MLClass. "Прикладной анализ данных".

Курс "Инструментарий Data Science".

Преподаватель ФКН НИУ ВШЭ Кашницкий Юрий

"Инструментарий" - первый курс серии, посвящённый изучению основ и методологии программирования на языке Python, который широко используется для разработки и реализации методов анализа данных. Рассматриваются основные структуры и алгоритмы, работа с данными разного вида (тексты, бинарные файлы, изображения), создание графических пользовательских интерфейсов, инструменты и средства разработки и отладки программ.

Содержание курса

- Урок 1. 23.09.15. Введение в Python и средства разработки
  - Установка Anaconda и знакомство с интерпретатором Python
  - Обзор и настройка среды разработки PyCharm
  - IPython и Jupyter для воспроизводимых исследований
  - Введение в систему контроля версий git и веб-сервис GitHub
  - Обзор возможностей сборки библиотек Anaconda
  - Регистрация на сайте дистанционного курса по информатике
- Урок 2. 27.09.15. Основы языка Python
  - Типы объектов языка Python
  - Основные операции с типами
  - Обзор стандартной библиотеки
  - Операторы, условные конструкции, циклы
  - Практика решения простейших задач
- Урок 3. 30.09.15. Структуры данных I
  - Последовательности: строки, списки, кортежи
  - Алгоритмы поиска в одномерных списках
  - Обзор стандартной библиотеки
  - Методы сортировки одномерных списков
  - Алгоритмы на строках
  - Разбор самых распространенных задач на списки и строки
- Урок 4. 04.10.15. Структуры данных II
  - Введение в продвинутые структуры данных
  - Словари
  - Стек, очередь, куча, дерево, граф
  - Поиск в глубину и поиск в ширину
  - Обзор классических алгоритмов на графах
  - Разбор задач на слова и множества. Задачи на стек и очередь
- Урок 5. 07.10.15. Функции. Рекурсия
  - Понятие функции, ее сигнатуры
  - Модули
  - Понятие индукции
  - Рекурсия
  - Парадигма "Разделяй и Властвуй"
  - Разбор практических задач на рекурсию

Урок 1. Введение в Python и средства разработки

Часть 1. Установка Anaconda и знакомство с интерпретатором Python

Python - это свободный интерпретируемый объектно-ориентированный расширяемый встраиваемый язык программирования очень высокого уровня (Г.Россум, Ф.Л.Дж.Дрейк, Д.С.Откидач "Язык программирования Python")

- свободный - все исходные тексты открыты для любого использования, даже коммерческого
- интерпретируемый - использует "позднее связывание"
- объектно-ориентированный - классическая ОО модель с множественным наследованием
- расширяемый - имеет строго определенные API для создания модулей, типов и классов на C или C++
- встраиваемый - имеет строго определенные API для встраивания интерпретатора в другие программы
- очень высокого уровня - динамическая типизация, встроенные типы данных высокого уровня (например, словари), классы, модули, исключения

Код на Python чаще всего намного более читаемый, чем на C или C++, потому что:

- типы данных высокого уровня позволяют выражать сложные операции одной или несколькими простыми командами
- группирование инструкций выполняется отступами, а не фигурными скобками
- не надо объявлять переменные

В этом курсе мы используем сборку библиотек Anaconda (http://continuum.io/downloads), включающую интерпретатор Python, библиотеки для научных вычислений NumPy и SciPy, анализа данных Pandas, машинного обучения Scikit-learn и около 200 других.

Следуем инструкциям, проверяем установку. В командной строке набираем

>> python

Дзен языка Python

In [1]:

```
import this
```

```
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to
do it.
Although that way may not be obvious at first unless you're Dut
ch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good ide
a.
Namespaces are one honking great idea -- let's do more of thos
e!
```

Интерпретатор можно использовать в качестве настольного калькулятора

In [4]:

```
print(2 + 2)
print(3 ** 7)
import math
print(round(math.sqrt(17), 2))
```

```
4
2187
4.12
```

Но Python способен и на большее. Выведем все двузначные члены последовательности Фиббоначи.

In [9]:

```
a, b = 1, 1
while a < 100:
    print(ax)
    a, b = b, a + b
```

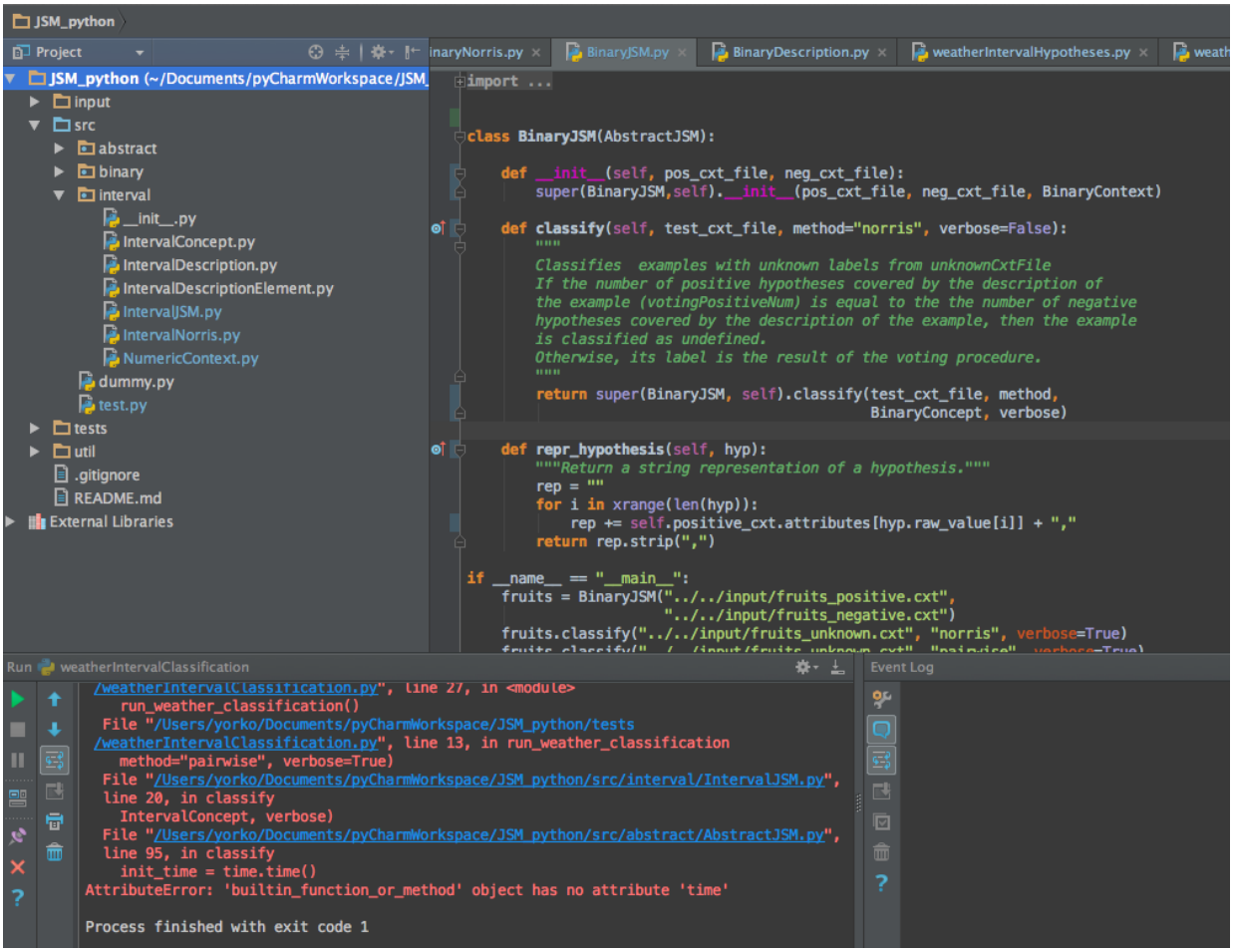
```
1
1
2
3
5
8
13
21
34
55
89
```

Изучать Python продолжим на следующем уроке, а пока настроим среду для грамотной работы с кодом и решения задач.

Часть 2. Обзор и настройка среды разработки PyCharm

PyCharm (https://www.jetbrains.com/pycharm/) компании JetBrains - одна из самых популярных сред разработки (IDE) на языке Python. Из плюсов, перечисленных в одном из обзоров (http://habrahabr.ru/post/122018/) и важных для нас в этом курсе:

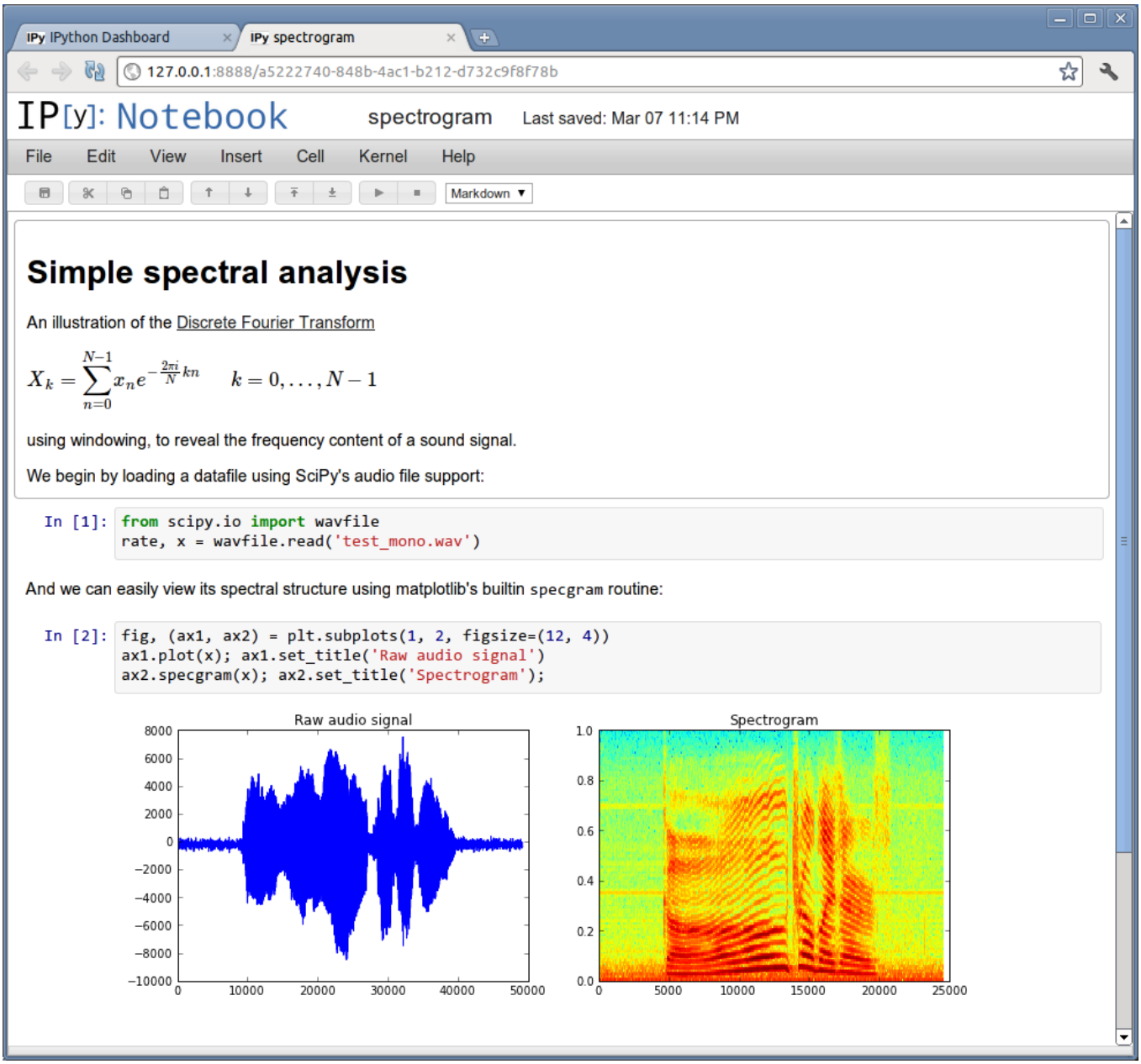
- интуитивно понятная интеграция с Git
- удобное автоматическое дополнение текста
- множество "горячих клавиш" для перехода к определения сущности, списка упоминаний конструкции, подсказок по сигнатуре функций и прочих мелочей
- встроенная проверка кода на соответствие стандарту PEP



Введение в PyCharm (http://pythonworld.ru/osnovy/pycharm-python-tutorial.html)

Часть 3. IPython и Jupyter для воспроизводимых исследований

То, что Вы сейчас видите - это тетрадка IPython (Ipython notebook), интерактивная среда для вычислений, способная совмещать код, картинки, markdown-разметку и графики.



Пример с сайта http://ipython.org/ (http://ipython.org/)

Тетрадка IPython имеет расширение *.ipynb* и запускается локально командой

>> ipython notebook

Выполнять и редактировать тетрадку можно в браузере http://localhost:8888/ (http://localhost:8888/)

Начиная с версии IPython 4.0, большая часть проекта перешла в Jupyter. Jupyter (http://jupyter.org/) - это веб-приложение для обмена документами, в которых можно выполнять код на 40 популярных в анализе данных языках, включая Python, R, Julia и Scala.

Тетрадки IPython - отличный способ представить идею, сразу поделиться кодом, хороши для воспроизводимости исследований, включающих написание кода.

Часть 4. Введение в систему контроля версий git и веб-сервис GitHub

Git - популярная система контроля версий, GitHub - популярный веб-сервис для совместной работы над IT-проектами. Обзор (http://sixrevisions.com/git/interactive-git-tutorials/) трех ресурсов для интерактивного введения в Git. Еще один (http://githowto.com/).

Скачиваем (клонируем) репозиторий с данной тетрадкой IPython

>> git clone https://github.com/Yorko/python4da_intro.git

Добавим файл. Например, файл primes_sum_1000.py с кодом для подсчета суммы всех простых чисел, не превосходящих 1000.

>> git add primes_sum_1000.py

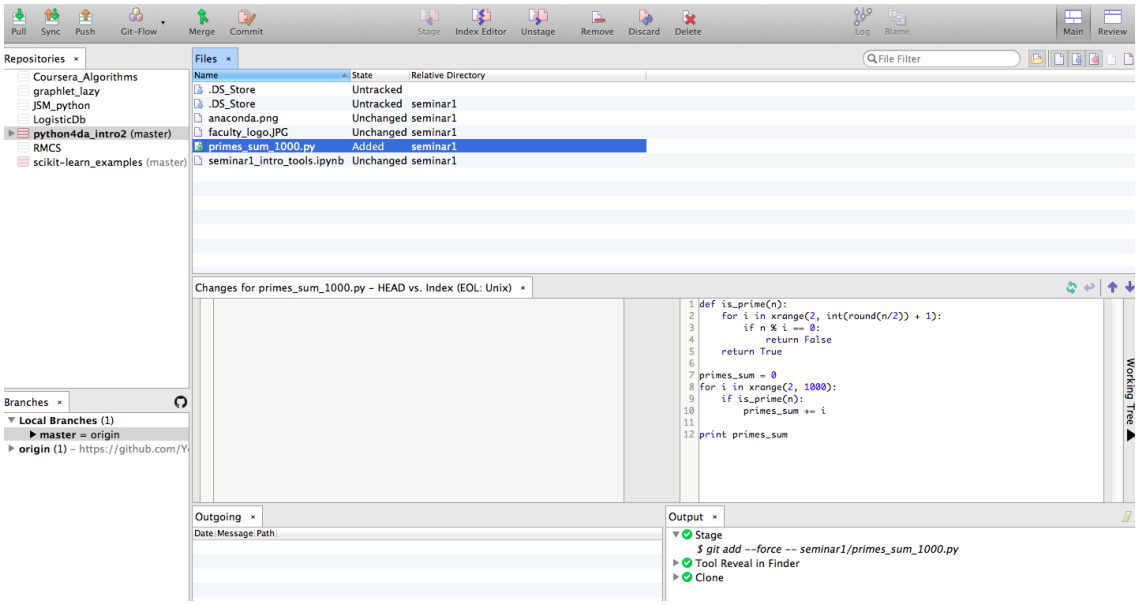
Внесем локально изменения (зафиксируем их в локальном репозитории)

>> git commit -m "add primes sum calculator"

Отправим изменения на сервер в ветку master

>> git push -u origin master

То же самое можно делать используя графический интерфейс, например SmartGit (http://www.syntevo.com/smartgit/).



При использовании Git и GitHub приходится иметь дело с разделением ветвей кода (branching), слиянием изменений (merging), восстановлением потерянного кода и другими вопросами, многие из которых рассмотрены в курсе (https://www.udacity.com/course/how-to-use-git-and-github--ud775) на Udacity. Введение (http://habrahabr.ru/post/125799/) в GitHub на русском на Хабрахабре.

Часть 5. Обзор возможностей сборки библиотек Anaconda

Сборка Anaconda включает очень много полезных для анализа данных библиотек. Посмотреть все установленные можно с помощью команды

>> conda list

Среди библиотек, которыми мы будем пользоваться в курсе:

- Numpy (http://numpy.org) - эффективные вычисления с векторами и матрицами, преобразования Фурье, линейная алгебра, интеграция с C/C++ и Fortran и прочее.
- SciPy (http://numpy.org) - научные вычисления. Методы оптимизации, интегрирования, модули обработки сигналов и изображений, статистика, линейная алгебра, сплайны, кластеризация и многое другое.
- Pandas (http://pandas.pydata.org/) - реализация эффективных структур для анализа данных, в том числе популярного аналога *DataFrame* из языка R. Предназначена для данных разной природы - матричных, панельных данных, временных рядов. Претендует на звание самого мощного и гибкого средства для анализа данных с открытым исходным кодом.
- Scikit-learn (http://scikit-learn.org/stable/) - реализация очень многих методов машинного обучения и интеллекуального анализа данных (data mining) с отличной документацией.
- matplotlib (http://http://matplotlib.org/) - для визаулизации данных, в основном двухмерная графика. Построена на принципах ООП, но имеет процедурный интерфейс *pylab*, который предоставляет аналоги команд *MATLAB*.

Часть 6. Регистрация на сайте дистанционного курса по информатике

Сдача и прием задач курса будет проходить на базе платформы МЦНМО.

Регистрация учетной записи на http://informatics.mccme.ru/ (http://informatics.mccme.ru/).

1) Необходимо зайти на сайт http://informatics.mccme.ru/ (http://informatics.mccme.ru/) и в боковом меню нажать ссылку *Регистрация*

2) На открывшейся странице необходимо ввести:

a. *Логин* и *Пароль* для авторизации в системе (придумайте и запомните),

b. Ваши *Фамилию*, *Имя*, *Отчество*, *Город* (Москва), *Страну* (Россия).

c. в поле *email* введите Вашу почту

d. в поле *Школа* укажите *MLClass*

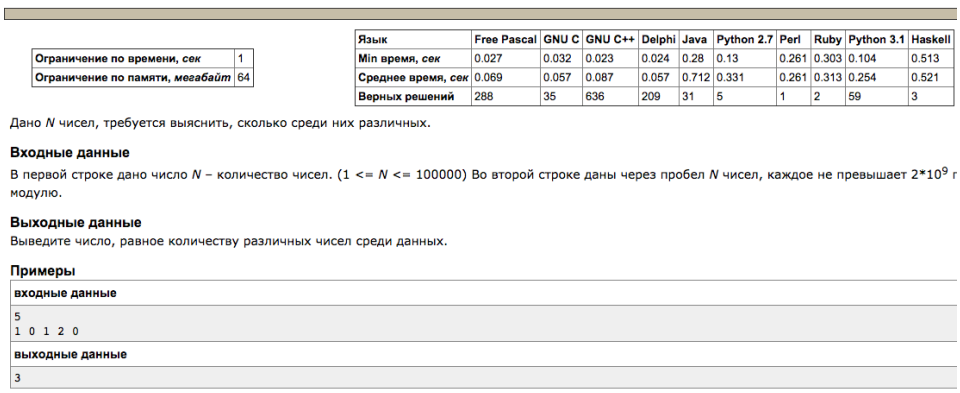
e. Класс 11 и год выпуска 2015

f. Поставьте галочку, что Вы не робот, и нажмите кнопку *Сохранить*

3) Далее в открывшемся окне введите Ваш логин и пароль и нажмите кнопку *Вход*.

4) После удачной авторизации Вы будете перенаправлены на страницу курсов, можете найти в списке Курсов “MLClass” или пройти сразу на страницу http://informatics.mccme.ru/course/view.php?id=1165 (http://informatics.mccme.ru/course/view.php?id=1165). По мере наступления практических занятий там будут появляться задачи.

Пример (http://informatics.mccme.ru/mod/statements/view.php?id=1129) одной из задач.



Ограничение по времени, *сек*	1
Ограничение по памяти, *мегабайт*	64

Язык	Free Pascal	GNU C	GNU C++	Delphi	Java	Python 2.7	Perl	Ruby	Python 3.1	Haskell
Min время, *сек*	0.027	0.032	0.023	0.024	0.28	0.13	0.261	0.303	0.104	0.513
Среднее время, *сек*	0.069	0.057	0.087	0.057	0.712	0.331	0.261	0.313	0.254	0.521
Верных решений	288	35	636	209	31	5	1	2	59	3

Дано *N* чисел, требуется выяснить, сколько среди них различных.

Входные данные

В первой строке дано число *N* – количество чисел. (1 <= *N* <= 100000) Во второй строке даны через пробел *N* чисел, каждое не превышает $2*10^9$ по модулю.

Выходные данные

Выведите число, равное количеству различных чисел среди данных.

Примеры

входные данные
5 1 0 1 2 0
выходные данные
3

Не только формулируется условие задачи, но и даются примеры входных и выходных данных, ограничения по времени и памяти, а также статистика времени выполнения кода на разных языках.

Домашние задания

После некоторых уроков Вы получите задачи для самостоятельного решения. Решенную задачу (код) необходимо загрузить в систему автоматического приема задач на http://informatics.mccme.ru/ (http://informatics.mccme.ru/).