 Міністерствоосвіти, науки, сім’ї, молоді та спорту України

Національний технічний університет України

«Київський політехнічний інститут

Лабораторна робота №1

*з дисципліни* Архітектура комп’ютерів

Варіант №5

Виконали студенти групи КВ-31

Хуповець Дмитро

Коршун Артем

Чухлиб Юрій

Перевірив(ла)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Київ – 2016

**Постановка задачі**

Мета роботи -­ оволодіти процедурними конструкціями мови Python та основним інструментарієм розробника.

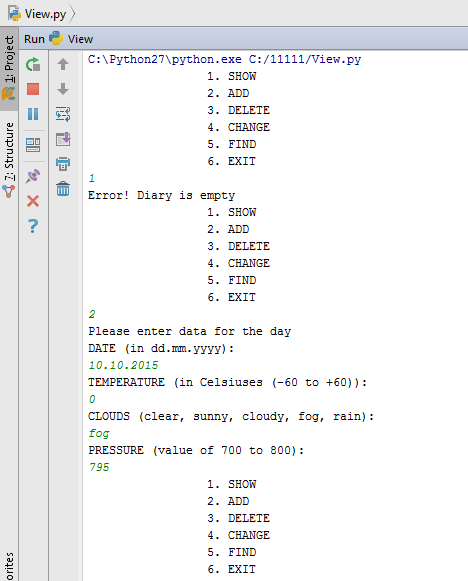
**Варіант завдання:**

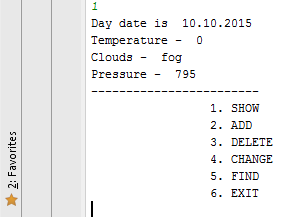
Щоденник погоди

**Посилання на репозиторій:**

<https://github.com/MrDeveloper4/PythonLaba1>

**Приклад сесії взаємодії з розробленими програмними засобами**:





**Код програми:**

**controller.py**

**import** doctest  
**import** re  
**import** pydoc  
  
*# checking date on correct***def** checkingDate(date):  
 *"""  
  
 >>> checkingDate('12.02.2015')  
 1  
 """* **if** (len(date) == 10):  
 **if** ((date[0] == **'1'**) **or** (date[0] == **'2'**)):  
 **if** (date[3] == **'1'**):  
 p = re.compile(**r"^ [0-2][0-9][.][1][0-2][.][0-9][0-9][0-9]** [0-9]$**")  
 if** (date[3] == **'0'**):  
 p = re.compile(**r"^ [0-2][0-9][.][0][0-9][.][0-9][0-9][0-9]** [0-9]$**")  
 if** (date[0] == **'3'**):  
 **if** (date[3] == **'1'**):  
 p = re.compile(**r"^ [03][0-1][.][1][0-2][.][0-9][0-9][0-9]** [0-9]$**")  
 if** (date[3] == **'0'**):  
 p = re.compile(**r"^[3][0-1][.][0][0-9][.][0-9][0-9][0-9][0-9]$"**)  
 **if** p.search(date):  
 **return** 1  
  
*# checking temperature on correct***def** checkingTemperature(temperature):  
 *"""  
  
 >>> checkingTemperature(2)  
 1  
 """* a = int(temperature)  
 **if** ((a >= -60) **and** (a <= 60)):  
 **return** 1  
  
*# checking clouds on correct***def** checkingClouds(clouds):  
 *"""  
 ;param checkingClouds: contact temperature  
 >>> checkingClouds('clear')  
 1  
 """* reserved\_clouds = [**'clear'**, **'sunny'**, **'cloudy'**, **'fog'**, **'rain'**]  
 **for** i **in** reserved\_clouds:  
 **if** i == clouds:  
 **return** 1  
  
*# checking pressure on correct***def** checkingPressure(pressure):  
 *"""  
 ;param checkingPressure: contact temperature  
 >>> checkingPressure(750)  
 1  
 """* a = int(pressure)  
 **if** ((a >= 700) **and** (a <= 800)):  
 **return** 1  
  
**if** (\_\_name\_\_ == **'\_\_main\_\_'**):  
 **import** doctest  
 doctest.testmod()

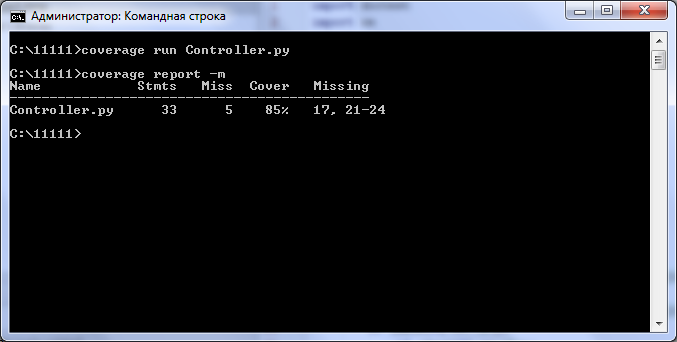
**model.py**

*# class which contains constructor, destructor and methods  
# for change attributes this class***class** CalendarDay:  
 **def** \_\_init\_\_(self, date, temperature, clouds, pressure):  
 self.date = date  
 self.temperature = temperature  
 self.clouds = clouds  
 self.pressure = pressure  
  
 **def** displayDayInfo(self):  
 **print "Day date is "**, self.date  
 **print "Temperature - "**, self.temperature  
 **print "Clouds - "**, self.clouds  
 **print "Pressure - "**, self.pressure  
  
 **def** setNewTemperature(self, temperature):  
 self.temperature = temperature  
  
 **def** setNewClouds(self, clouds):  
 self.clouds = clouds  
  
 **def** setNewPressure(self, pressure):  
 self.pressure = pressure  
  
diary = []

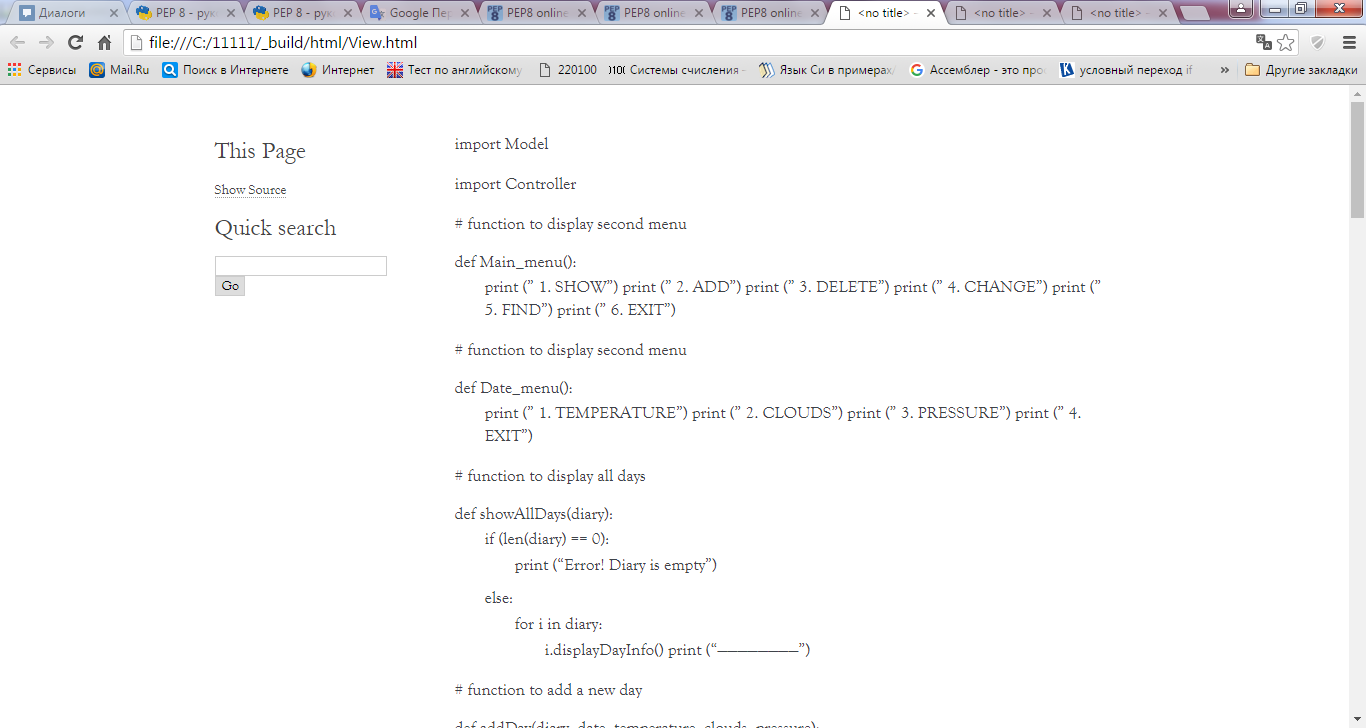
**view.py**

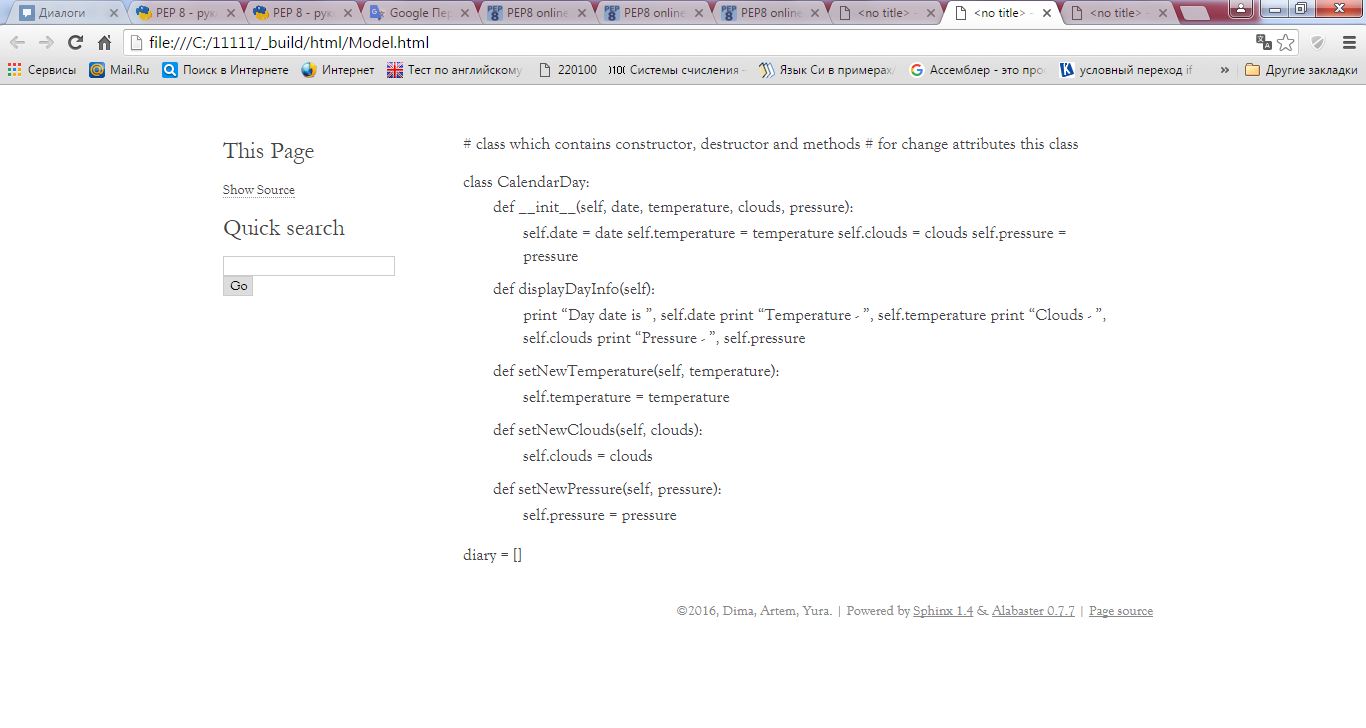
**import** Model  
**import** Controller  
  
*# function to display second menu***def** Main\_menu():  
 **print** (**" 1. SHOW"**)  
 **print** (**" 2. ADD"**)  
 **print** (**" 3. DELETE"**)  
 **print** (**" 4. CHANGE"**)  
 **print** (**" 5. FIND"**)  
 **print** (**" 6. EXIT"**)  
  
*# function to display second menu***def** Date\_menu():  
 **print** (**" 1. TEMPERATURE"**)  
 **print** (**" 2. CLOUDS"**)  
 **print** (**" 3. PRESSURE"**)  
 **print** (**" 4. EXIT"**)  
  
*# function to display all days***def** showAllDays(diary):  
 **if** (len(diary) == 0):  
 **print** (**"Error! Diary is empty"**)  
 **else**:  
 **for** i **in** diary:  
 i.displayDayInfo()  
 **print** (**"------------------------"**)  
  
*# function to add a new day***def** addDay(diary, date, temperature, clouds, pressure):  
 **for** i **in** diary:  
 **if** i.date == date:  
 **print**(**"Error! Date already exists"**)  
 **else**:  
 diary.append(Model.CalendarDay(date, temperature, clouds, pressure))  
  
  
Main\_menu()  
  
**while** True:  
 key = raw\_input()  
 **if** key == **"1"**:  
 showAllDays(Model.diary)  
 Main\_menu()  
 **elif** key == **"2"**:  
 **print** (**"Please enter data for the day"**)  
 **while** True:  
 **print** (**"DATE (in dd.mm.yyyy): "**)  
 date = raw\_input()  
 **if** (Controller.checkingDate(date)):  
 **break  
  
 while** True:  
 **print** (**"TEMPERATURE (in Celsiuses (-60 to +60)): "**)  
 temperature = raw\_input()  
 **if** (Controller.checkingTemperature(temperature)):  
 **break  
  
 while** True:  
 **print** (**"CLOUDS (clear, sunny, cloudy, fog, rain): "**)  
 clouds = raw\_input()  
 **if** (Controller.checkingClouds(clouds)):  
 **break  
  
 while** True:  
 **print** (**"PRESSURE (value of 700 to 800): "**)  
 pressure = raw\_input()  
 **if** (Controller.checkingPressure(pressure)):  
 **break** addDay(Model.diary, date, temperature, clouds, pressure)  
 date = None  
 temperature = None  
 clouds = None  
 pressure = None  
 Main\_menu()  
 **elif** key == **"3"**:  
 **while** True:  
 **print** (**"DATE (in dd.mm.yyyy): "**)  
 date = raw\_input()  
 **if** (Controller.checkingDate(date)):  
 **break  
  
 for** day **in** Model.diary:  
 **if** day.date == date:  
 Model.diary.remove(day)  
 **break  
 else**:  
 **print** (**"Error! Date not found"**)  
 date = None  
 Main\_menu()  
 **elif** key == **"4"**:  
  
 **while** True:  
 **print** (**"DATE (in dd.mm.yyyy): "**)  
 date = raw\_input()  
 **if** (Controller.checkingDate(date)):  
 **break  
  
 for** day **in** Model.diary:  
 **if** day.date == date:  
 Date\_menu()  
 **while** True:  
 key = raw\_input()  
 **if** key == **"1"**:  
 **while** True:  
 **print** (**"TEMPERATURE (in Celsiuses): "**)  
 temperature = raw\_input()  
 **if** (Controller.checkingTemperature(temperature)):  
 **break** day.setNewTemperature(temperature)  
 Date\_menu()  
 temperature = None  
 **elif** key == **"2"**:  
 **while** True:  
 **print** (**"CLOUDS (clear, sunny, cloudy, fog, rain):"**)  
 clouds = raw\_input()  
 **if** (Controller.checkingClouds(clouds)):  
 **break** day.setNewClouds(clouds)  
 Date\_menu()  
 clouds = None  
 **elif** key == **"3"**:  
 **while** True:  
 **print** (**"PRESSURE (value of 700 to 800): "**)  
 pressure = raw\_input()  
 **if** (Controller.checkingPressure(pressure)):  
 **break** day.setNewPressure(pressure)  
 Date\_menu()  
 pressure = None  
 **elif** key == **"4"**:  
 Main\_menu()  
 **break  
  
 break  
 else**:  
 **print** (**"Error! Date not found"**)  
 **elif** key == **"5"**:  
 **while** True:  
 **print** (**"DATE (in dd.mm.yyyy): "**)  
 date = raw\_input()  
 **if** (Controller.checkingDate(date)):  
 **break  
 for** day **in** Model.diary:  
 **if** day.date == date:  
 day.displayDayInfo()  
 **break  
 else**:  
 **print** (**"Error! Date not found"**)  
 Main\_menu()  
 **elif** key == **"6"**:  
 **print** (**"Goodbye"**)  
 **break  
  
if** (\_\_name\_\_ == **'\_\_main\_\_'**):  
 **import** doctest  
 doctest.testmod()

**Покриття тестами:**



**Згенерована документація:**

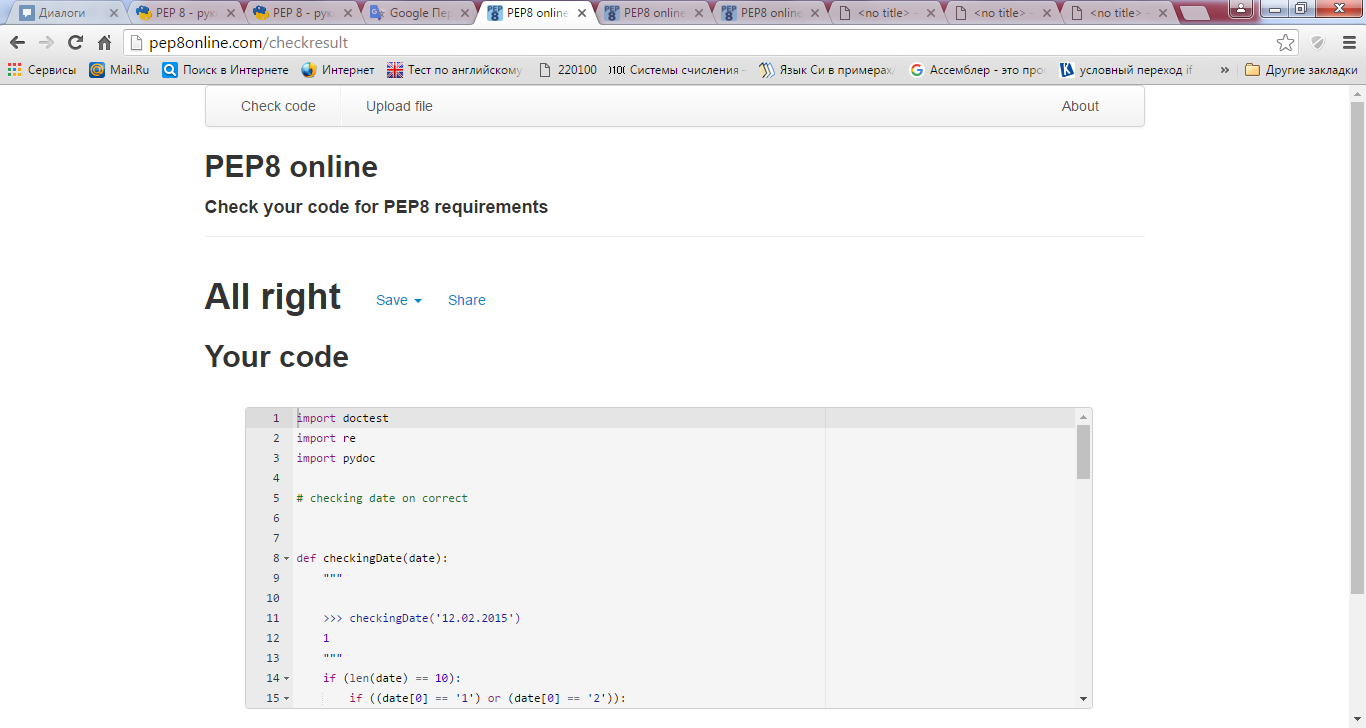




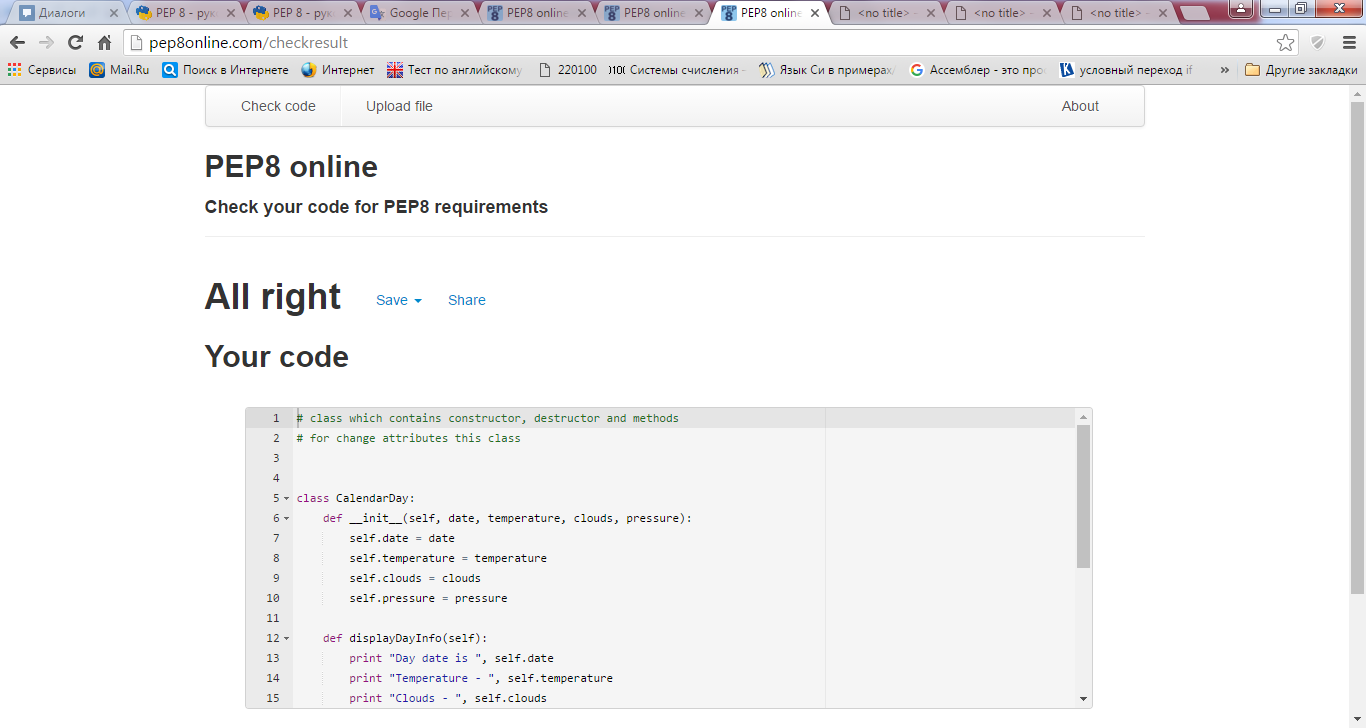


**Звіт програм pep8, pyflakes:**

Controller.py



Model.py



View.py

