**Министерство науки и высшего образования РФ**

**федерального государственное бюджетное образование**

**учреждение высшего образования**

**Московский авиационный институт**

**(национальный технический университет)**

**МАИ**

Отчет по выполненным лабораторным работам № 1-4 (вариант 1)

по дисциплине “Программирование и основы алгоритмизации”

Выполнил студент гр. М3О-101С-21

Белых С.Е.

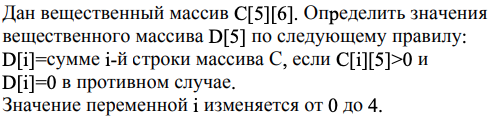
Принял: доц. каф. 301, канд. тех. наук Кананадзе С. С.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва 2022

**Лабораторная работа №0**

**Задание 1**:



**Описание программы:**

Создаем два массива: C[5][6] и D[5] и переменную s. Заполняем массив C случайными значениями. Проходимся по массивам массива C. Если последний элемент больше нуля, суммируем все элементы массива, проходясь по ним циклом и записываем сумму в переменную s. Если последний элемент равен нулю, в переменную s записываем 0. Затем добавляем значение переменной s в массив D, после чего обнуляем s. После всего этого выводим на экран массивы C и D.

**Текст программы:**

from random import randint

c = [[], [], [], [], []]

d = []

s = 0

def generateList(): # заполняет массив размерами [n][6], где n - длина массива

for i in range(0, len(c)):

for \_ in range(0,6):

x = randint(99, 1000)

if x == 99: x = 0

x /= 100

c[i].append(x)

def main():

global c, d, s;

generateList();

for i in c:

if i[5] > 0:

for j in i:

s += j

elif i[5] == 0:

s = 0

d.append(round(s, 3))

s = 0

print(f"C[5][6] = {c}\n\nD[5] = {d}")

if \_\_name\_\_ == "\_\_main\_\_":

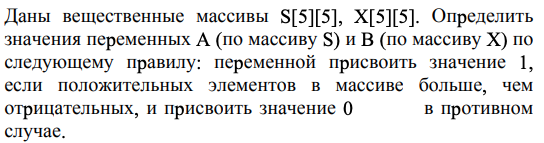
main();

**Результат работы программы:**

C[5][6] = [[7.99, 1.36, 7.08, 5.97, 4.03, 9.84], [7.02, 5.95, 8.61, 6.13, 8.64, 6.24], [1.08, 5.13, 5.73, 3.58, 4.53, 9.13], [1.6, 7.07, 6.11, 9.98, 7.06, 5.28], [2.45, 6.4, 8.22, 1.8, 8.95, 5.2]]

D[5] = [36.27, 42.59, 29.18, 37.1, 33.02]

**Задание 2:**

****

**Описание программы:**

Создаем 2 массива S[5][5] и X[5][5], две переменные a\_s и a\_x и две переменные-счетчика negative и positive. Заполняем массивы случайными числами. Проходимся по каждому элементу каждого из массивов и проверяем, больше или меньше данное число нуля и на основании этого добавляем в соответствующую переменную-счетчик единицу. Далее сравниваем значения счетчиков. Если положительных больше, переменной a\_s или a\_x присваиваем 1, иначе присваиваем 0. После чего выводим оба массива и a\_s, a\_x на экран.

**Текст программы:**

from random import randint

s = [[],[],[],[],[]]

x = [[],[],[],[],[]]

a\_s, a\_x = None, None;

def generateList(ls: list): # заполняет массив размерами [n][5], где n - длина передаваемого массива

for i in range(0, len(ls)):

for \_ in range(0,5):

x = randint(-999, 1000);

x /= 100;

ls[i].append(x);

return ls;

def checkList(ls: list): # сравнивает количество положительных и отрицательных чисел в массиве

positive = 0;

negative = 0;

for i in ls:

for j in i:

if j < 0: negative += 1;

else: positive += 1;

if positive > negative: a = 1;

else: a = 0;

return a;

def main():

global s, x, a\_s, a\_x;

s = generateList(s);

x = generateList(x);

a\_s = checkList(s);

a\_x = checkList(x);

print(f"S[5][5] = {s}\n\nX[5][5] = {x}\n\na\_s = {a\_s}\n\na\_x = {a\_x}")

if \_\_name\_\_ == "\_\_main\_\_":

main();

**Результат работы программы:**

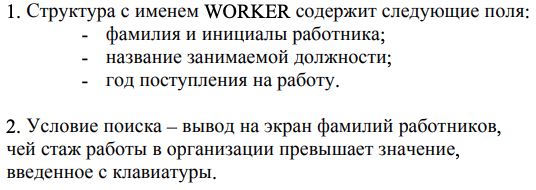
S[5][5] = [[-6.33, -5.38, 6.4, 7.87, -4.96], [0.82, 3.6, 7.27, 0.53, -2.5], [5.53, -5.89, -0.28, -4.41, 3.59], [-6.99, -3.41, 1.42, -9.44, 4.12], [-2.02, 3.72, 5.77, -4.11, 2.76]]

X[5][5] = [[0.13, 0.76, 6.25, -3.94, -7.65], [3.73, 6.0, 7.25, -7.22, 4.99], [-6.48, -4.92, 3.56, -7.4, -8.87], [-5.43, 6.48, -1.87, -7.1, -7.76], [-8.56, -4.06, -9.93, -3.72, 7.75]]

a\_s = 1

a\_x = 0

**Задание 3:**



**Описание программы:**

Создаем класс WORKER с параметрами fullName, post, year и массив, куда будут помещены все элементы данного класса. Затем просим пользователя 10 раз ввести ФИО, должность и год начала работы, после чего, создавая экземпляры класса WORKER, добавляем их в массив. Просим пользователя ввести стаж работы. Вычитаем из настоящего года стаж работы, получая год устройства на работу. После чего, проходимся по всем элементам массива и проверяем, чей год устройства меньше искомого. Если находим такого работника, выводим его ФИО на экран.

**Текст программы:**

import datetime

class WORKER:

def \_\_init\_\_(self, SN\_INITs: str, post: str, YAW: int) -> None:

self.fullName = SN\_INITs;

self.post = post;

self.year = YAW;

return;

ls = []

def atl():

global ls;

for \_ in range(0,10):

ls.append(WORKER(input("Введите Фамилию ИО: "), input("Введите занимаемую должность: "), int(input("Введите год начала работы: "))))

def main():

global ls;

atl();

print(ls);

year = datetime.date.today().year - int(input("Введите стаж работы: "));

for i in ls:

if year > i.year: print(i.fullName)

if \_\_name\_\_ == "\_\_main\_\_":

main();

**Результат работы программы:**

Введите Фамилию ИО: ываыа ывпып

Введите занимаемую должность: выпвыр

Введите год начала работы: 1234

Введите Фамилию ИО: ыаипт

Введите занимаемую должность: вапрть

Введите год начала работы: 1234

Введите Фамилию ИО: апрь м

Введите занимаемую должность: лрпапи

Введите год начала работы: 9643

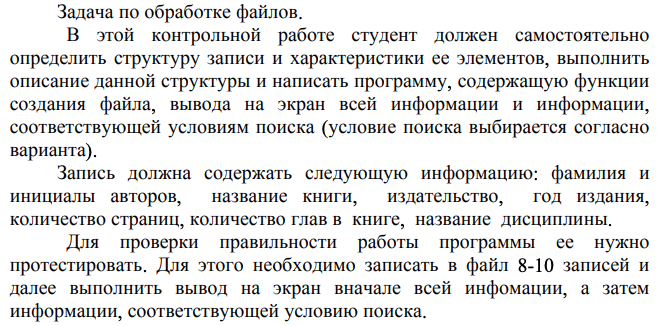
[<\_\_main\_\_.WORKER object at 0x104ecbf10>, <\_\_main\_\_.WORKER object at 0x104ecbd30>, <\_\_main\_\_.WORKER object at 0x104ecbd90>]

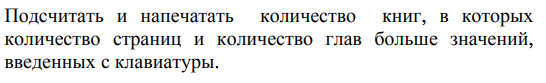
Введите стаж работы: 10

ываыа ывпып

ыаипт

**Задание 4:**





**Описание программы:**

**Текст программы:**

import json

class BOOK:

def \_\_init\_\_(self, SN\_INITs: str, name: str, publisher: str, year: int,

pagesCount: int, chaptersCount: int, topic: str):

self.name = name;

self.topic = topic;

self.authorName = SN\_INITs;

self.publisher = publisher;

self.year = year;

self.pages = pagesCount;

self.chapters = chaptersCount;

def readFile():

with open("Лаба 0/files/test.json", encoding="utf-8") as doc:

doc = json.load(doc);

return doc;

def writeFile(toWrite):

with open('Лаба 0/files/test.json', 'w', encoding="utf-8") as outfile:

json.dump(toWrite, outfile, ensure\_ascii=False, indent=4)

def generateFile():

bks = [];

bks.append(BOOK("ldn", "svr", "lkvnfv", 1932, 144, 5, "lrnf"));

bks.append(BOOK("kfj", "ewf", "sdv", 2007, 561, 1, "sklnc"));

bks.append(BOOK("jkbsvk", "aef", "sdvsdv", 2001, 311, 2, "kjsdnf"));

bks.append(BOOK("ksdjbvf", "sdm v", "sdvm", 1816, 680, 1, "lrnf"));

bks.append(BOOK("kjsndv", "sd dcmk", "sdvsv", 1986, 421, 3, "eklfnj"));

bks.append(BOOK("ksjdbvn", "ssdklc", "sdklvn", 1998, 767, 1, "kjsdnf"));

bks.append(BOOK("ksjdv", "lkvns", "wqof", 1912, 30, 1, "kenf"));

bks.append(BOOK("ksdv", "kldms", "lkvnfv", 2015, 1024, 2, "sleknf"));

doc = [];

for i in bks:

doc.append(eval(json.dumps(i.\_\_dict\_\_)))

writeFile(doc);

def printAllFile():

file = readFile();

for i in file:

print(i);

def findBooks(file, pages: int, chapters: int) -> list:

b = [];

for i in file:

if i["pages"] >= pages and i["chapters"] >= chapters:

b.append(i);

return b;

def printFoundBooks(list: list):

message = f"Найдено {len(list)} подходящих книг, вот их список:";

for i in list:

message += f"\n{i}";

print(message);

def main():

printAllFile();

printFoundBooks(findBooks(readFile(), int(input("Введите минимальное число страниц для поиска: ")),

int(input("Введите минимальное число глав для поиска: "))));

if \_\_name\_\_ == "\_\_main\_\_":

main();

**Результат работы программы:**

{'name': 'svr', 'topic': 'lrnf', 'authorName': 'ldn', 'publisher': 'lkvnfv', 'year': 1932, 'pages': 144, 'chapters': 5}

{'name': 'ewf', 'topic': 'sklnc', 'authorName': 'kfj', 'publisher': 'sdv', 'year': 2007, 'pages': 561, 'chapters': 1}

{'name': 'aef', 'topic': 'kjsdnf', 'authorName': 'jkbsvk', 'publisher': 'sdvsdv', 'year': 2001, 'pages': 311, 'chapters': 2}

{'name': 'sdm v', 'topic': 'lrnf', 'authorName': 'ksdjbvf', 'publisher': 'sdvm', 'year': 1816, 'pages': 680, 'chapters': 1}

{'name': 'sd dcmk', 'topic': 'eklfnj', 'authorName': 'kjsndv', 'publisher': 'sdvsv', 'year': 1986, 'pages': 421, 'chapters': 3}

{'name': 'ssdklc', 'topic': 'kjsdnf', 'authorName': 'ksjdbvn', 'publisher': 'sdklvn', 'year': 1998, 'pages': 767, 'chapters': 1}

{'name': 'lkvns', 'topic': 'kenf', 'authorName': 'ksjdv', 'publisher': 'wqof,', 'year': 1912, 'pages': 30, 'chapters': 1}

{'name': 'kldms', 'topic': 'sleknf', 'authorName': 'ksdv', 'publisher': 'lkvnfv', 'year': 2015, 'pages': 1024, 'chapters': 2}

Введите минимальное число страниц для поиска: 10

Введите минимальное число глав для поиска: 2

Найдено 4 подходящих книг, вот их список:

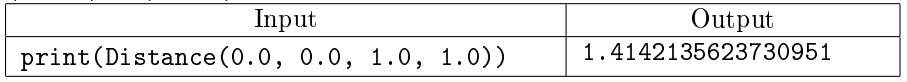
{'name': 'svr', 'topic': 'lrnf', 'authorName': 'ldn', 'publisher': 'lkvnfv', 'year': 1932, 'pages': 144, 'chapters': 5}

{'name': 'aef', 'topic': 'kjsdnf', 'authorName': 'jkbsvk', 'publisher': 'sdvsdv', 'year': 2001, 'pages': 311, 'chapters': 2}

{'name': 'sd dcmk', 'topic': 'eklfnj', 'authorName': 'kjsndv', 'publisher': 'sdvsv', 'year': 1986, 'pages': 421, 'chapters': 3}

{'name': 'kldms', 'topic': 'sleknf', 'authorName': 'ksdv', 'publisher': 'lkvnfv', 'year': 2015, 'pages': 1024, 'chapters': 2}

**Лабораторная работа №1**: Даны четыре вещественных числа: 𝑥1, 𝑦1, 𝑥2, 𝑦2. Напишите функцию Distance(x1, y1, x2, y2), принимающую на вход четыре вещественных числа и возвращающую расстояние между точкой (𝑥1, 𝑦1) и (𝑥2, 𝑦2).



**Описание программы:**

Расстояние между двумя точками вычисляется по формуле:



Создадим функцию, которая принимает 4 аргумента и возвращает расстояние между точками.

**Текст программы:**

from math import sqrt;

def distance(x1, y1, x2, y2):

return sqrt((x1 - x2)\*\*2 + (y1 - y2)\*\*2);

def main():

print(distance(0.0, 0.0, 1.0, 1.0))

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Результат работы программы:**

1.4142135623730951

**Лабораторная работа №2:** Найти максимальное число, встречающееся в за­данном векторе более одного раза.

**Описание программы:**

Генерируем массив чисел. При помощи модуля Counter считаем количество повторений каждого числа в заданном массиве. В новый массив добавляем только числа, число повторений которых в изначальном массиве больше единицы. При помощи функции max() вычисляем наибольшее число в новом массиве. Выводим на экран наибольшее число и изначальный массив.

**Текст программы:**

from collections import Counter

from random import randint

def generateList(len):

ls = []

for \_ in range(0, len):

ls.append(randint(-15, 15))

return ls

def countValues(list):

counter = dict(Counter(list))

return counter

def noName(dict):

list = []

for i in dict.keys():

if dict[i] > 1:

list.append(i)

return list

def main():

ls = generateList(15)

list = noName(countValues(ls))

if list == []: print(f"Вектор: {ls}\nНет повторяющихся значений")

else: print(f"Вектор: {ls}\n{max(ls)}")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Результат работы программы:**

Вектор: [-11, -3, -15, 0, 9, 9, 11, 14, 15, 0, -9, 14, -4, -6, -5]

15

**Лабораторная работа №3:** В файле операционной системы "Task4.in'' хранит­ся в текстовой форме ведомость сдачи экзаменов студентами неко­торой группы. Каждая строка этого файла содержит сведения об од­ном студенте, представленные в следующем формате: позиции 1…2 - порядковый номер студента в группе; позиция 3 - пробельная лите­ра; позиции 4...22 - фамилия студента длиной не более 18 символов, в произвольном месте поля; позиция 23 - пробельная литера; пози­ция 24. - четыре оценки по четырем предметам, разделенные не ме­нее чем одной пробельной литерой. Количество студентов в группе равно 16. Пример строк указанного файла:

  
Написать: 1) определение массива структур для хранения ука­занной ведомости; 2) фрагмент программы, который заполнит экза­менационную ведомость данными, вводимыми из файла операцион­ной системы "Task4.in" (ввод данных должен осуществляться в тек­стовом режиме; 3) фрагмент программы, который вычисляет сред­нюю экзаменационную оценку по всем предметам и студентам (т.е. среднюю оценку из 64 оценок), а затем выводит значение этого по­казателя в файл операционной системы "Task4.out".

**Описание программы:**

Создаем класс Student с параметрами number – номер в списке, lastName – фамилия студента и marks – массив оценок данного студента. Читаем файл Task4.in и передаем содержимое в виде массива, разбив текст файла по строкам, в переменную f. Создаем массив студентов, в который помещаем экземпляры класса Student, обходя каждый элемент (строку) переменной f. Зная, с какого символа в каждой строке лежит интересующая нас информация, “обрезаем” строку, оставляя только нужные данные. В поле с фамилией заменяем все пробелы пустой строкой, тем самым удаляя лишние пробелы. Созданный массив со студентами обходим циклом for, извлекая из данных каждого студента их оценки, а затем, посчитав их количество, находим среднее арифметическое, после чего записываем данное значение в файл Task4.out

**Текст программы:**

class Student:

def \_\_init\_\_(self, number:int, lastName: str, marks: list) -> None:

self.number = number

self.lastName = lastName

self.marks = marks

def readFile(path):

file = open(path, "r")

f = file.read()

file.close()

f = f.split("\n")

return f

def generateDict(file):

students = []

for i in file:

students.append(Student(i[:2], i[3:22].replace(" ", ""), [int(i[23:24]), int(i[25:26]), int(i[27:28]), int(i[29:30])]))

return students

def calcMidValue(dict):

midMark = 0

n = 0

for i in dict:

for j in i.marks:

n += 1

midMark += j

return midMark / n

def writeMidMark(midMark, path):

file = open(path, "w")

file.write(str(midMark))

file.close()

def main():

writeMidMark(calcMidValue(generateDict(readFile("Лаба 3/files/Task4.in"))), "Лаба 3/files/Task4.out")

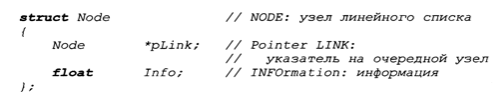
if \_\_name\_\_ == "\_\_main\_\_":

main()

**Результат работы программы:**

files/Task4.out: 4.666666666666667

**Лабораторная работа №4:** Определен следующий структурный тип:



В текстовом файле операционной системы ''Test8.in'' содер­жится некоторое количество вещественных чисел, разделенных символами пробельной группы ( ' ', '\t', '\n' ).

Написать прототип, определение и пример вызова функции, кото­рая должна ввести из файла "TestS.in" содержащиеся в нем веществен­ные числа и запомнить их в узлах линейного списка, в котором каждый узел (динамически размещенная в памяти структура) имеет тип Node. При этом первое прочитанное число должно находиться в последнем от начала узле линейного списка, второе число - в предпоследнем узле и т.д.

Все исходные данные (указатель на \*\*имя. расширение\*\* файла ввода) и все результаты работы функции (указатель на начало линейного списка) должны передаваться через список параметров. С целью обработки ошибок предусмотреть кон­троль значений, возвращаемых функциями библиотеки Си "fopen", "fscanf" и операцией new. Подключить необходимые стандартные заголовочные файлы.

**Описание программы:**

Открываем файл files/Task8.in для чтения. Читаем файл и записываем в переменную f содержимое файла. Затем закрываем файл. Создаем массив, в который добавляем значения из файла, предварительно заменив "\t" и "\n" на " " и разделив этот текст по " " при помощи метода .split(), проверяя, не пустое ли значение добавляется в массив. После чего выводим на экран данный массив.

**Текст программы:**

file = open("Лаба 4/files/Task8.in", "r")

f = file.read()

file.close()

f = [v for v in f.replace("\t", " ").replace("\n", " ").split(" ") if v]

print(f)

**Результат работы программы:**

['0.13811028740099907', '0.1900648902056189', '0.8179155541718748', '0.04130872321423651', '0.5082415401618877', '0.9214289338537682']