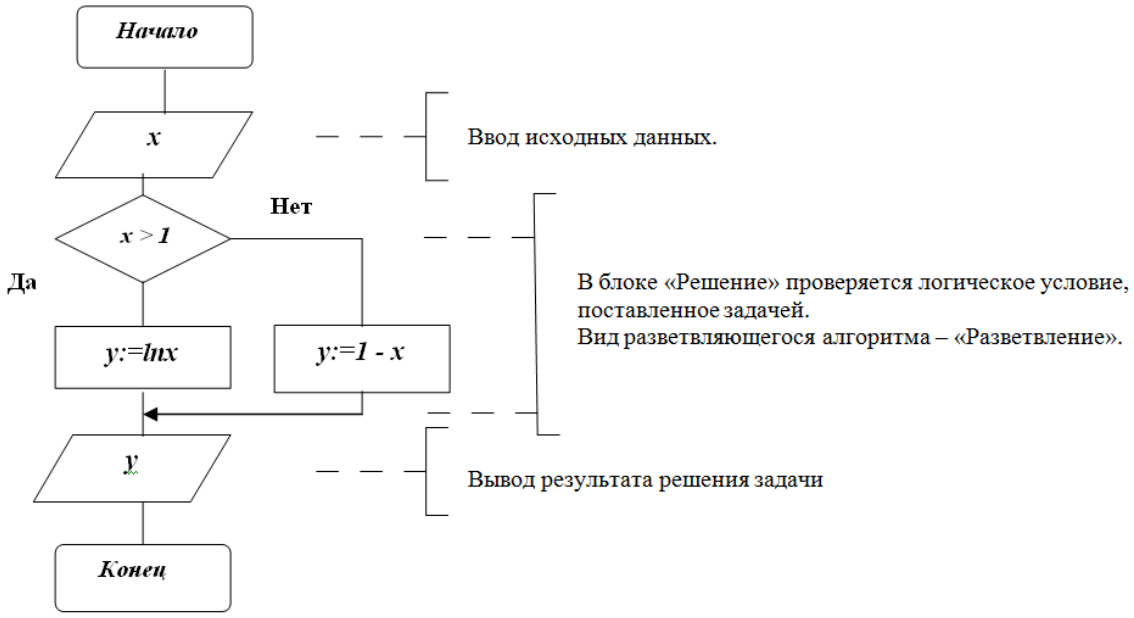
Задание 2: Составить разветвляющиеся алгоритмы, используя предложенные задания.

Рассмотрите примерные упражнения:

1. Составьте блок-схему вычисления значения функций:

$$y = \begin{cases} \ln x, \ x > 1, \\ 1 - x, \ x \le 1 \end{cases}$$

Исходные данные: x. Результат: y.



Выполнить самостоятельно

Составьте алгоритмы решения следующих задач:

1. Изобразите блок-схему простого диалогового алгоритма, который обращается к пользователю с просьбой ввести сначала строку имя, а затем строку настроение. В результате диалога может появиться следующий совместный текст:

Программа> Здравствуйте! Как Ваше имя?
Пользователь> Гаврик
Программа> Доброе утро, Гаврик! Как настроение?
Пользователь> так себе
Программа> У меня тоже так себе, Гаврик!

2. Даны три действительных числа. Возвести в квадрат те из них, значения которых неотрицательны, и в четвертую степень – отрицательные.

3. Составьте блок-схему вычисления значения функций:

$$y = \begin{cases} \dfrac{5x^2 + 2}{x + 4}, & \text{если } x > -4 \\ 3x^2 + 7, & \text{если } x \le -4 \end{cases}$$

4. Составьте программу, заменяющую меньшее из двух данных чисел суммой, а большее – произведением этих чисел.

5. Составить блок-схему алгоритма вычисления площади треугольника 3-мя различными способами на выбор пользователя.

Практическая работа №2

Программирование линейных алгоритмов.

Программирование задач с операторами условия.

Цель работы:

- Закрепить на практике теоретические знания по созданию программ с линейными алгоритмами, закрепить на практике навыки работы в Python
- научиться записывать на языке Python математические выражения;
- научиться использовать стандартные функции языка Python
- научиться работать с целыми типами данных;
- научиться работать с вещественными типами данных

Исходные данные (задание):

Задание 1 Написать в среде Python программы, реализующие следующие линейные алгоритмы, запустить их на выполнение.

$b=\left(1+tg^2\frac{z}{2}\right)$	$b=1+\frac{z^2}{3+z^2/5}$
$a=\frac{1+\sin^2(x+y)}{2+\left\lvert x-2x/\left(1+x^2y^2\right)\right\rvert}+x$	$a=\ln\left\lvert\left(y-\sqrt{\lvert x\rvert}\right)\right\rvert\left(x-\frac{y}{z+x^2/4}\right)$
$b=\cos^2\left(arctg\frac{1}{z}\right)$	$f=6{,}673\cdot10^{-8}\cdot\frac{m_1\cdot m_2}{r^2}$
$t=\frac{xyz-3{,}3\left\lvert x+\sqrt[4]{y}\right\rvert}{10^7+\sqrt{\lg 4}}$	$b=e^{\lvert x-y\rvert}\ln(1+e)\log_2 tg2$
$y=\frac{\beta+\sin^2\pi^4}{\cos2+\lvert ctg\gamma\rvert}$	$y=\sqrt[8]{x^8}+8^x$

Задачи: 2. Написать в среде Python программы, реализующие следующие линейные алгоритмы, запустить их на выполнение

1. Написать программу, которая для заданного целого числа a печатает следующую таблицу:

$$a$$

$$a^3 \quad a^6$$

$$a^6 \quad a^3 \quad a$$

2. Вычислить длину окружности, площади круга и объема шара одного и того же заданного радиуса.

3. Написать программу вычисления периметра и площади прямоугольного треугольника по длинам двух катетов.

4. Написать программу, которая по координатам трех вершин некоторого треугольника вычисляет его площадь и периметр.

5. Найти площадь кольца, внутренний радиус которого равен 20, а внешний – заданному числу r (r>20).

Порядок выполнения

1. Составить в тетради алгоритм
2. Написать программу
3. Отладить программу
4. Протестировать программу
5. Записать в тетради текст программы
6. Записать в тетради входные и выходные данные

Задание 3: Написать в среде Python программы, реализующие следующие алгоритмы, запустить их на выполнение.

1. Написать программу вычисления Y:

$$y = \begin{cases} x & , x \geq 1 \text{ или } x < -2 \\ 2x + 16 - x & , \text{остальных случаях} \end{cases}$$

2. Написать программу вычисления X:

$$x = \begin{cases} 14 & m = d, \quad l < k \\ 15 & m = d, \quad l \geq k \\ 16 & m \neq d \end{cases}$$

3. Составить программу для подсчета числа букв А, Б, В - в предложении. Предложение вводится отдельными символами до тех пор пока не введен символ (.).

4. В строке введенных символов подсчитать количество символов C, D, P, считая концом ввода символ (*).

5. По введенному номеру месяца напечатать наименование времени года и наименование месяца.

6. Улица Байкальская разбита на участки:

с 1 по 100 дом - 1 участок
с 101 по 203 дом - 2 участок
с 204 по 308 дом - 3 участок
остальные дома - 4 участок.

По введенному номеру дома выдать № участка.

7. Лежит ли введенное значение X на отрезке в интервале [4,5.4].

8. В пятиэтажном доме на каждом этаже расположено по четыре квартиры. Составить программу, которая по номеру этажа, печатает номера квартир на этом этаже.

9. Написать программу нахождения наибольшего числа из введенных чисел, считая концом ввода 0.

Практическая работа №3

Программирование задач с оператором цикла FOR

Цель работы: Научиться работать с оператором цикла FOR.

Задание 1: Программирование задач с оператором цикла FOR. Написать в среде Python программы, реализующие следующие алгоритмы, запустить их на выполнение.

1.Составить программу определения разрядности введенного целого числа.

2. Подсчитать сумму отрицательных чисел последовательности.

3. Составить программу подсчитывающую сумму цифр вводимого натурального числа.

4. Имеется одномерный массив из 15 чисел (a1, a2, a3 ... a15). С оставить программу их ввода. Упорядочить массив по убыванию.

5.Ввести 5-тизначное число. Вывести последовательность цифр начиная с конца.

Пример: ввод - 42891
вывод - 1 9 8 2 4

Практическая работа №4

Программирование задач с оператором цикла WHILE

Цель работы: Научиться работать с оператором цикла WHILE (цикл с предусловием).

Задание 2: Программирование задач с оператором цикла WHILE. Написать в среде Python программы, реализующие следующие алгоритмы, запустить их на выполнение.

1.Подсчитать ax*x+bx+c , при a=18.5, b=-0.5,c=134 , x изменяется от -1 до 0.6 с шагом 0.2. Значения выдавать на каждом шаге итерации.

2. Подсчитать сумму положительных чисел последовательности.

3. Подсчитать w=(a+bx)sqrt(x+1), при a=2.8, b=-0.3, x изменяется от 1 до 3 с шагом 0.5. Выдавать значения на каждом шаге итерации

4. Имеется одномерный массив из 20 чисел (B1,B2,B3 … B20). Составить программу их ввода. Упорядочить массив по возрастанию.

5. Подсчитать

$$zz = \begin{cases} fv/(f-6)x & x \geq 13.6 \\ (f+v)xx/v+3 & x < 4.8 \\ x/vf & \text{в остальных случаях} \end{cases}$$

Практическая работа №5, 6

Обработка кортежей и списков

Цель работы: Научиться работать с кортежами и списками

Теоретическая информация

В Python кортеж определяется как один из видов последовательностей, с которыми можно работать в этом языке программирования.
Последовательность данных, объединенная в кортеж, не позволяет изменять свои значения. В этом есть определенные преимущества:

1. увеличивается скорость обработки элементов кортежа, поскольку системе заранее известно, что значения не будут изменяться.
2. кортеж, может применяться для образования других структур
3. кортеж занимает меньше памяти, чем, например, списки, и, кроме того, элементы кортежа защищены от случайных изменений.

кортеж в Python может содержать совершенно разнородные объекты, например, строковые и числовые значения или звуковые файлы, файлы изображений.
Синтаксис объявления кортежей следующий:

Имя кортежа = (элемент1, элемент2, …элементN)

Например
korteg=(1, 2, 3, 4, 5)
Возможна и другая запись, например, для значений строкового типа,
Имя кортежа=(«Элемент1»,
«Элемент2»,
«Элемент3»,
«ЭлементN»)

Доступ к каждому элементу кортежа в программе осуществляется с помощью **индекса** – целого числа, служащего своеобразным именем элемента в кортеже.

При упоминании в программе элемента кортежа сразу за его именем должен следовать индекс элемента в квадратных скобках, например, korteg[i].

Нумерация элементов списка начинается с 0.

Обработка осуществляется в цикле «для».

У кортежей есть 2 встроенных метода:

1. **count()**

Ведет подсчет количества конкретного элемента в кортеже.

2. **index()**

Возвращает индекс первого появления элемента. Может смещать поиск от и до определенного индекса при задании дополнительных параметров.

Задание 1: Поясните результат работы методов

```
korteg = (1, 2, 2, 3, 1, 2, 4, 3, 2, 2)
korteg.count(3)
korteg.index(3)
korteg.index(2, 2)
korteg.index(2, 4, 6)
```

Задание 2. В кортеже, состоящего из заранее заданных целых чисел найдите сумму элементов.

```
sum=0
korteg=(1, 2, 3, 4, 5)
print("Кортеж")
for i in korteg:
print(i, end= " ")
for i in range(5):
sum=sum+korteg[i]
print("\nСумма элементов кортежа = ", sum)
```

Задание 3. Решите задачи

1. Составить программу, которая бы опрашивала присутствующих на уроке и сообщала о количестве отсутствующих. Всего в группе 12 человек.

2. Составить программу, которая подсчитывала бы какую литературу за неделю покупают в книжном магазине больше: художественную или техническую.

Обработка списков

Цель работы: Научиться писать программы с использованием списков

- научиться описывать одномерные массивы;
- научиться вводить одномерные массивы;
- научиться генерировать одномерные массивы;
- научиться выводить одномерные массивы;
- осуществлять поиск минимального и максимального элементов массива;
- упорядочивать массивы по возрастанию и убыванию.

Задание Написать в среде Turbo Pascal программы, реализующие следующие алгоритмы, запустить их на выполнение

1. Вычисляя значения переменной $x=8d+f$ при всех значениях d=1,2,3 и f = -3, 3, -6. Создать одномерный массив. Вывести значения элементов этого массива и значения d, f.

2. Создать одномерный массив из случайно сгенерированных чисел (от 1 до 36). Подсчитать сумму нечетных чисел элементов массива. Выдать на экран массив и найденную сумму.

3. Из случайно сгенерированных чисел (от 1 до 25) создать одномерный массив. Выдать массив на экран в виде:

```
a1
        a2
                a3
                        a4
```

4. Вычисляя значения переменной Z=3A/(9+X) при всех значениях A=0,12,20 и X=-2,-3,-8. Создать одномерный массив. Вывести значения элементов этого массива на экран:

z(a1,x1)	z(a1,x2)	z(a1,x3)
z(a2,x1)	z(a2,x2)	z(a2,x3)
z(a3,x1)	z(a3,x2)	z(a3,x3)

5. Ввести данные для формирования одномерного массива X[10]. Поменять местами элементы массива, введя их номера с клавиатуры.

Практическая работа №7,8

Использование стандартных функций и процедур для работы со строками

Цель работы:

Научиться писать программы по программированию строковых массивов

Задание 1. Написать в среде Python программы, реализующие следующие алгоритмы, запустить их на выполнение

1. Найти наибольшее слово в тексте

```
txt = input()
c =txt.split()
v = []
for i in c:
    v.append(len(i))
        g = v.index(max(v))
print (c[g])
```

2. Найти в тексте все слова длиной более 10 символов

```
txt = input()
c =txt.split()
v = []
for i in c:
   v.append(len(i))
for j in range (len(c)):
   if v[j]>=10:
       print (c[j])
```

3. Равна ли сумма цифр заданного числа некоторому третьему числу

```
a=input()
c=int(input())
spisok=list(a)
for i in range(len(spisok)):
    spisok[i]=int(spisok[i])
if sum(spisok)==c:
    print ("yes")
```

Задание 2. Решить задачи. Оформить в тетеради алгоритм в форме блок-схемы.

1. Определить, сколько раз в тексте встречается заданная буква.

2. Во введенной строке заменить один символ на другой
3. Во введенной строке удалить заданный символ
4. В заданной строке вставить после каждого знака препинания пробел.
5. Подсчитать, сколько слов в тексте начинается на заданную букву
6. Разработайте программу, которая зашифровывает введенный с клавиатуры текст. Процесс шифровки производится следующим образом: из десятичного кода каждого введенного с клавиатуры символа вычитается число десять. Получившаяся в результате вычитания величина интерпретируется как десятичный код некоторого другого символа, который и выводится на экран компьютера.

7. Разработайте программу, проверяющую степень надежности пароля пользователя, при этом критерии сложности пароля следующие:
   длина пароля свыше 8 символов, включает одновременно прописные буквы, строчные буквы и цифры.

Практическая работа №9

Обработка вложенных последовательностей (двумерных массивов). Базовые алгоритмы обработки.

Цель работы: Научиться писать программы на обработку двумерных массивов.

- научиться описывать двухмерные массивы;
- научиться вводить двухмерные массивы;
- научиться генерировать двухмерные массивы;
- научиться выводить двухмерные массивы;
- осуществлять поиск минимального и максимального элементов массива;
- упорядочивать массивы по возрастанию и убыванию.

Задание: Написать в среде Python программы, реализующие следующие алгоритмы, запустить их на выполнение:

Задача 1. Нахождение количества значений элементов вложенной последовательности больше 10.

```
Ввод матрицы
for i in range(n):
  for j in range(m):
    if a[i][j]>=10:
      kol+=1
print("\nКоличество элементов вложенной последовательности >= 10 равно ", kol)
```

Задача 2. Нахождение суммы значений элементов вложенной последовательности больших 10

```
Ввод матрицы
sum=0
for i in range(n):
  for j in range(m):
    if a[i][j]>=10:
      sum+=a[i][j]
print("\nСумма элементов вложенной последовательности >= 10 равно ", sum)
```

Задача 3. Нахождение произведения значений элементов вложенной последовательности при заданном условии

```
Ввод матрицы
pr=1
for i in range(n):
for j in range(m):
  if a[i][j]>=10:
    pr*=a[i][j]
print("\nПроизведение элементов вложенной последовательности >= 10 равно ", pr)
```

Вывод матрицы

```
for i in range(0, len(A)):
    for j in range(0, len(A[i])):
        print(A[i][j], end=' ')
print()
```

Ввод матрицы

```
n,m = (int(i) for i in input().split())
a = [[int(i) for i in input().split()] for _ in range(n)]
```

1. Найдите сумму элементов матрицы
2. Найдите сумму элементов матрицы по строкам, по столбцам
3. Найдите количество положительных, отрицательных и нулевых элементов матицы.
4. Задана матрица. Найти максимальный элемент в каждой строке
5. В матрице элементы главной диагонали замените на 0, все элементы выше главной диагонали на значение их суммы, а ниже главной диагонали на значение их произведения.

Практическая работа №10
Программирование задач сортировки массивов

Цель работы: Научиться писать программы по упорядочиванию массивов по возрастанию и убыванию

Сортировка данных

Сортировка массива – упорядочение элементов массива по возрастанию или убыванию.
Алгоритм сортировки состоит из:

1. сравнения, определяющего упорядоченность пары элементов;
2. перестановки, меняющей местами пару элементов.

Задача1: Упорядочить по возрастанию элементы данного действительного вектора.

И.д. n Є N, a(i) Є R, i=1,2,…n
В.д. a(j) Є R, j=1,2,…n
Связь: a(j) Є {a(i): a(1)<a(2)<…<a(n)}, i,j=1,2,…n.

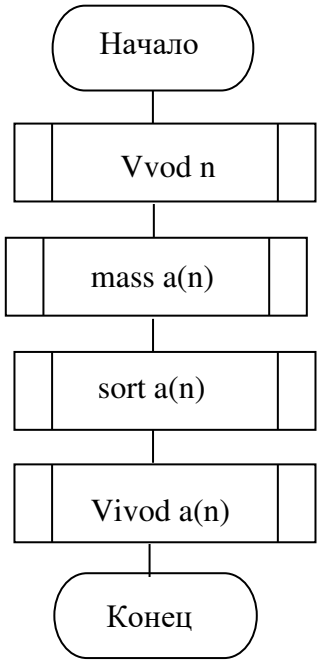
Тех. Задание

Условия на исходные данные	***Действия программы.***
n Є N и a(i) Є R, i=1,2,…n	Получить упорядоченную последовательность a(j)
n Є N\R и a(i) Є R, i=1,2,…n	Ввод заново значения n
n $\notin$ R или a(i) $\notin$ R, i=1,2,…n	ДСТ

Тесты

n	**a(i)**	**a(j)**	**Примечание**
4	(4, -5, 6, 3)	(-5, 3, 4, 6)	
1	3	3	
-1			Ввод заново значения n
1.5			Ввод заново значения n
*			ДСТ
2	(2, *)		ДСТ
3	(*, *, *)		ДСТ

Алгоритм.



Методы сортировки.

Метод пузырька (сортировка обменами).

Просматриваем последовательность чисел от конца к началу:

1. Сравниваем (n-1) и n элементы. Если они не расположены в порядке возрастания (убывания), то меняем их местами: значением n элемента будет значение (n-1), а значением (n-1) элемента будет значение n.

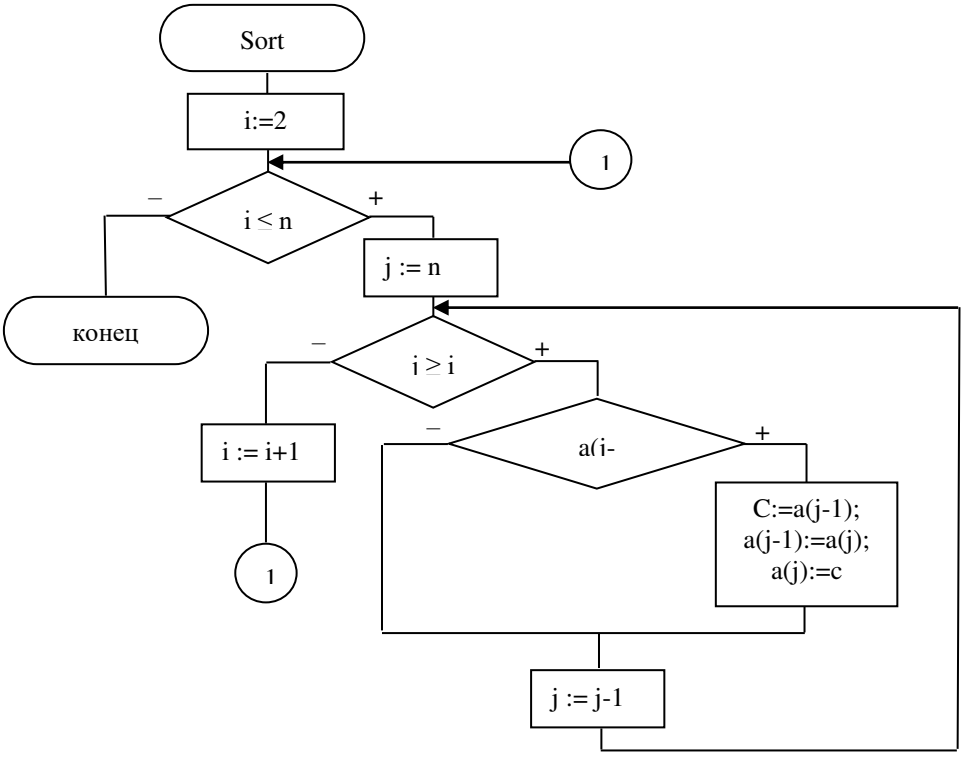
Если они расположены в порядке возрастания (убывания), то оставляем их на месте.

2. Сравниваем (n-2) и (n-1) элементы...
3. …

N. Сравниваем 1 и 2 элементы…

В результате просмотра получим упорядоченную таблицу, т.е. каждый следующий просмотр будет приводить к постановке очередного минимального элемента в левый конец рассматриваемой части таблицы.

№	**Элементы**	**Перестановки**									
1.	8	8	8	8↕	1	1	1	1	1	1	1
2.	3	3	3↕	1	8	8	8↕	2	2	2	2
3.	6	6↕	1	3	3	3↕	2	8	8↕	3	3
4.	1↕	1	6	6	6↕	2	3	3↕	3	8↕	6
5.	2	2	2	2	2	6	6	6	6	6	8

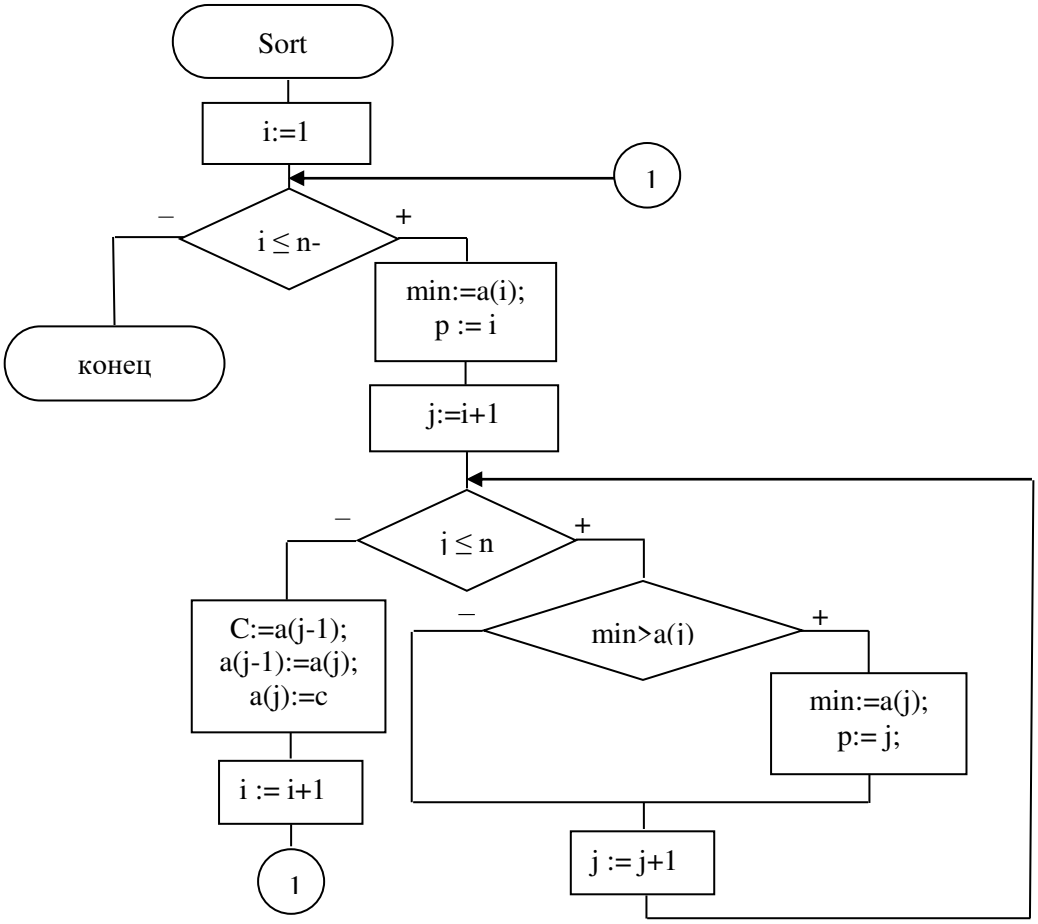


Минимаксный метод (сортировка выбором).

1. Среди элементов массива находим минимальный (максимальный) и меняем его местами с 1 элементом.
2. Среди оставшихся ищем минимальный (максимальный) и меняем его местами со 2 элементом.
3. …

N-1. Среди оставшихся ищем минимальный (максимальный) и меняем его местами с n элементом.

№	**Элементы**	**Перестановки**			
1.	8	1	1	1	1
2.	3	3	2	2	2
3.	6	6	6	3	3
4.	1 - min	8	8	8	6
5.	2	2 - min	3 - min	6 - min	8



Задача 2: Известен список сотрудников предприятия и их заработная плата. Упорядочить список фамилий сотрудников по возрастанию их заработной платы.
Задача 3: При зачислении в институт зачисляются абитуриенты, набравшие 15 баллов по результатам 3 экзаменов. После зачисления остается количество мест, которые заполняются в порядке убывания баллов. Известно количество человек, сдавших экзамены, количество мест, количество набранных баллов, организовать зачисление студентов.

Практическая работа №11

Программирование процедур и функций пользователя

Цель работы: Научиться работать с использованием процедур и функций пользователя

Задание 1. Написать в среде Python программы, реализующие следующие алгоритмы, запустить их на выполнение:

Подпрограмма – представляет собой группу операторов, к которым можно обратьcя из различных мест программы. Использование подпрограмм позволяет улучшить структурированность программы и сократить ее размер.

Подпрограммы:

Процедуры

Функции

Процедура - это независимая часть программы, которую можно вызвать в любой части программы. Процедура не может быть использована в операторах. Упоминание имени процедуры приводит к ее активизации и называется вызовом процедуры. Процедура используется для облегчения работы программиста, т.е. различные повторения в программе можно реализовать при их помощи.

Параметр – средство передачи данных в одном направлении: программа => процедура

Параметр-переменная – передача данных в двух направлениях: программа => процедура и

процедура => программа

Функция – это подпрограмма, результатом работы которой является некоторое значение, подобное переменной

Пример: Определите значение z

$$z = \frac{sing(x) + y^2}{sign(y) - \sqrt{|x|}}$$

где sign - сигнум-функция, а x и y - вещественные числа:

$$\text{sign(a)} = \begin{cases} 1, \text{если a} > 0 \\ 0. \text{если a} = 0 \\ -1, \text{если a} < 0 \end{cases}$$

```
def sgn(x):
  # global k
   if x>0:
      k=1
   elif x==0:
      k=0
   else:
      k=-1
   return k
```

```
a = float(input())
y = float(input())
z = (sgn(a)+y**2)/(sgn(y)-(abs(a))**0.5)
print("Ответ:", z)
```

1. Даны действительные числа s,t. Получить g(15,-s) + g(t,s^2+25) - g(21s-1,st-t^3)
   где g(a,b) = $\frac{a^2+b^2}{a^2+3ab-3a^2b+8}$
2. Даны действительные s,t. Получить f(t,-2s,1.7) + f(2.2,t,s-t), где f(a,b,c) = (2a - b - sin(c))/(5 + |c|)

3. Даны две последовательности целых чисел. Определить произведение элементов, какой последовательности больше и на сколько?

4. Написать функцию, которая вычисляет объём цилиндра. Параметрами функции должны быть радиус и высота цилиндра.

5. Написать функцию, которая сравнивает два целых числа и возвращает результат сравнения в виде одного из знаков > , < или =.

6. Написать функцию, которая вычисляет ab . Числа a и b могут быть дробными положительными числами.

7. Написать функцию Dohod, которая вычисляет доход по вкладу. Исходными данными для функции являются: величина вклада, процентная ставка (годовых) и срок вклада (количество дней)

Программирование задач с рекурсией

Цель работы: Научиться работать с использованием рекурсивных алгоритмов
Исходные данные (задание):

Задание 1 Написать в среде Python программы, реализующие следующие алгоритмы, запустить их на выполнение

Рассмотрим функцию вычисления факториала n!

n! = 1*2*3*...*n

Такое произведение можно легко вычислить с помощью итеративных конструкций (итерация - это повторение), например, оператора цикла for:

```
def fact(n):
    f = 1
    for i in range(1, n + 1):
        f= f*i
    return f
print(fact(int(input())))
```

Однако существует также другое определение факториала, в котором n! выражается через предыдущий (n-1)!, т.е. используется рекуррентная формула:

0! = 1

для любого n>0 n! = n*(n-1)!

Наличие рекуррентного соотношения позволяет использовать рекурсию. Например, программа, использующая рекурсивную функцию для вычисления факториала n! имеет следующий вид:

```
def rec(n):
    if n == 1: # граничный случай
        return 1
    else: # рекурсивный случай
        return n * rec(n - 1)
print(rec(int(input())))
```

Задание 2 Написать в среде Python программы, реализующие следующие алгоритмы, запустить их на выполнение

1. Написать рекурсивную программу вычисления максимального числа Фибоначи, ближайшего к заданному n по недостатку.
2. Написать рекурсивную программу поиска индекса минимального элемента массива
3. Написать рекурсивную программу поиска минимального элемента массива

Практическая работа №12

Программирование личных библиотек пользователя

Цель работы: Научиться создавать личные библиотеки и использовать их в работе.

Практическая работа №13

Работа с файлами

Цель работы: Научиться работать с файлами данных

Задача 1. Разработайте программу, которая позволит записать сведения об автомобилях в текстовый файл.
В результате выполнения оператора zap=open("avto.txt", "w+", encoding='utf-8') мы создаем в текущем каталоге файл с именем avto.txt и открываем его в режиме w+, что позволит при отладке программы многократно создавать и перезаписывать файл. Организация цикла for i in range(1,3) позволит вводить данные о двух автомобилях, указывая их марку, год выпуска и цвет. При этом мы работаем с переменной zap, поскольку именно ей присвоили результат работы функции open. Закончив выполнение операторов внутри цикла, закрываем файл оператором zap.close().

```
zap=open("avto.txt", "w+", encoding='utf-8') #Открываем файл на запись
for i in range(1,3):
    marka=input("\nВведите марку автомобиля   ")
    zap.write(marka)
    zap.write("\n")  #Перевод курсора на следующую строку
    god=input("\nВведите год выпуска    ")
    zap.write(god)
    zap.write("\n")
    color=input("\nВведите цвет машины     ")
    zap.write(color)
    zap.write("\n")
zap.close() #Закрываем файл
```

2. Чтение информации из текстового файла
Задача 2. Разработайте программу, которая позволит прочитать информацию об автомобилях, содержащуюся в текстовом файле.
В операторе open("avto.txt", "r", encoding='utf-8') изменим только режим доступа. При записи в файл использовался режим "w+", сейчас "r", который, используется при чтении информации из файла.
Метод read() с пустыми круглыми скобками используется по отношению к файловой переменной chten для чтения всей информации из файла. Далее методом close() мы закрываем файл.

```
chten=open("avto.txt", "r", encoding='utf-8') #Открываем файл на чтение
print(chten.read())
chten.close() #Закрываем файл
```

При работе с файлами возможно возникновение ошибок, связанных с обработкой пути к файлу. Такой тип ошибок можно перехватить, используя обработку исключений. Если предположить, что ошибочно задан путь к файлу avto.txt, то программа выдаст сообщение "Ошибка при открытии файла".

```
try:
   chten=open("c:\\Primer\\avto.txt", "r", encoding='utf-8')
except:
   print("Ошибка при открытии файла")
else:
   print(chten.read())
```

chten.close()

Если в скобках метода read() указать значение, например десять, то с помощью такой записи можно будет прочитать десять символов строки.

Если нужно прочитать некий текст, содержащийся в файле, построчно, то тогда следует использовать метод, отличие которого от метода read() заключается в том, что читать символы можно только в текущей строке. Обратите внимание: метод read()прочтет содержимое всего файла, между тем, как метод readline() прочтет только первую строку

ЗАПИСЬ ИНФОРМАЦИИ В ДВОИЧНЫЙ ФАЙЛ

Двоичные файлы или бинарные файлы, представляют собой набор байтов, а не набор символов, которые содержатся в текстовых файлах. Бинарные файлы хранят информацию в том виде, в котором она представлена в памяти компьютера, поэтому, открыв такой файл с помощью текстового редактора, мы получим набор нечитаемых символов.

В Python существует такое понятие, как консервация данных. Она позволяет хранить в файлах не просто набор символов, а более сложные структуры, например, списки или словари. Такое представление данных позволяет получить представление некоторого объекта в виде набора байтов, причем, сохраненный и переданный на другой компьютер, такой набор может быть расконсервирован, т. е. восстановлен.

Для совершения всех этих операций, в Python предусмотрены два модуля:

модуль **pickle** (от англ. pickling – консервация) и модуль **shelve** (от англ. shelving – стеллаж, размещение на полках). Соответственно, первый метод позволяет консервировать структуры данных, а второй метод – осуществить доступ к объектам, хранящихся «на полках».

Для того чтобы записать информацию в бинарный файл, его требуется **открыть**

Файловая переменная=open (FileName, Mode)

Где:

Файловая переменная – переменная, в которой будет находиться значение, возвращенное функцией open;

FileName – представляет собой путь к открываемому файлу;

Mode – режим доступа. Может принимать значения:

"rb" Чтение информации из файла. Если файл не существует, возникает исключение IOError

"wb" Запись информации в файл. Если файл существует, его содержимое полностью заменится; если файл не существует, он будет создан.

"ab" Дозапись в двоичный (бинарный) файл. Если файл существует, новые данные будут дописаны в конец. Если файл не существует, он будет создан

"rb+" Чтение и запись в двоичный (бинарный) файл. Если файл не существует, возникает исключение IOError

"wb+" Чтение и запись из двоичного (бинарного) файла. Если файл существует, он будет перезаписан. Если файл не существует, он будет создан

"ab+" Дозапись и чтение из двоичного (бинарного) файла. Если файл существует, новые данные будут дописаны в конец. Если файл не существует, он будет создан

Запись осуществляется с помощью **метода dump(),** имеющего следующий синтаксис:

pickle.dump (данные, файловая_переменная)

Задача 3. Разработайте базу данных, содержащую сведения об автомобилях.

О каждом автомобиле известно следующее:

- марка автомобиля;
- цвет автомобиля;
- год выпуска автомобиля.

Требуется создать:

файл с законсервированной информацией обо всех автомобилях;

осуществить чтение и расконсервацию данных из файла;

создать новый файл, в котором организовать размещение списков на полках и осуществить вывод хранящейся информации по указанному пользователем автомобилю.

Комментарий. В самом начале программы мы создадим три списка avto1, avto2, avto3 в которых находятся сведения о каждом автомобиле: марка, цвет и год выпуска. Четвертый список avto4 создадим в диалоговом режиме. Далее, оператором zap=open("avto.dat","wb") открывается файл с именем avto.dat на запись двоичной информации, на что указывает режим "wb" (С помощью последующих четырех методов dump() происходит так называемая сериализация объектов (последовательность перевода какой-либо структуры данных в последовательность битов).

В следующей части программы происходит расконсервация данных. Она основана на применении функции load(), которая читает данные из файла и преобразовывает их в объект. Ее синтаксис следующий:

Имя_переменной=pickle.load(имя_файловой_переменной)

В один файл можно сохранить сразу несколько объектов. Далее оператором print выводим списки автомобилей на экран.

```
import pickle
avto1=["Лада Веста","Белый","2022"]
avto2=["Жигули", "Красный", "1980"]
avto3=["Победа","Зеленый", "1950"]
marka=input("\nВведите марку автомобиля   ")
color=input("\nВведите цвет автомобиля   ")
god=input("\nВведите год автомобиля   ")
avto4=[marka,color,god]
zap=open("avto.dat","wb")
pickle.dump(avto1,zap) #консервация данных
pickle.dump(avto2,zap)
pickle.dump(avto3,zap)
pickle.dump(avto4,zap)
zap.close()
#--------------------------------
zap=open("avto.dat","rb+")
avto1=pickle.load(zap)  #расконсервация данных
avto2=pickle.load(zap)
avto3=pickle.load(zap)
avto4=pickle.load(zap)
print(avto1)
print(avto2)
print(avto3)
print(avto4)
zap.close()
```

Создадим новый проект Python и, подключив модуль shelve, откроем файл avto1.dat с помощью функции open. Использование оператора shelve.open предполагает наличие файла с консервированными объектами.

Теперь мы можем вывести данные по каждому автомобилю, запросив, например, его название. Полка (polka) подобна словарю, и сведения, хранящиеся на полке, доступны по соответствующему ключу: "avto1", "avto2", "avto3", "avto4".

polka["avto1"], polka["avto2"] polka["avto3"], polka["avto4"].

Как в словаре, каждому значению, хранящемуся на полке, соответствует ключ, который записывается в квадратных скобках: "avto1", "avto2", "avto3", "avto4".

С помощью **метода sync()** объекта-полки, осуществляется запись данных.

```
import shelve
polka=shelve.open("avto1.dat")
polka["avto1"]=["Лада","Белый","2020"]
polka["avto2"]=["Жигули", "Красный", "1980"]
polka["avto3"]=["Победа","Зеленый", "1950"]
marka=input("\nВведите марку автомобиля   ")
color=input("\nВведите цвет автомобиля   ")
god=input("\nВведите год автомобиля   ")
polka["avto4"]=[marka,color,god]
polka.sync()
#----------------------------------
marka=input("\nВведите марку автомобиля   ")
if marka=="Лада":
    print("Данные по машине",marka, polka["avto1"])
if marka=="Жигули":
    print("Данные по машине",marka,polka["avto2"])
if marka=="Победа":
    print("Данные по машине",marka,polka["avto3"])
if marka=="Тойота":
    print("Данные по машине",marka,polka["avto4"])
polka.close()
```

Задача 3. Разработайте программу, формирующую кортеж из чисел, которые вводит пользователь с клавиатуры. Данные кортежа консервируются и записываются в двоичный файл. Во второй программе осуществите чтение кортежа из файла, создайте функцию, которая осуществляет нахождение минимального из положительных чисел. Результат выведите в текстовый файл.

Ввод элементов кортежа:

```
import pickle
sp=[]  #Объявление кортежа
n=int(input("\nВведите количество будущих элементов кортежа  "))
for i in range(n):
    chislo=int(input("\nВведите число   "))
    sp.append(chislo)
#-------------------------------------------
zap=open("korteg.dat","wb")
pickle.dump(sp,zap)
zap.close()
```

извлечение кортежа из файла, нахождение минимального из положительных чисел и запись результата в текстовый файл.

```
import pickle
zap=open("korteg.dat","rb+")
chislo=pickle.load(zap)
print(chislo)
zap.close()
#-------------------------------------------
def F_min(*arg): #Функция осуществляет поиск минимального числа
    min = 32767
    for i in arg:
        if i>0: #Проверка на положительность
            if i<min:    #Поиск минимального элемента
                min=i
    return min
min_=F_min(*chislo) #Вызов функции
print("Минимальное число из положительных равно  ", min_)
minim=str(min_) #Преобразование полученного результата в строковое
#значение
f=open("rezult.txt", "w+", encoding='utf-8') #Открываем файл на запись
f.write(minim)
f.close()
```

Контрольные вопросы

1. Перечислите этапы работы с любым файлом с точки зрения программирования?
2. Напишите синтаксис функции open(), предназначенной для открытия файла.
3. Напишите синтаксис функции write(), предназначенной для записи информации в файл.
4. Напишите синтаксис функции close(), предназначенной для закрытия файла.
5. Объясните, каким образом происходит обработка ошибок, возникающих при работе с файлами.
6. В чем особенность метода readline().
7. Дайте характеристику бинарных файлов.
8. Какова цель консервации данных, используемой в языке Python?
9. Опишите назначение модулей pickle и shelving.
10. Какая инструкция используется для записи информации в бинарный файл?
11. Какие возможные значения режима Mode функции open(), используемой при работе с бинарными файлами, вы знаете?

12. Каким образом осуществляется запись информации в бинарный файл?

Задачи для самостоятельного решения

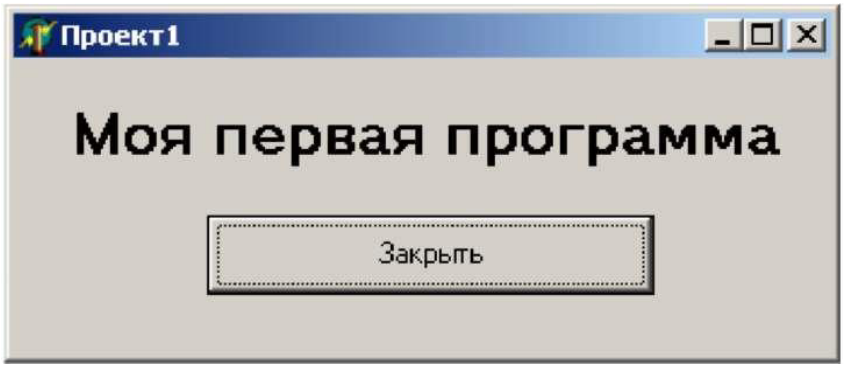
1. Сформируйте файл вещественных чисел. Найдите максимальный и минимальный элементы, имеющие четные индексы. Выведите результаты выполнения программы на экран
2. Сформируйте файл вещественных чисел. Отсортируйте их по возрастанию. Выведите результаты выполнения программы на экран
3. Создайте файл, в котором будут храниться коэффициенты квадратного уравнения. Найдите корни квадратного уравнения и выведите на экран соответствующее сообщение

Практическая работа №14

Создание приложения с графическим интерфейсом

Задание 1 Цель работы - создать программу, выполняющую следующие действия:

1. После запуска программы появляется окно.



2. Для выхода из программы необходимо щелкнуть мышью на кнопке "Закрыть".

Описание плана разработки программы

1. Открыть новый проект.

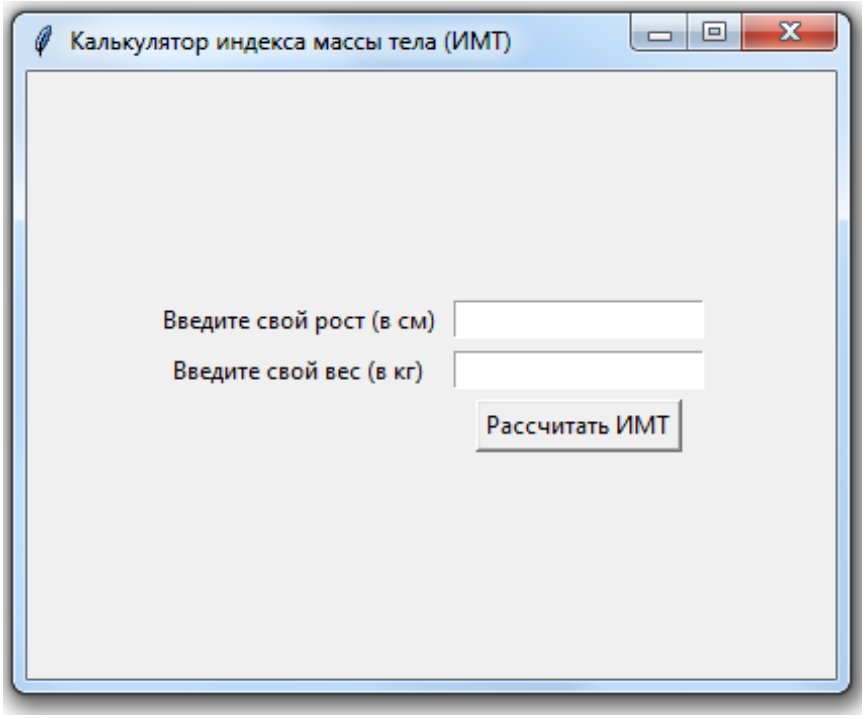
2. Разместить на форме экземпляры компонентов: метку и кнопку.

4. Сохраните проект, запустите и протестируйте его.

Задание 2. Напишите калькулятор индекса массы тела. ИМТ — это важный медицинский показатель, который позволяет оценить, есть ли у человека избыточный вес или ожирение. Он рассчитывается по следующей формуле:

ИМТ = вес (в кг) / рост^2 (в метрах)

Вид приложения:



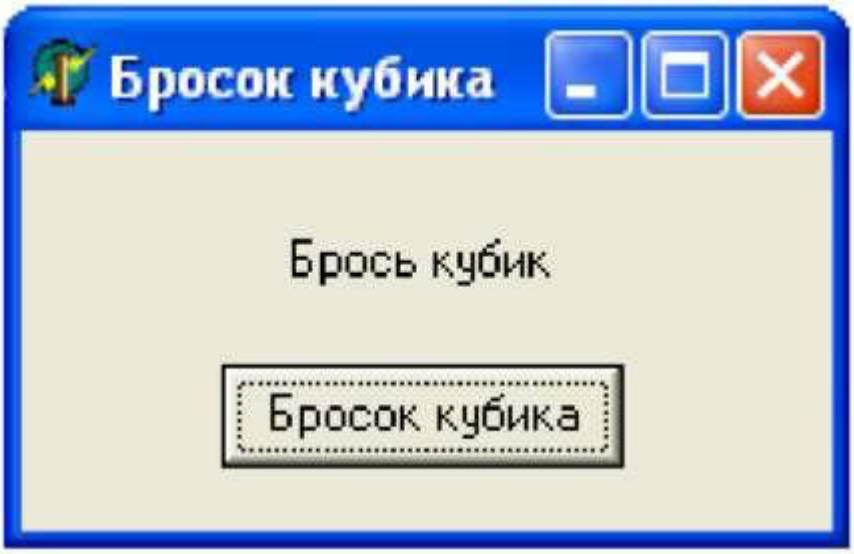
```
import tkinter as tk

from tkinter import *

from tkinter import messagebox

window = Tk() #Создаём окно приложения.

window.title("Калькулятор индекса массы тела (ИМТ)") #Добавляем название приложения.

window.geometry('400x300')

frame = Frame(

  window, #Обязательный параметр, который указывает окно для размещения Frame.

  padx = 10, #Задаём отступ по горизонтали.

  pady = 10 #Задаём отступ по вертикали.

)

frame.pack(expand=True) #Не забываем позиционировать виджет в окне. Здесь используется метод pack. С помощью свойства expand=True указываем,

#что Frame заполняет весь контейнер, созданный для него.

height_lb = Label(

  frame,

  text="Введите свой рост (в см)  "

)

height_lb.grid(row=3, column=1)

weight_lb = Label(

  frame,

  text="Введите свой вес (в кг)  ",

)

weight_lb.grid(row=4, column=1)

height_tf = Entry(

  frame, #Используем нашу заготовку с настроенными отступами.

)

height_tf.grid(row=3, column=2)

weight_tf = Entry(
```

```
    frame,
)
weight_tf.grid(row=4, column=2, pady=5)
cal_btn = Button(
    frame, #Заготовка с настроенными отступами.
    text='Рассчитать ИМТ', #Надпись на кнопке.
)
cal_btn.grid(row=5, column=2) #Размещаем кнопку в ячейке, расположенной ниже, чем наши надписи, но во втором столбце, то есть под ячейками для ввода информации.
def calculate_bmi(): #Объявляем функцию.
    kg = int(weight_tf.get()) #С помощью метода .get получаем из поля ввода с именем weight_tf значение веса, которое ввёл пользователь и конвертируем в целое число с помощью int().
    m = int(height_tf.get())/100 #С помощью метода .get получаем из поля ввода с именем height_tf значение роста и конвертируем в целое число с помощью int(). Обязательно делим его на 100, так как пользователь вводит рост в сантиметрах, а в формуле для расчёта ИМТ используются метры.
    bmi = kg/(m*m)#Рассчитываем значение индекса массы тела.
    bmi = round(bmi, 1) #Округляем результат до одного знака после запятой.
    if bmi < 18.5:
        messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует недостаточному весу')
    elif (bmi > 18.5) and (bmi < 24.9):
        messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует нормальному весу')
    elif (bmi > 24.9) and (bmi < 29.9):
        messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует избыточному весу')
    else:
        messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует ожирению')
cal_btn = Button(
    frame,
    text='Рассчитать ИМТ',
    command=calculate_bmi #Позволяет запустить событие с функцией при нажатии на кнопку.
)
cal_btn.grid(row=5, column=2)
```

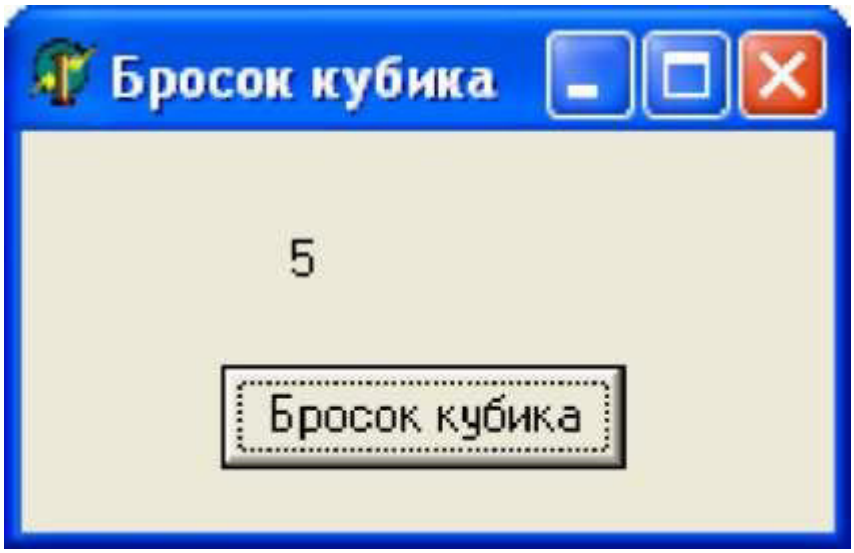
Практическая работа №15

Случайный выбор из списка

Цель работы - создать программу, выполняющую следующие действия: 1. После запуска программы появляется надпись "Брось кубик".



2. По щелчку мышью на кнопке "Бросок кубика" появляется сообщение, выдающее числа-очки в диапазоне 0 - 6.



.

3. Для выхода из программы необходимо щелкнуть мышью на закрывающей кнопке в строке заголовка.

Описание плана разработки программы

1. Открыть новый проект.

2. Разместить на форме экземпляры компонентов: метку Label и кнопку Button.

3. Выполнить следующие действия:

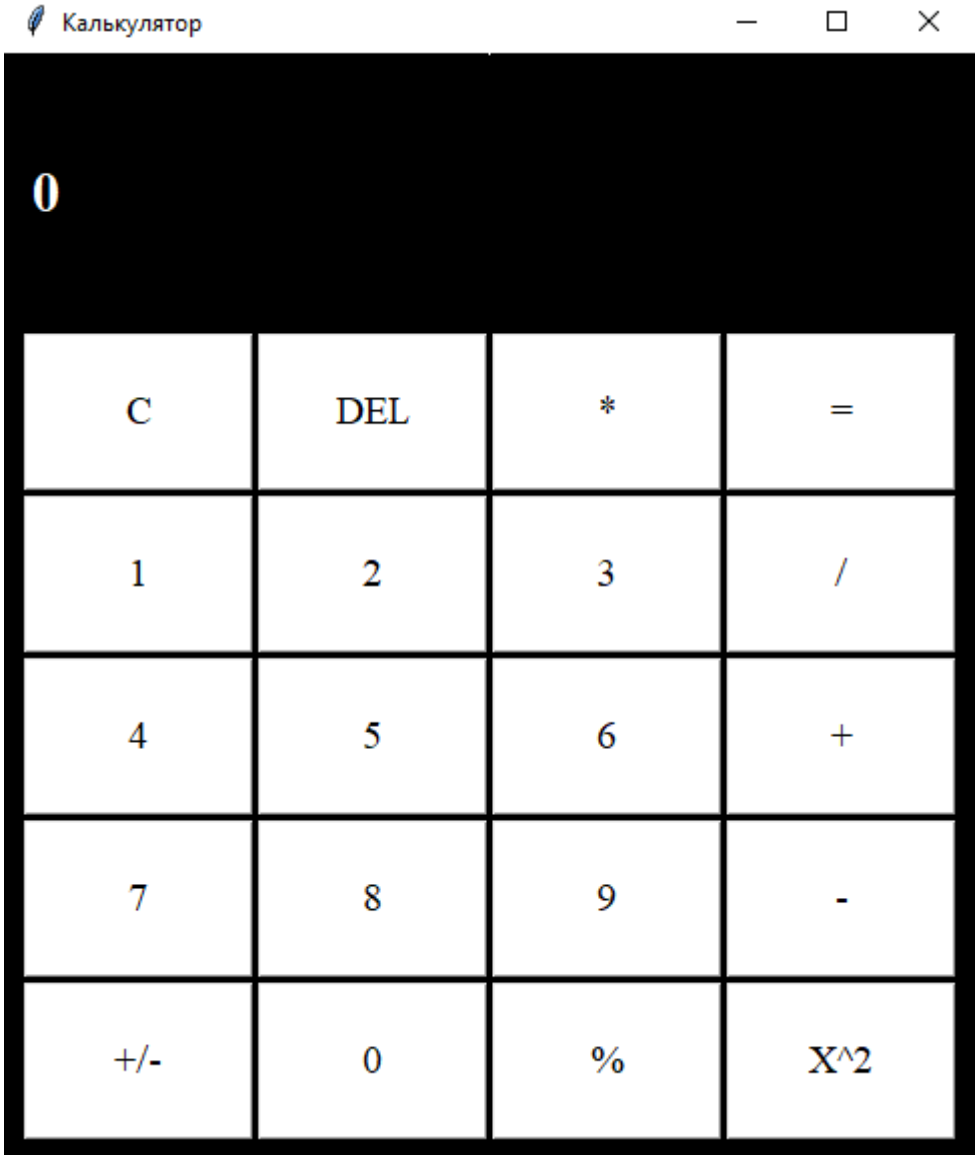
Выделенный объект	Действие
Forml	Установка имени формы "Бросок кубика"
Label1	Ввод текста надписи "Брось кубик"
Button!	Установка имени кнопки "Бросок кубика"

4. Сохраните проект, запустите и протестируйте его.

Практическая работа №16

Разработка приложения «Обычный калькулятор»

Цель работы - создать программу, выполняющую действия обычного калькулятора. Описание плана разработки программы



Создаём окно 485 на 550. Размеры не важны.. Так же указываем, что окно не будет изменяться.

```
from tkinter import *


class Main(Frame):
    def __init__(self, root):
        super(Main, self).__init__(root)
        self.build()

    def build(self):
        pass
```

```
    def logicalc(self, operation):
        pass

    def update():
       pass


if __name__ == '__main__':
    root = Tk()
    root["bg"] = "#000"
    root.geometry("485x550+200+200")
    root.title("Калькулятор")
    root.resizable(False, False)
    app = Main(root)
    app.pack()
    root.mainloop()
```

Делаем кнопки

В методе *build* создаём такой список:

```
btns = [
            "C", "DEL", "*", "=",
            "1", "2", "3", "/",
            "4", "5", "6", "+",
            "7", "8", "9", "-",
            "+/-", "0", "%", "X^2"
        ]
```

Отображаем кнопки в цикле. Для этого в том же методе пишем следующее:

```
x = 10
        y = 140
        for bt in btns:
            com = lambda x=bt: self.logicalc(x)
            Button(text=bt, bg="#FFF",
                   font=("Times New Roman", 15),
                   command=com).place(x=x, y=y,
                                      width=115,
                                      height=79)
            x += 117
            if x > 400:
                x = 10
```

```
                y += 81
```

Добавляем надпись с выводом результата.

```
self.formula = "0"
self.lbl = Label(text=self.formula, font=("Times New Roman", 21, "bold"),
                 bg="#000", foreground="#FFF")
self.lbl.place(x=11, y=50)
```

```
def logicalc(self, operation):
    if operation == "C":
        self.formula = ""
    elif operation == "DEL":
        self.formula = self.formula[0:-1]
    elif operation == "X^2":
        self.formula = str((eval(self.formula))**2)
    elif operation == "=":
        self.formula = str(eval(self.formula))
    else:
        if self.formula == "0":
            self.formula = ""
        self.formula += operation
    self.update()

def update(self):
    if self.formula == "":
        self.formula = "0"
    self.lbl.configure(text=self.formula)
```

Полный код версии калькулятора:

```
from tkinter import *
```

```
class Main(Frame):
    def __init__(self, root):
        super(Main, self).__init__(root)
        self.build()

    def build(self):
        self.formula = "0"
        self.lbl = Label(text=self.formula, font=("Times New Roman", 21, "bold"), bg="#000",
foreground="#FFF")
        self.lbl.place(x=11, y=50)

        btns = [
            "C", "DEL", "*", "=",
            "1", "2", "3", "/",
            "4", "5", "6", "+",
            "7", "8", "9", "-",
            "(", "0", ")", "X^2"
        ]

        x = 10
        y = 140
        for bt in btns:
            com = lambda x=bt: self.logicalc(x)
            Button(text=bt, bg="#FFF",
                   font=("Times New Roman", 15),
                   command=com).place(x=x, y=y,
                                      width=115,
                                      height=79)
            x += 117
            if x > 400:
                x = 10
                y += 81

    def logicalc(self, operation):
        if operation == "C":
            self.formula = ""
        elif operation == "DEL":
            self.formula = self.formula[0:-1]
        elif operation == "X^2":
            self.formula = str((eval(self.formula))**2)
        elif operation == "=":
            self.formula = str(eval(self.formula))
        else:
            if self.formula == "0":
                self.formula = ""
            self.formula += operation
        self.update()

    def update(self):
        if self.formula == "":
            self.formula = "0"
        self.lbl.configure(text=self.formula)


if __name__ == '__main__':
    root = Tk()
    root["bg"] = "#000"
    root.geometry("485x550+200+200")
    root.title("Калькулятор")
    root.resizable(False, False)
    app = Main(root)
    app.pack()
    root.mainloop()
```

Практическая работа №17

Нахождение минимального и максимального числа в массиве

Цель работы - создать программу, которая находит минимальное и максимальное числа в введенном массиве.

Описание плана разработки программы

1. Открыть новый проект.

2. Разместить на форме экземпляры компонентов: Button, Edit, Label.

3. Внешний вид программы