

**НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
УКРАИНЫ
“КИЕВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ”
ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
КАФЕДРА ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ**

ЛАБОРАТОРНАЯ РАБОТА №2

Дисциплина: «Специальные разделы программирования»

Тема: «Наука про данные: обмен результатами и начальный анализ»

Выполнила

студентка 2 курса группы ФИ-41

Лавягина Ольга Алексеевна

Проверил

Колотий Андрей Всеволодович

1 ЗАДАНИЕ

- Зарегистрироваться на сайте GitHub, создать репозиторий, добавить в репозиторий код и данные из лабораторной работы №1, продемонстрировать навыки работы с системой контроля версий git на работе с проектом GitHub;
- создать веб-приложение с использованием модуля Spurge, которое позволит:
 - выбрать часовой ряд VCI, TCI, VNI для наборы данных из лабораторной работы 1 (выпадающий список);
 - выбрать область, для которой будет выполняться анализ (выпадающий список);
 - указать интервал недель, за которые отбираются данные;
 - создать несколько вкладок для отображения таблицы с данными на графике хода индексов;
- код разработанного приложения добавить к созданному репозиторию.

2 ЛИСТИНГ КОДА

```

from spyre import server

import pandas as pd
import urllib2
import matplotlib.pyplot as plt
import numpy as np

class StockExample(server.App):
    title = "Inputs"

    inputs = [{
        "type": 'dropdown',
        "label": 'Index',
        "options" : [ {"label": "VCI", "value": "VCI"},
                      {"label": "TCI", "value": "TCI"},
                      {"label": "VHI", "value": "VHI"}],
        "key": 'index',
        "action_id": "update_data"},

        {
            "type": 'dropdown',
            "label": 'Region',
            "options" : [ {"label": "Vinnitsya", "value": "01"},
                          {"label": "Volyn", "value": "02"},
                          {"label": "Dnipropetrovsk", "value": "03"},
                          {"label": "Donetsk", "value": "04"},
                          {"label": "Zhytomyr", "value": "05"},
                          {"label": "Transcarpathia", "value": "06"},
                          {"label": "Zaporizhzhya", "value": "07"},
                          {"label": "Ivano-Frankivsk", "value": "08"},
                          {"label": "Kiev", "value": "09"},
                          {"label": "Kirovohrad", "value": "10"},
                          {"label": "Luhansk", "value": "11"},
                          {"label": "Lviv", "value": "12"},
                          {"label": "Mykolayiv", "value": "13"},
                          {"label": "Odessa", "value": "14"},
                          {"label": "Poltava", "value": "15"},
                          {"label": "Rivne", "value": "16"},
                          {"label": "Sumy", "value": "17"},
                          {"label": "Ternopil", "value": "18"},
                          {"label": "Kharkiv", "value": "19"},
                          {"label": "Kherson", "value": "20"},
                          {"label": "Khmelnysky", "value": "21"},
                          {"label": "Cherkasy", "value": "22"},
                          {"label": "Chernivtsi", "value": "23"},
                          {"label": "Chernihiv", "value": "24"},
                          {"label": "Crimea", "value": "25"},
                          {"label": "KievCity", "value": "26"},

```

```

        {"label": "Sevastopol", "value": "27"}],
        "key": 'region',
        "action_id": "update_data"},

{ "input_type": "text",
  "variable_name": "year",
  "label": "Year",
  "value": 1981,
  "key": 'year',
  "action_id": "update_data"},

{ "type": 'slider',
  "label": 'First□week',
  "min" : 1, "max" : 52, "value" : 35,
  "key": 'first',
  "action_id": 'update_data'},

{ "type": 'slider',
  "label": 'Last□week',
  "min" : 1, "max" : 52, "value" : 35,
  "key": 'last',
  "action_id": 'update_data'},

{ "type": 'slider',
  "label": 'Percent□of□area',
  "min" : 0, "max" : 100, "value" : 0,
  "key": 'percent',
  "action_id": 'update_data'},

{ "type": 'slider',
  "label": 'Minimum□VHI',
  "min" : 0, "max" : 100, "value" : 0,
  "key": 'minimum',
  "action_id": 'update_data'},

{ "type": 'slider',
  "label": 'Maximum□VHI',
  "min" : 0, "max" : 100, "value" : 100,
  "key": 'maximum',
  "action_id": 'update_data'},]

controls = [{ "type" : "hidden",
               "id" : "update_data"}]

tabs = ["Plot", "Table", "Drought"]

outputs = [{ "type" : "plot",
              "id" : "plot",
              "control_id" : "update_data",

```

```

        "tab" : "Plot"},
    { "type" : "table",
      "id" : "table_id",
      "control_id" : "update_data",
      "tab" : "Table"},
    { "type" : "html",
      "id" : "html_id",
      "control_id" : "update_data",
      "tab" : "Drought"}]]

def getData(self, params):
    index = params['index']
    region = params['region']
    year = params['year']
    first = params['first']
    last = params['last']

    path = '/home/helga/ipt/proga/lab1/clean_data/06_03_5pm{}.csv'.format(region)

    df = pd.read_csv(path, index_col=False, header=True,
                     names=['year', 'week', 'SMN', 'SMT', 'VCI', 'TCI', 'VHI', 'VHI<15', 'VHI<35'])
    df1 = df[(df['year'] == int(year)) & (df['week'] >= int(first)) & (df['week'] <= int(last))]
    df1 = df1[['week', index]]
    return df1

def getPlot(self, params):
    index = params['index']
    year = params['year']
    first = params['first']
    last = params['last']
    df = self.getData(params).set_index('week')
    plt_obj = df.plot()
    plt_obj.set_ylabel(index)
    plt_obj.set_title('Index_{}_for_{}_from_{}_to_{}_weeks'.format(index=index,
        year=int(year), first=int(first), last=int(last)))
    fig = plt_obj.get_figure()
    return fig

def getHTML(self, params):
    region = params['region']
    minimum = params['minimum']
    maximum = params['maximum']
    percent = params['percent']

    path = '/home/helga/ipt/proga/lab1/clean_data/06_03_5pm{}.csv'.format(region)
    df = pd.read_csv(path, index_col=False, header=True,
                     names=['year', 'week', 'SMN', 'SMT', 'VCI', 'TCI', 'VHI', 'VHI<15', 'VHI<35'])
    df1 = df[(df['VHI'] < int(maximum)) & (df['VHI'] > int(minimum)) & (df['VHI<15'] > int(percent))]
    df1 = df1[['year', 'VHI', 'VHI<15']]

```

```
    return 'Years with percent of area > {percent} with drought: {years}'.format(percent=int(percent),
    years = pd.unique(df1.year.ravel()))

app = StockExample()
app.launch()
```

3 ПОЯСНЕНИЕ

Класс SimpleApp наследует сервер. В работе было сделано веб-приложение, которое содержит 3 вкладки (Plot, Table и Drought), то есть отображает таблицу, график и HTML (текст). Для этого были переопределены методы getData, getPlot и getHTML. Метод getData получает и генерирует данные, которые будут отображаться в таблице. Так же как getPlot и getHTML, он принимает на вход аргумент params, который является словарём, содержащим все входные переменные. Метод getData возвращает фрейм с данными (pandas DataFrame), метод getPlot — график, а метод getHTML — строку.

Данные, которые необходимо вывести генерируются в данных методах на основе входных данных (inputs), которые указываются в полях ввода. В данной работе было использовано 8 таких полей с типами «dropdown» — выпадающий список (выбор индекса и области), «text» — текст (выбор года) и «slider» — ползунок (выбор диапазона недель, значений индекса VHI, а также минимального процента области, где данный индекс имеет значение меньше 15).

Результатами (выходами, outputs) являются «plot», «table» и «html». Все результаты загружаются на страницу по умолчанию. Приложение требует множественный вывод, поэтому каждый результат находится на отдельной вкладке.

В работе отображаются значения индексов VHI, THI и VHI для выбранной области за определённый период в виде таблицы и графика, каждый на своей вкладке. Также на вкладке Drought отображаются года, когда индекс VHI и процент области с индексом VHI были в пределах выбранных значений.

ВЫВОДЫ