**4**

**Міністерство освіти та науки України  
Національний технічний університет України  
«Київський політехнічний інститут ім. Сікорського»  
Факультет прикладної математики  
Кафедра системного програмування і спеціалізованих  
комп’ютерних систем**

**Лабораторна робота №1**з дисципліни

**«Архітектура комп’ютерів 2»**

Виконав: Любимов Олександр Сергійович

Латюк Сергій Олександрович

Федоров Данило \_\_\_\_\_\_\_\_\_\_\_

Студент групи КВ-42

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

**м. Київ**

**2017**

**Pep8**

D:\Cypress\MoneyAccounter>pep8 controller.py

D:\Cypress\MoneyAccounter>pep8 view.py

view.py:10:80: E501 line too long (87 > 79 characters)

view.py:58:80: E501 line too long (93 > 79 characters)

view.py:72:80: E501 line too long (92 > 79 characters)

view.py:73:80: E501 line too long (90 > 79 characters)

D:\Cypress\MoneyAccounter>pep8 Accounter.py

**Pyflakes**

D:\Cypress\MoneyAccounter>pyflakes view.py

view.py:1: 'from PyQt5.QtWidgets import \*' used; unable to detect undefined names

view.py:9: 'QTableWidgetItem' may be undefined, or defined from star imports: PyQt5.QtWidgets

view.py:10: 'QTableWidgetItem' may be undefined, or defined from star imports: PyQt5.QtWidgets

view.py:11: 'QTableWidgetItem' may be undefined, or defined from star imports: PyQt5.QtWidgets

view.py:19: 'QTableWidgetItem' may be undefined, or defined from star imports: PyQt5.QtWidgets

view.py:20: 'QTableWidgetItem' may be undefined, or defined from star imports: PyQt5.QtWidgets

view.py:21: 'QTableWidgetItem' may be undefined, or defined from star imports: PyQt5.QtWidgets

view.py:24: 'QHeaderView' may be undefined, or defined from star imports: PyQt5.QtWidgets

D:\Cypress\MoneyAccounter>pyflakes controller.py

controller.py:2: 'from PyQt5.QtCore import \*' used; unable to detect undefined names

controller.py:3: 'from PyQt5.QtGui import \*' used; unable to detect undefined names

controller.py:6: 'from view import \*' used; unable to detect undefined names

controller.py:11: 'QApplication' may be undefined, or defined from star imports: PyQt5.QtCore, PyQt5.QtGui, view

controller.py:12: 'QDialog' may be undefined, or defined from star imports: PyQt5.QtCore, PyQt5.QtGui, view

controller.py:13: 'QShortcut' may be undefined, or defined from star imports: PyQt5.QtCore, PyQt5.QtGui, view

controller.py:13: 'QKeySequence' may be undefined, or defined from star imports: PyQt5.QtCore, PyQt5.QtGui, view

controller.py:17: 'QDate' may be undefined, or defined from star imports: PyQt5.QtCore, PyQt5.QtGui, view

controller.py:18: 'QDate' may be undefined, or defined from star imports: PyQt5.QtCore, PyQt5.QtGui, view

controller.py:19: 'QDate' may be undefined, or defined from star imports: PyQt5.QtCore, PyQt5.QtGui, view

controller.py:25: 'load\_to\_table' may be undefined, or defined from star imports: PyQt5.QtCore, PyQt5.QtGui, view

controller.py:27: 'make\_push\_button\_clicked' may be undefined, or defined from star imports: PyQt5.QtCore, PyQt5.QtGui, view

controller.py:28: 'make\_pop\_button\_clicked' may be undefined, or defined from star imports: PyQt5.QtCore, PyQt5.QtGui, view

controller.py:29: 'make\_filter\_button\_clicked' may be undefined, or defined from star imports: PyQt5.QtCore, PyQt5.QtGui, view

D:\Cypress\MoneyAccounter>pyflakes Accounter.py

**Coverage**

D:\Cypress\MoneyAccounter>coverage report Accounter.py

Name Stmts Miss Cover

----------------------------------

Accounter.py 84 17 80%

**Documentation**

**Help on module Accounter:**

NAME

Accounter

CLASSES

builtins.object

Accounter

class Accounter(builtins.object)

| This class implements simple money accounter.

| You can use it for storing notes about your income and

| outcome, search notes and group them by their attributes.

|

| Methods defined here:

|

| \_\_init\_\_(self, acc=Empty DataFrame

| Columns: []

| Index: [])

| Create new Accounter object from existing DataFrame.

| >>> print(Accounter())

| List of notes is empty

| >>> print(Accounter(pd.DataFrame([{'date': pd.Timestamp("20170404"),

| ... 'value': 100, 'comment': 'new money'}])))

| comment date value

| 0 new money 2017-04-04 100

|

| \_\_str\_\_(self)

| Return str(self).

|

| add\_new\_data(self, value, comment, date='20170411')

| Create new note from given value, date and comment.

| Date is read automatically from OS time.

| Date is python date.

| >>> acc = Accounter()

| >>> acc.add\_new\_data(25, 'nashel v kurtke', '20170321')

| >>> print(acc)

| comment date value

| 0 nashel v kurtke 2017-03-21 25

| >>> acc.add\_new\_data(-45, 'kupil shaurmu', '20170325')

| >>> print(acc)

| comment date value

| 0 nashel v kurtke 2017-03-21 25

| 1 kupil shaurmu 2017-03-25 -45

|

| drop\_data(self)

| Delete all notes.

| >>> acc = Accounter()

| >>> acc.add\_new\_data(25, 'nashel v kurtke', '20170321')

| >>> acc.add\_new\_data(-45, 'kupil shaurmu', '20170325')

| >>> print(acc)

| comment date value

| 0 nashel v kurtke 2017-03-21 25

| 1 kupil shaurmu 2017-03-25 -45

| >>> acc.drop\_data()

| Empty DataFrame

| Columns: []

| Index: []

| >>> print(acc)

| List of notes is empty

|

| get\_by\_comment(self, comm)

| Return Accounter object with notes that have given comment.

| >>> acc1 = Accounter(pd.DataFrame([

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': 250, 'comment': 'Neshta vernul dolg'},

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc1)

| comment date value

| 0 new money 2017-04-04 100

| 1 Neshta vernul dolg 2017-04-04 250

| 2 kupil chai v happy cake 2017-04-04 -25

| 3 zaplatil za obschagu 2017-04-04 -2000

| >>> print(acc1.get\_by\_comment("new money"))

| comment date value

| 0 new money 2017-04-04 100

| >>> print(acc1.get\_by\_comment("not com"))

| List of notes is empty

|

| get\_by\_date(self, date1, date2)

| Return Accounter object with notes that have date attribute in given range.

| >>> acc1 = Accounter(pd.DataFrame([

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': pd.Timestamp("20170405"),

| ... 'value': 250, 'comment': 'Neshta vernul dolg'},

| ... {'date': pd.Timestamp("20170406"),

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': pd.Timestamp("20170407"),

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc1)

| comment date value

| 0 new money 2017-04-04 100

| 1 Neshta vernul dolg 2017-04-05 250

| 2 kupil chai v happy cake 2017-04-06 -25

| 3 zaplatil za obschagu 2017-04-07 -2000

| >>> print(acc1.get\_by\_date("20170404", "20170405"))

| comment date value

| 0 new money 2017-04-04 100

| 1 Neshta vernul dolg 2017-04-05 250

| >>> print(acc1.get\_by\_date("20170404", "20170404"))

| comment date value

| 0 new money 2017-04-04 100

|

| get\_income(self)

| Return Accounter object only with notes where you had an income.

| >>> acc1 = Accounter(pd.DataFrame([

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': 250, 'comment': 'Neshta vernul dolg'},

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc1)

| comment date value

| 0 new money 2017-04-04 100

| 1 Neshta vernul dolg 2017-04-04 250

| 2 kupil chai v happy cake 2017-04-04 -25

| 3 zaplatil za obschagu 2017-04-04 -2000

| >>> print(acc1.get\_income())

| comment date value

| 0 new money 2017-04-04 100

| 1 Neshta vernul dolg 2017-04-04 250

| >>> acc2 = Accounter(pd.DataFrame([

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc2.get\_income())

| List of notes is empty

| >>> acc1.drop\_data()

| Empty DataFrame

| Columns: []

| Index: []

| >>> print(acc1.get\_income())

| List of notes is empty

|

| get\_income\_by\_range(self, from\_value, to\_value)

| Return Accounter object with notes that have income in given range.

| >>> acc = Accounter(pd.DataFrame([

| ... {'date': pd.Timestamp('20170404'),

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': pd.Timestamp('20170404'),

| ... 'value': 250, 'comment': 'Neshta vernul dolg'},

| ... {'date': pd.Timestamp('20170404'),

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': pd.Timestamp('20170404'),

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc.get\_income\_by\_range(200, 500))

| comment date value

| 1 Neshta vernul dolg 2017-04-04 250

|

| get\_income\_sum(self)

| Return sum of all incomes.

| >>> acc1 = Accounter(pd.DataFrame([

| ... {'date': "20170404",

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': "20170403",

| ... 'value': 250, 'comment': 'new money'},

| ... {'date': "20170402",

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': "20170405",

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc1)

| comment date value

| 0 new money 20170404 100

| 1 new money 20170403 250

| 2 kupil chai v happy cake 20170402 -25

| 3 zaplatil za obschagu 20170405 -2000

| >>> print(acc1.get\_income\_sum())

| 350

|

| get\_outcome(self)

| Return Accounter object only with notes where you had an outcome.

| >>> acc1 = Accounter(pd.DataFrame([

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': 250, 'comment': 'Neshta vernul dolg'},

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc1)

| comment date value

| 0 new money 2017-04-04 100

| 1 Neshta vernul dolg 2017-04-04 250

| 2 kupil chai v happy cake 2017-04-04 -25

| 3 zaplatil za obschagu 2017-04-04 -2000

| >>> print(acc1.get\_outcome())

| comment date value

| 2 kupil chai v happy cake 2017-04-04 -25

| 3 zaplatil za obschagu 2017-04-04 -2000

| >>> acc2 = Accounter(pd.DataFrame([

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': 250, 'comment': 'Neshta vernul dolg'}

| ... ]))

| >>> print(acc2.get\_outcome())

| List of notes is empty

| >>> acc1.drop\_data()

| Empty DataFrame

| Columns: []

| Index: []

| >>> print(acc1.get\_outcome())

| List of notes is empty

|

| get\_outcome\_by\_range(self, from\_value, to\_value)

| Return Accounter object with notes that have outcome in given range.

| >>> acc = Accounter(pd.DataFrame([

| ... {'date': pd.Timestamp('20170404'),

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': pd.Timestamp('20170404'),

| ... 'value': 250, 'comment': 'Neshta vernul dolg'},

| ... {'date': pd.Timestamp('20170404'),

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': pd.Timestamp('20170404'),

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc.get\_outcome\_by\_range(2000, 4000))

| comment date value

| 3 zaplatil za obschagu 2017-04-04 -2000

|

| get\_outcome\_sum(self)

| Return sum of all outcomes.

| >>> acc1 = Accounter(pd.DataFrame([

| ... {'date': "20170404",

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': "20170403",

| ... 'value': 250, 'comment': 'new money'},

| ... {'date': "20170402",

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': "20170405",

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc1)

| comment date value

| 0 new money 20170404 100

| 1 new money 20170403 250

| 2 kupil chai v happy cake 20170402 -25

| 3 zaplatil za obschagu 20170405 -2000

| >>> print(acc1.get\_outcome\_sum())

| -2025

|

| get\_sum(self)

| Return sum of all income and outcome.

| >>> acc1 = Accounter(pd.DataFrame([

| ... {'date': "20170404",

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': "20170403",

| ... 'value': 250, 'comment': 'new money'},

| ... {'date': "20170402",

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': "20170405",

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc1)

| comment date value

| 0 new money 20170404 100

| 1 new money 20170403 250

| 2 kupil chai v happy cake 20170402 -25

| 3 zaplatil za obschagu 20170405 -2000

| >>> print(acc1.get\_sum())

| -1675

|

| group\_by\_comment(self)

| Return Accounter object with notes grouped by comment.

| >>> acc1 = Accounter(pd.DataFrame([

| ... {'date': pd.Timestamp("20170404"),

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': pd.Timestamp("20170405"),

| ... 'value': 250, 'comment': 'new money'},

| ... {'date': pd.Timestamp("20170406"),

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': pd.Timestamp("20170407"),

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc1)

| comment date value

| 0 new money 2017-04-04 100

| 1 new money 2017-04-05 250

| 2 kupil chai v happy cake 2017-04-06 -25

| 3 zaplatil za obschagu 2017-04-07 -2000

| >>> acc1.group\_by\_comment().account.values

| array([[ -25],

| [ 350],

| [-2000]], dtype=int64)

|

| group\_by\_date(self)

| Return Accounter object with notes grouped by date.

| >>> acc1 = Accounter(pd.DataFrame([

| ... {'date': "20170404",

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': "20170404",

| ... 'value': 250, 'comment': 'new money'},

| ... {'date': "20170405",

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': "20170405",

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc1)

| comment date value

| 0 new money 20170404 100

| 1 new money 20170404 250

| 2 kupil chai v happy cake 20170405 -25

| 3 zaplatil za obschagu 20170405 -2000

| >>> acc1.group\_by\_date().account.values

| array([[ 350],

| [-2025]], dtype=int64)

|

| load\_data(self)

| Load data from binary file using \_pickle.

|

| print\_data(self)

| Show all notes in accounter.

| >>> Accounter(pd.DataFrame([{'date': pd.Timestamp('20170404'),

| ... 'comment': 'new money', 'value': 50}])).print\_data()

| \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

| comment date value

| 0 new money 2017-04-04 50

| >>> Accounter().print\_data()

| \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

| List of notes is empty

|

| save\_data(self)

| Save data into binary file using \_pickle.

|

| sort\_by\_date(self)

| Sort notes using date as key.

| >>> acc = Accounter(pd.DataFrame([

| ... {'date': pd.Timestamp('20170404'),

| ... 'value': 100, 'comment': 'new money'},

| ... {'date': pd.Timestamp('20170402'),

| ... 'value': 250, 'comment': 'Neshta vernul dolg'},

| ... {'date': pd.Timestamp('20170411'),

| ... 'value': -25, 'comment': 'kupil chai v happy cake'},

| ... {'date': pd.Timestamp('20170408'),

| ... 'value': -2000, 'comment': 'zaplatil za obschagu'}

| ... ]))

| >>> print(acc)

| comment date value

| 0 new money 2017-04-04 100

| 1 Neshta vernul dolg 2017-04-02 250

| 2 kupil chai v happy cake 2017-04-11 -25

| 3 zaplatil za obschagu 2017-04-08 -2000

| >>> acc.sort\_by\_date()

| >>> print(acc)

| comment date value

| 1 Neshta vernul dolg 2017-04-02 250

| 0 new money 2017-04-04 100

| 3 zaplatil za obschagu 2017-04-08 -2000

| 2 kupil chai v happy cake 2017-04-11 -25

|

| sort\_by\_indexes(self)

|

| ----------------------------------------------------------------------

| Data descriptors defined here:

|

| \_\_dict\_\_

| dictionary for instance variables (if defined)

|

| \_\_weakref\_\_

| list of weak references to the object (if defined)

**Help on module view:**

NAME

view

FUNCTIONS

load\_to\_table(ui, acc)

Loads contents of acc to table widget

make\_filter\_button\_clicked(acc, ui)

Function for filter button clocking

make\_pop\_button\_clicked(acc, ui)

Function for pop button clocking

make\_push\_button\_clicked(acc, ui)

Function for push button clocking

qDrawBorderPixmap(...)

qDrawBorderPixmap(QPainter, QRect, QMargins, QPixmap)

qDrawPlainRect(...)

qDrawPlainRect(QPainter, int, int, int, int, Union[QColor, Qt.GlobalColor], lineWidth: int = 1, fill: Union[QBrush, QColor, Qt.GlobalColor, QGradient] = None)

qDrawPlainRect(QPainter, QRect, Union[QColor, Qt.GlobalColor, QGradient], lineWidth: int = 1, fill: Union[QBrush, QColor, Qt.GlobalColor, QGradient] = None)

qDrawShadeLine(...)

qDrawShadeLine(QPainter, int, int, int, int, QPalette, sunken: bool = True, lineWidth: int = 1, midLineWidth: int = 0)

qDrawShadeLine(QPainter, QPoint, QPoint, QPalette, sunken: bool = True, lineWidth: int = 1, midLineWidth: int = 0)

qDrawShadePanel(...)

qDrawShadePanel(QPainter, int, int, int, int, QPalette, sunken: bool = False, lineWidth: int = 1, fill: Union[QBrush, QColor, Qt.GlobalColor, QGradient] = None)

qDrawShadePanel(QPainter, QRect, QPalette, sunken: bool = False, lineWidth: int = 1, fill: Union[QBrush, QColor, Qt.GlobalColor, QGradient] = None)

qDrawShadeRect(...)

qDrawShadeRect(QPainter, int, int, int, int, QPalette, sunken: bool = False, lineWidth: int = 1, midLineWidth: int = 0, fill: Union[QBrush, QColor, Qt.GlobalColor, QGradient]

= None)

qDrawShadeRect(QPainter, QRect, QPalette, sunken: bool = False, lineWidth: int = 1, midLineWidth: int = 0, fill: Union[QBrush, QColor, Qt.GlobalColor, QGradient] = None)

qDrawWinButton(...)

qDrawWinButton(QPainter, int, int, int, int, QPalette, sunken: bool = False, fill: Union[QBrush, QColor, Qt.GlobalColor, QGradient] = None)

qDrawWinButton(QPainter, QRect, QPalette, sunken: bool = False, fill: Union[QBrush, QColor, Qt.GlobalColor, QGradient] = None)

qDrawWinPanel(...)

qDrawWinPanel(QPainter, int, int, int, int, QPalette, sunken: bool = False, fill: Union[QBrush, QColor, Qt.GlobalColor, QGradient] = None)

qDrawWinPanel(QPainter, QRect, QPalette, sunken: bool = False, fill: Union[QBrush, QColor, Qt.GlobalColor, QGradient] = None)

row\_to\_table(ui, row\_series, num=None)

Writes row\_series to ui.table

update\_balance(ui, acc)

DATA

QWIDGETSIZE\_MAX = 16777215

qApp = <PyQt5.QtWidgets.QApplication object>

FILE

d:\cypress\moneyaccounter\view.py

**Screenshots**

