МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
Кафедра «Измерительно-вычислительные комплексы»

«Методы искусственного интеллекта»

Отчёт по лабораторной работе №3

Вариант №6

Выполнил:
студент группы ИСТбд-41
Евтушенко Александр
Проверил:
доцент кафедры ИВК, к.т.н.
Шишкин В.В.

Ульяновск
2022

Задание 1.
“Генерация данных”.

Создать симулированный набор данных и записать его на диск в виде csv файла со следующими параметрами:
- количество строк не менее 1000 (задается случайным образом);
- структура набора:
    - табельный номер;
    - Фамилия И.О.;
    - пол;
    - год рождения;
    - год начала работы в компании;
    - подразделение;
    - должность;
    - оклад;
    - количество выполненных проектов

Результат.

```
Процесс генерации

number , full_name , gender , birth_date , start_date , division , position , salary , completed_projects ,
1 , Маврина Р.К. , Жен , 2.1.1995 , 6.12.2013 , Отдел поддержки ИТ , Специалист , 26924 , 38 ,
2 , Райкин С.К. , Муж , 8.9.1987 , 19.12.2016 , Отдел контроля качества и процесов , Специалист , 100506 , 24 ,
3 , Щукин Г.К. , Муж , 26.12.1994 , 20.6.2022 , Отдел разработки ПО , Middle разработчик , 68244 , 7 ,
4 , Павлов Ж.Б. , Муж , 27.10.1994 , 9.8.2014 , Отдел поддержки ИТ , Специалист , 27356 , 18 ,
5 , Павлова У.З. , Жен , 8.5.1990 , 10.1.2016 , Отдел развития ИТ , Руководитель отдела развития ИТ , 76308 , 51 ,
6 , Кабаков Г.М. , Муж , 14.12.1993 , 26.1.2012 , Отдел разработки ПО , Middle разработчик , 50172 , 2 ,
7 , Данилина Г.И. , Жен , 27.11.1986 , 9.8.2013 , Отдел информационной безопасности , Начинающий специалист , 55460 , 17 ,
8 , Хабалова Э.Л. , Жен , 17.9.1996 , 9.11.2017 , Отдел поддержки ИТ , Специалист , 28422 , 22 ,
9 , Царёв Б.К. , Муж , 2.12.1989 , 25.2.2012 , Отдел поддержки ИТ , Руководитель отдела поддержки ИТ , 32868 , 45 ,
```

Задание 2.
“Получение статистических характеристик при помощи библиотеки numpy”.

Прочитать сгенерированный набор данных в виде списков и получить с помощью программирования и методов библиотеки numpy для разных по типу признаков столбцов (не менее 3) основные статистические характеристики (например для порядкового типа: минимум, максимум, среднее, дисперсия, стандартное отклонение, медиана, мода).

Результат.

```
Статистические данные

Для столбца номер: min=1 ; max=1499 ; ave=750.0 ; disp=187250.0 ; std=432.7239304683761 ; median=750.0 ; mode=1
Для столбца зарплата: min=10047 ; max=299190 ; ave=89567.54169446297 ; disp=4218172655.3289824 ; std=64947.46073041642 ; median=71196.0 ; mode=30472
Для столбца проекты: min=1 ; max=60 ; ave=21.211474316210808 ; disp=237.95327727936035 ; std=15.425734254140396 ; median=18.0 ; mode=18

CSV файл проанализирован (Нажмите Enter)
```

Задание 3.
“Получение статистических характеристик при помощи библиотеки pandas”.

Прочитать сгенерированный набор данных в виде датафрейма и получить с помощью методов библиотеки pandas для тех же столбцов те же статистические характеристики. Продемонстрировать применение не менее 3 методов библиотеки pandas.

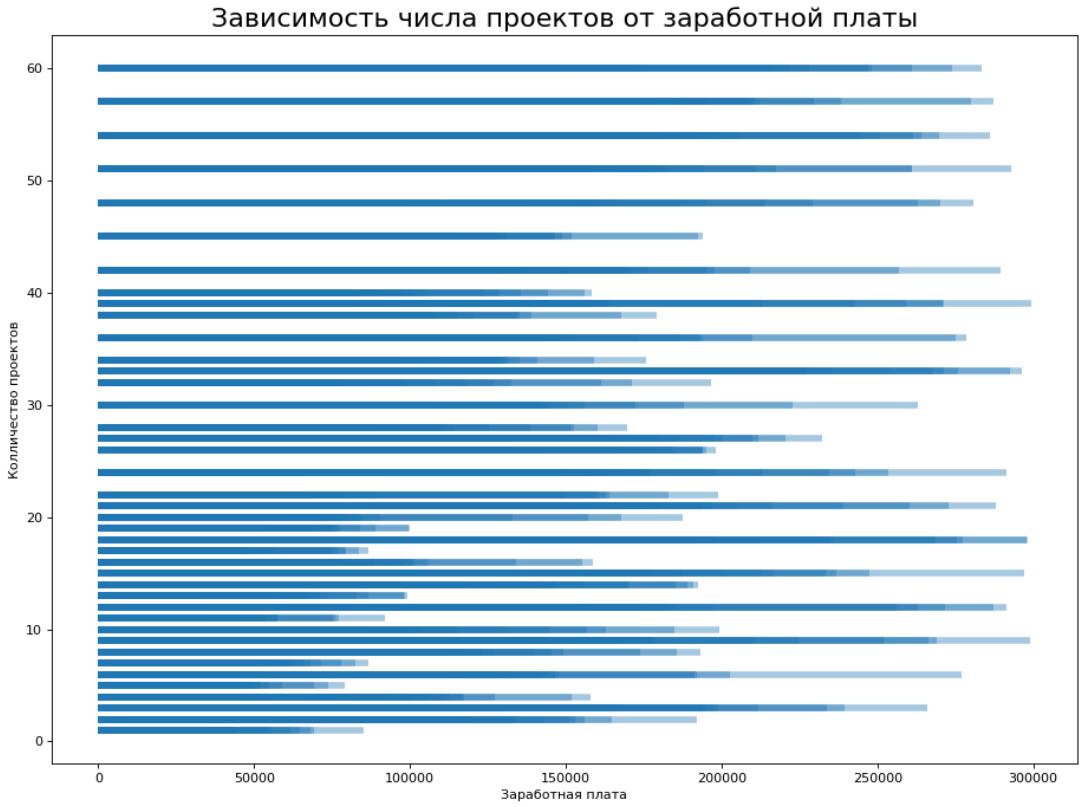
Результат.

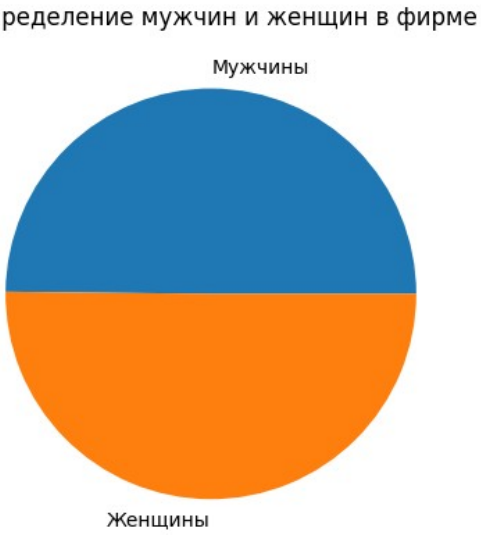
```
Статистические данные

Для столбца номер: min=1 ; max=1499 ; ave=750.0 ; disp=187375.0 ; std=432.8683402606386 ; median=750.0 ; mode=1
Для столбца зарплата: min=10047 ; max=299190 ; ave=89567.54169446297 ; disp=4220988524.92533 ; std=64969.135171443755 ; median=71196.0 ; mode=30472
Для столбца проекты: min=1 ; max=60 ; ave=21.211474316210808 ; disp=238.11212459396606 ; std=15.430882171605292 ; median=18.0 ; mode=18

CSV файл проанализирован (Нажмите Enter)
```

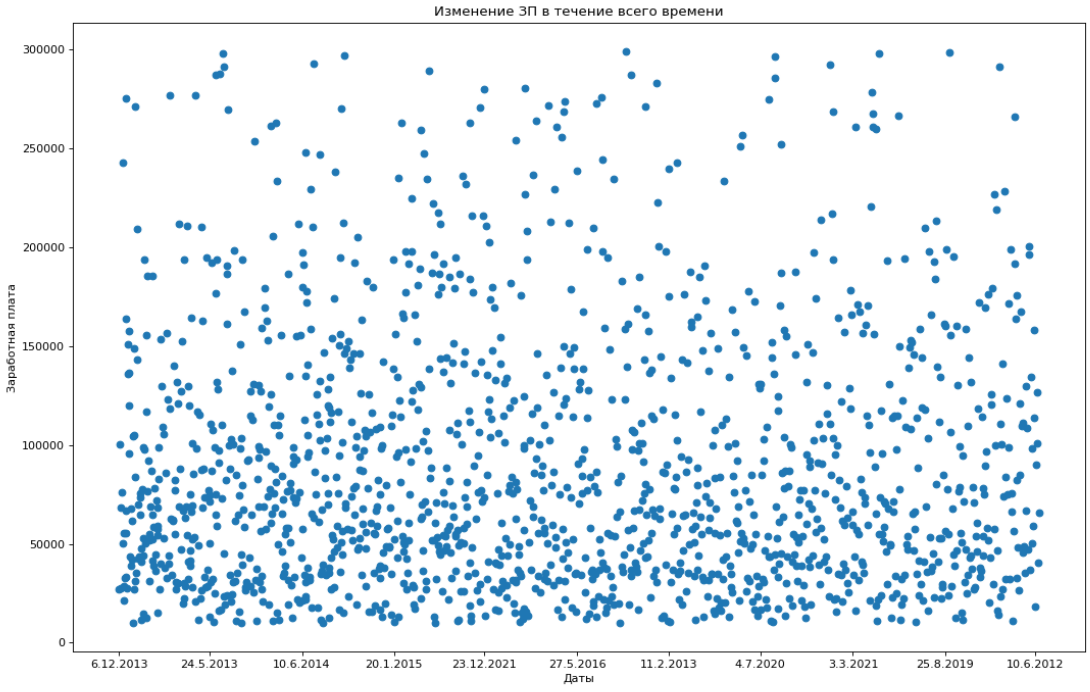
Задание 4.
“Построение графиков”.

Построить не менее 3 разнотипных графиков.

Результат.




Распределение мужчин и женщин в фирме
Мужчины
Женщины


Изменение ЗП в течение всего времени
Заработная плата
300000
250000
200000
150000
100000
50000
0
6.12.2013
24.5.2013
10.6.2014
20.1.2015
23.12.2021
27.5.2016
11.2.2013
4.7.2020
3.3.2021
25.8.2019
10.6.2012
Даты

Задание 5.
“Оценка библиотек numpy и pandas”.

Оценить возможности библиотек csv, numpy, pandas в форме отчета по лабораторной работе.

Результат.

CSV.
Формат CSV (Comma Separated Values) является наиболее часто встречающимся и эффективным форматом для работы с данными в виде таблиц. Формат является достаточно универсальным и позволяет хранить разносортные данные.
Модуль CSV позволяет работать с CSV файлами не задумываясь о их структуре и т.д.
NumPy.
Библиотека NumPy позволяет использовать различные математические операции. Библиотека является быстрой т.к. базируется на коде написанном на языках C/C++/Fortran (скорость работы которых может быть очень высока). Из положительных сторон можно отметить: использование высокоуровневых функций при работе с библиотекой; наличие разнообразных математических средств ; высокая скорость работы.
Pandas.
Библиотека Pandas позволяет осуществлять анализ данных. Библиотека является достаточно быстрой т.к. базируется на библиотеке NumPy. Из положительных сторон можно отметить: использование высокоуровневых функций при работе с библиотекой; наличие разнообразных средств для анализа данных; высокая скорость работы.

Код.

```
import csv
import numpy as nup
from numpy.random import choice
from numpy import genfromtxt
import datetime as dati
import os
import pandas as pads
import matplotlib.pyplot as plt
import matplotlib.ticker as mticker
import matplotlib.dates as mdates

def graphics():

    NewDataFrame = pads.read_csv('new_data.csv')
    numbers = NewDataFrame["number"]
    num_min = numbers.min()
```

```
    num_max = numbers.max()
    num_ave = numbers.mean()
    num_disp = numbers.var()
    num_std = numbers.std()
    num_median = numbers.median()
    num_mode = mode_val(numbers)
    salary = NewDataFrame["salary"]
    sal_min = salary.min()
    sal_max = salary.max()
    sal_ave = salary.mean()
    sal_disp = salary.var()
    sal_std = salary.std()
    sal_median = salary.median()
    sal_mode = mode_val(salary)
    projects = NewDataFrame["completed_projects"]
    pjt_min = projects.min()
    pjt_max = projects.max()
    pjt_ave = projects.mean()
    pjt_disp = projects.var()
    pjt_std = projects.std()
    pjt_median = projects.median()
    pjt_mode = mode_val(projects)

    plt.figure(figsize=(14, 10), dpi=80)
    plt.hlines(y=projects, xmin=0, xmax=salary, color='C0', alpha=0.4, linewidth=5)
    plt.gca().set(ylabel='Колличество проектов', xlabel='Заработная плата')
    plt.title('Зависимость числа проектов от заработной платы', fontdict={'size': 20})
    plt.show()

    data=[NewDataFrame["gender"].value_counts()
["Муж"],NewDataFrame["gender"].value_counts()["Жен"]]
    plt.pie(data, labels=["Мужчины","Женщины"])
    plt.title("Распределение мужчин и женщин в фирме")
    plt.ylabel("")
    plt.show()

    data=NewDataFrame
    myLocator = mticker.MultipleLocator(4)
    plt.figure(figsize=(16, 10), dpi=80)
    plt.plot_date(data["start_date"],data["salary"])
    plt.gca().xaxis.set_major_locator(mdates.AutoDateLocator())
    plt.ylabel('Заработная плата')
    plt.xlabel('Даты')
    plt.title("Изменение ЗП в течение всего времени")
    plt.show()

def mode_val(values):

    dict={}
    for elem in values:
        if elem in dict:
            dict[elem]+=1
        else:
            dict[elem]=1
    v = list(dict.values())
    k = list(dict.keys())

    return k[v.index(max(v))]

def generate_data():

    MyData = [["number", "full_name", "gender", "birth_date", "start_date", "division",
```

```
"position", "salary",
            "completed_projects"]]
    Gender = ["Муж", "Жен"]
    Surnames = ["Антипов", "Бабаев", "Вавилов", "Галкин", "Данилин", "Евсюткин",
"Жеглов", "Задорнов", "Ивачев",
             "Кабаков", "Лабутин",
             "Маврин", "Назаров", "Овсеев", "Павлов", "Райкин", "Савочкин", "Табаков",
"Уваров", "Фандеев",
             "Хабалов", "Царёв", "Чадов",
             "Шаляпин", "Щукин", "Эвентов", "Юров", "Ягодин"]
    Initials = ["А", "Б", "В", "Г", "Д", "Ж", "З", "И", "К", "Л", "М", "Н", "Р", "С", "Т", "У", "Ф",
"Э", "Ю", "Я"]
    Divisions = ["Отдел информационной безопасности", "Отдел разработки ПО", "Отдел
контроля качества и процесов",
              "Отдел развития ИТ", "Отдел поддержки ИТ"]
    Positions = [["Руководитель отдела информационной безопасности", "Специалист",
"Начинающий специалист"],
             ["Руководитель отдела разработки ПО", "Senior разработчик", "Middle
разработчик"],
             ["Руководитель отдела контроля качества и процесов", "Специалист",
"Работник"],
             ["Руководитель отдела развития ИТ", "Специалист", "Работник"],
             ["Руководитель отдела поддержки ИТ", "Специалист", "Работник"]]

    for i in range(1, 1500):
        nup.random.seed(i)
        num = i
        full_name = ""
        gend = ""
        birth = ""
        start = ""
        div = ""
        pos = ""
        salary = 0
        completed_projects = 0

        gender = nup.random.randint(0, 2)
        gend = Gender[gender]
        if (gender != 0):
            full_name = Surnames[nup.random.randint(0, 27)] + "a " +
Initials[nup.random.randint(0, 19)] + "." + Initials[
                nup.random.randint(0, 19)] + "."
        else:
            full_name = Surnames[nup.random.randint(0, 27)] + " " +
Initials[nup.random.randint(0, 19)] + "." + Initials[
                nup.random.randint(0, 19)] + "."
        current_date = dati.date.today()
        year = current_date.year - nup.random.randint(0, 11)
        month = nup.random.randint(1, 13)
        day = nup.random.randint(1, 29)
        start = str(day) + "." + str(month) + "." + str(year)
        byear = year - nup.random.randint(18, 31)
        bmonth = nup.random.randint(1, 13)
        bday = nup.random.randint(1, 29)
        birth = str(bday) + "." + str(bmonth) + "." + str(byear)
        divis = nup.random.randint(0, 5)
        div = Divisions[divis]
        posit = choice([0, 1, 2], 1, [0.1, 0.3, 0.6])[0]
        pos = Positions[divis][posit]
        salary = nup.random.randint(10000, 20001) * (5 - divis) * (3 - posit)
        completed_projects = nup.random.randint(1, 21) * (3 - posit)
```

```
        MyData.append([num, full_name, gend, birth, start, div, pos, salary,
completed_projects])

    for per in MyData:
        for param in per:
            print(param, end=" , ")
        print("")

    with open("new_data.csv", mode="w", encoding='utf-8') as w_file:
        file_writer = csv.writer(w_file, delimiter=",", lineterminator="\r")
        for per in MyData:
            file_writer.writerow(per)

def nup_statistic():

    MyData = []

    with open('new_data.csv', mode="r", encoding='utf-8') as f:
        reader = csv.reader(f)
        for row in reader:
            data=row

MyData.append([data[0],data[1],data[2],data[3],data[4],data[5],data[6],data[7],data[8]])

    for per in MyData:
        for param in per:
            print(param, end=" , ")
        print("")

    MyData=nup.array(MyData)
    numbers=nup.array(MyData[:,0])
    numbers=nup.delete(numbers, 0)
    numbers=[int(item) for item in numbers]
    num_min=nup.min(numbers)
    num_max=nup.max(numbers)
    num_ave= nup.average(numbers)
    num_disp=nup.var(numbers)
    num_std=nup.std(numbers)
    num_median=nup.median(numbers)
    num_mode=mode_val(numbers)
    salary=nup.array(MyData[:,7])
    salary=nup.delete(salary,0)
    salary = [int(item) for item in salary]
    sal_min=nup.min(salary)
    sal_max=nup.max(salary)
    sal_ave=nup.average(salary)
    sal_disp=nup.var(salary)
    sal_std=nup.std(salary)
    sal_median = nup.median(salary)
    sal_mode=mode_val(salary)
    projects=nup.array(MyData[:,8])
    projects = nup.delete(projects, 0)
    projects = [int(item) for item in projects]
    pjt_min = nup.min(projects)
    pjt_max = nup.max(projects)
    pjt_ave = nup.average(projects)
    pjt_disp = nup.var(projects)
    pjt_std = nup.std(projects)
    pjt_median = nup.median(projects)
    pjt_mode=mode_val(projects)
    print(numbers)
```

```
    print("")
    print("Статистические данные")
    print("")
    print("Для столбца номер: min="+str(num_min)+" ; max="+str(num_max)+" ;
ave="+str(num_ave)+" ; disp="+str(num_disp)+" ; std="+str(num_std)+" ;
median="+str(num_median)+" ; mode="+str(num_mode))
    print("Для столбца зарплата: min=" + str(sal_min) + " ; max=" + str(sal_max) + " ;
ave=" + str(sal_ave) + " ; disp=" + str(sal_disp) + " ; std=" + str(sal_std) + " ; median=" +
str(sal_median) + " ; mode=" + str(sal_mode))
    print("Для столбца проекты: min=" + str(pjt_min) + " ; max=" + str(pjt_max) + " ;
ave=" + str(pjt_ave) + " ; disp=" + str(pjt_disp) + " ; std=" + str(pjt_std) + " ; median=" +
str(pjt_median) + " ; mode=" + str(pjt_mode))

def pads_statistic():

    MyDataFrame = pads.read_csv('new_data.csv')
    numbers=MyDataFrame["number"]
    num_min = numbers.min()
    num_max = numbers.max()
    num_ave = numbers.mean()
    num_disp = numbers.var()
    num_std = numbers.std()
    num_median = numbers.median()
    num_mode = mode_val(numbers)
    salary = MyDataFrame["salary"]
    sal_min = salary.min()
    sal_max = salary.max()
    sal_ave = salary.mean()
    sal_disp = salary.var()
    sal_std = salary.std()
    sal_median = salary.median()
    sal_mode = mode_val(salary)
    projects = MyDataFrame["completed_projects"]
    pjt_min = projects.min()
    pjt_max = projects.max()
    pjt_ave = projects.mean()
    pjt_disp = projects.var()
    pjt_std = projects.std()
    pjt_median = projects.median()
    pjt_mode = mode_val(projects)
    print(MyDataFrame.to_string())

    print("")
    print("Статистические данные")
    print("")
    print("Для столбца номер: min=" + str(num_min) + " ; max=" + str(num_max) + " ;
ave=" + str(num_ave) + " ; disp=" + str(num_disp) + " ; std=" + str(num_std) + " ;
median=" + str(num_median) + " ; mode=" + str(num_mode))
    print("Для столбца зарплата: min=" + str(sal_min) + " ; max=" + str(sal_max) + " ;
ave=" + str(sal_ave) + " ; disp=" + str(sal_disp) + " ; std=" + str(sal_std) + " ; median=" +
str(sal_median) + " ; mode=" + str(sal_mode))
    print("Для столбца проекты: min=" + str(pjt_min) + " ; max=" + str(pjt_max) + " ;
ave=" + str(pjt_ave) + " ; disp=" + str(pjt_disp) + " ; std=" + str(pjt_std) + " ; median=" +
str(pjt_median) + " ; mode=" + str(pjt_mode))

if __name__ == '__main__':

    print("Процесс генерации")
    print("")

    generate_data()
```

```
print("")
print("CSV файл создан (Нажмите Enter)")
input()

print("Numpy статистика")
print("")

nup_statistic()

print("")
print("CSV файл проанализирован (Нажмите Enter)")
input()

print("Pandas статистика")
print("")

pads_statistic()

print("")
print("CSV файл проанализирован (Нажмите Enter)")
input()

print("Графики")
print("")

graphics()

print("")
print("CSV файл удалён")
os.remove("new_data.csv")
```