АНАЛИЗА И ОБРАДА ПОДАТАКА ПРИМЕНОМ ПРОГРАМСКОГ ЈЕЗИКА PYTHON И ЊЕГОВИХ БИБЛИОТЕКА

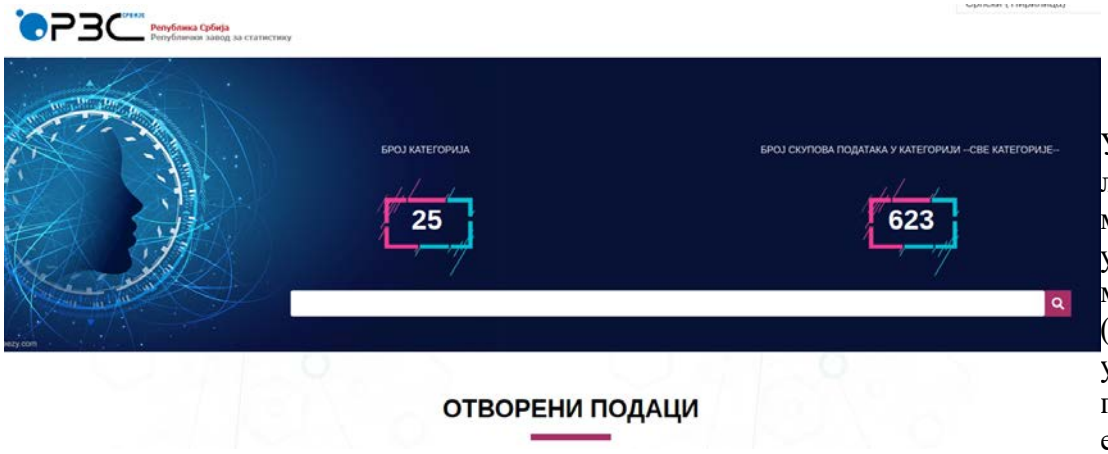
Милена Марић

О ПОДАЦИМА

Чињеница је да живимо у времену када смо са свих страна окружени подацима. У сваком тренутку настаје огромна количина различитих података. Можемо, слободно рећи да свако од нас свакога дана иза себе оставља такође велику количину података. Данас када постоје изузетно брзи рачунари, који могу да складиште све те силне податке, ваљало би искористити ове њихове бенефите како би се из свих тих података могао донети неки закључак који је нама од одређене користи.
Не треба занемарити да се око нас налази велика количина тзв. *отворених података* и да смо, уколико поседујемо одређена знања и вештине, на извору великих знања. Отворене податке можемо наћи на адреси opendata.stat.gov.rs.

Отворени подаци су (јавни) подаци који су доступни електронски у машински читљивом формату на начин да се могу директно користити за развој различитих апликација.

Отворени подаци представљају податке које прикупљају органи јавне власти као део њихове надлежности и уз употребу јавних средстава.



Умемо ли да манипулишемо (обрађујемо) податке?
Право је питање знамо ли шта нам подаци говоре и ако не знамо, можемо ли лако да савладамо вештину обраде и анализе података. Јасно је да се о некој изузетно прецизној обради и анализи података не може говорити без доброг познавања математике (статистике), али се неки почетни кораци могу направити.

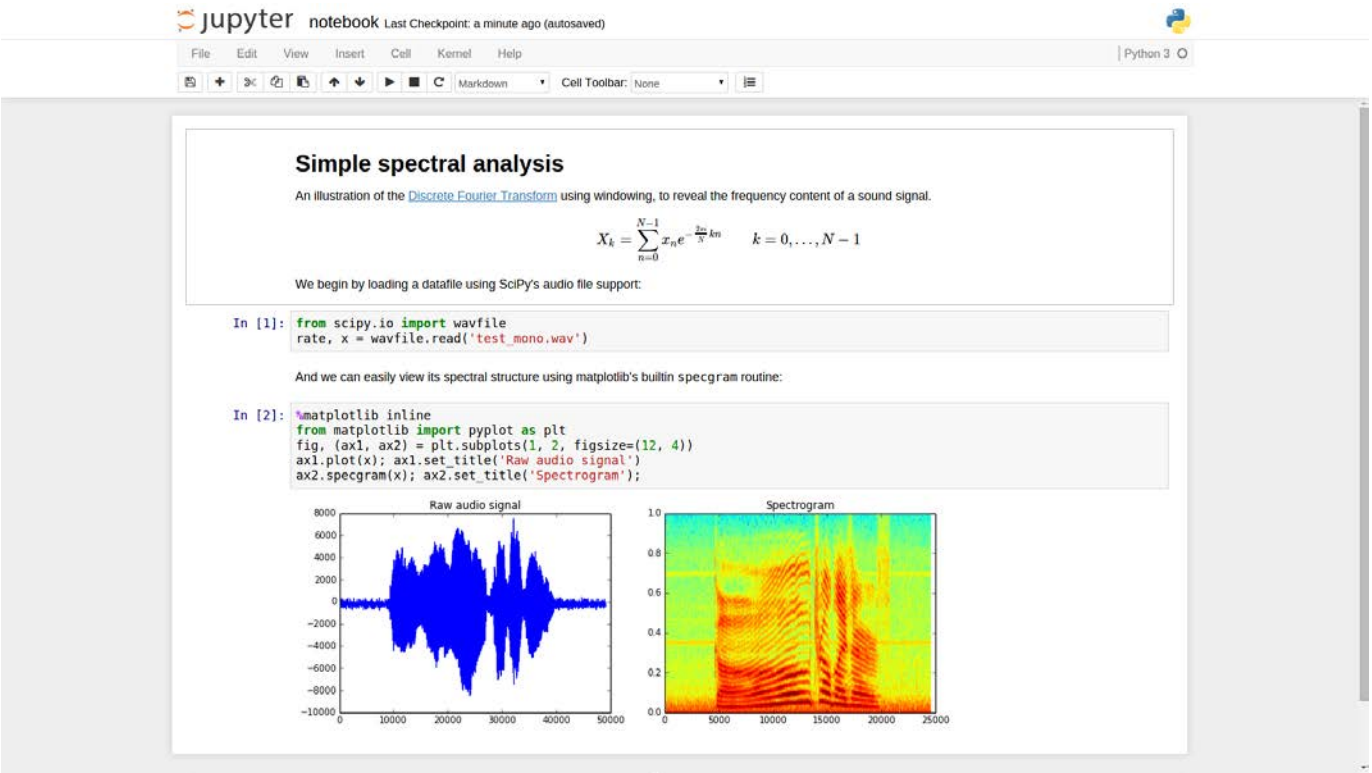
Циљ овог излагања је да да кратак увод у обраду и анализу података применом знања које ученик стиче у другом разреду гимназије. За ваљано овладавање овом области потребно је познавање основних елемената програмског језика Python.

Програмски језик Python је програмски језик који има робусне библиотеке које располажу функцијама са којима се лако могу обрађивати и представљати подаци. У даљем тексту биће речи о овим библиотекама. Такође, осврнућемо се и на интерактивне свеске, окружење Jupyter које такође служи за једноставнији рад приликом анализе и обраде података.

ПРЕДСТАВЉАЊЕ ПОДАТАКА

Подаци морају да буду представљени одређеним структурама података како бисмо могли да их обрађујемо. Можемо користити листе, низове, низове низова, речницима.
Још један од начина да се организују подаци је *табела*. Примена табеле нам је позната још од обраде података применом апликативног софтвера Excel за табеларна израчунавања. Овде треба напоменути да је Excel изузетно моћан софтвер за обраду података. Сада неко, са разлогом, може поставити питање зашто користимо програмски језик ако је Excel добар приликом обраде података. Један од одговора на ово питање може бити да је Excel комерцијалан софтвер, док је коришћење програмског језика сасвим бесплатно.

За представљање података користимо интерактивно окружење *Jupyter*.



Детаљно описан поступак преузимања и инсталирања радне свеске *Jupyter* можете наћи на порталу petlja.org. Такође, поред самог начина преузимања и инсталације интерактивног окружења *Jupyter* овде се могу наћи и сви значајни детаљи, а у вези са поменутим интерактивним свескама. Предност коришћења *Jupyter – a* је што у овиру једне интерактивне свеске имате све на једном месту. У простору где можете да видите табеле, графике, хистограме...можете да пишете и код.
Детаљно описан поступак преузимања и инсталирања радне свеске *Jupyter* можете наћи на порталу petlja.org. Такође, поред самог начина преузимања и инсталације интерактивног окружења *Jupyter* овде се могу наћи и сви значајни детаљи, а у вези са поменутим интерактивним свескама. Предност коришћења *Jupyter – a* је што у овиру једне интерактивне свеске имате све на једном месту. У простору где можете да видите табеле, графике, хистограме...можете да пишете и код.

Упознаћемо се са представљањем низова података и коришћењем библиотеке *mathplotlib.pyplot*.

2.3. Приказивање низова података

У наредној ћелији приказан је процењени број становника наше планете у разним историјским периодима, при чему је број људи исказан у милијардама:

In [7]:
```
godine = [1000,  1500, 1650, 1750, 1804, 1850, 1900, 1930, 1950, 1960, 1974,
1980, 1987, 1999, 2011, 2020, 2023, 2030, 2037, 2045, 2055, 2100]
ljudi  = [0.275, 0.45, 0.5,  0.7,  1,    1.2,  1.6,  2,    2.55, 3,    4,
4.5,  5,    6,    7,    7.8,  8,    8.5,  9,    9.5,  10,   11.2]
```

Желели бисмо да прикажемо ове податке у виду графикона. Пајтон има разне библиотеке за визуелизацију података, а ми ћемо користити библиотеку која се зове `matplotlib.pyplot`. Пошто је ово име веома дугачко и компликовано и пошто ће нам требати много функција из те библиотеке нећемо увозити функције једну по једну већ ћемо увести целу библиотеку и при томе јој дати краће име (такорећи, надимак) `plt`. (Енглеска реч *plot* значи "исцртати", а `plt` је скраћено од `plot`.)

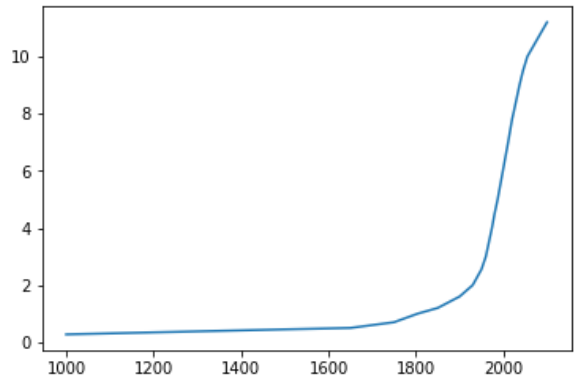
In [8]:
```
import matplotlib.pyplot as plt
```

Сада су нам све функције из ове библиотеке доступне. Важно је напоменути да *један* `import` *важи за целу радну свеску!* Дакле, довољно је библиотеку увести једном у радној свесци. Такође је важно рећи да ћемо у новој радној свесци морати поново да "импортујемо" библиотеке које нам требају за рад.

Применом ове библиотеке лако је податке представити и графиком. Наредни пасус приказује које смо то функције користили приликом писања кода за графичко представљање података.Упознаћемо се детаљно са функцијама:
Упознаћемо се детаљно са функцијама:

`plt.plot()`
`plt.show()`
`plt.close()`

```
In [9]: plt.plot(godine, ljudi)
        plt.show()
        plt.close()
```

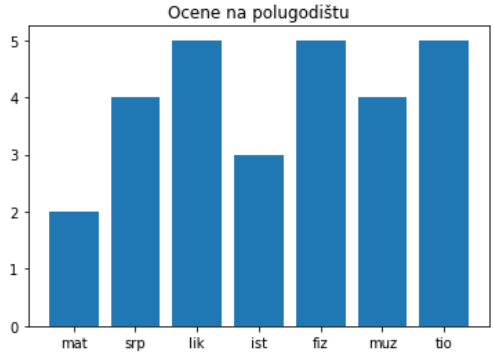


Податке лепо можемо представити хистограмом, те ћемо се упознати са функцијом `plt.bar()`.

```
In [2]: predmeti = ["mat", "srp", "lik", "ist", "fiz", "muz", "tio"]
        ocene    = [2,     4,     5,     3,     5,     4,     5    ]
```

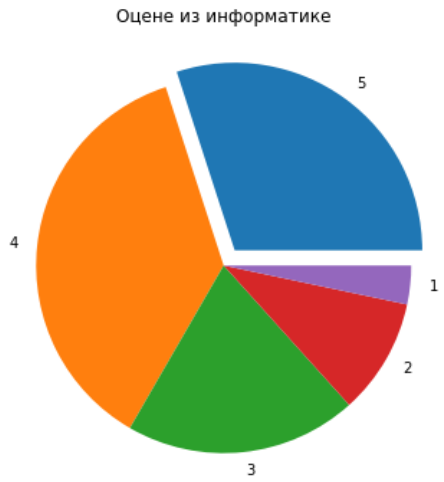
Графички их можемо представити низовима стубића користећи функцију `bar`. Ова врста дијаграма се на енглеском зове *bar chart* (дијаграм са стубићима), а ми их зовемо *хистограми*:

```
In [3]: plt.bar(predmeti, ocene)
        plt.title("Ocene na polugodištu")
        plt.show()
        plt.close()
```



Неке податке је много боље представити *секторским дијаграмима*. Зато ћемо се упознати са функцијом `plt.pie()`.

```
In [7]: frekvencije = [9,   11,  6,   3,   1]
        ocene       = ["5", "4", "3", "2", "1"]
        izmestanje  = [0.1, 0,   0,   0,   0]
        plt.figure(figsize=(6,6))
        plt.pie(frekvencije, labels=ocene, explode=izmestanje)
        plt.title("Оцене из информатике")
        plt.show()
        plt.close()
```



Подаци се могу представити и табеларно и зато ћемо се упознати са библиотеком *pandas* структуром података *Data Frames*. Такође, биће речи и о различитим фајловима и начинима како можемо увозити и извозити податке које обрађујемо.

ОБРАДА ПОДАТАКА

Приликом обраде података потребно је знати како манипулисати структуром података коју смо представили рецимо у Data Frame облику. За детаљне обраде потребно је научити нешто више у вези са индексирањем табеле, транспоновањем табеле, модификацијама табеле, као и записивањем табеле у датотеку. Такође, битно је знати извршити различита сортирања, филтрирања као и познавати тзв. Фреквенцијску анализу података.

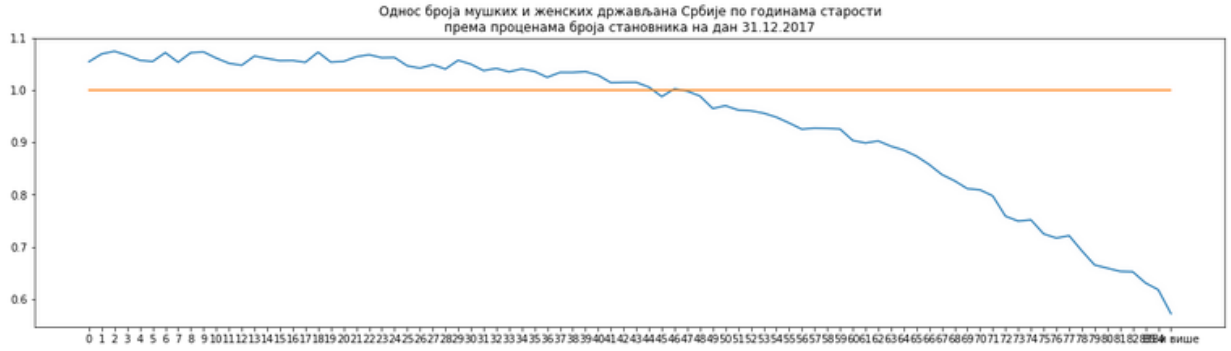
In [1]:
```
import pandas as pd
razred = [["Ana",      5, 3, 5, 2, 4, 5],
          ["Bojan",    5, 5, 5, 5, 5, 5],
          ["Vlada",    4, 5, 3, 4, 5, 4],
          ["Gordana", 5, 5, 5, 5, 5, 5],
          ["Dejan",    3, 4, 2, 3, 3, 4],
          ["Đorđe",    4, 5, 3, 4, 5, 4],
          ["Elena",    3, 3, 3, 4, 2, 3],
          ["Žaklina", 5, 5, 4, 5, 4, 5],
          ["Zoran",    4, 5, 4, 4, 3, 5],
          ["Ivana",    2, 2, 2, 2, 2, 5],
          ["Jasna",    3, 4, 5, 4, 5, 5]]
ocene = pd.DataFrame(razred)
ocene.columns=["Ime", "Informatika", "Engleski", "Matematika", "Fizika",
"Hemija", "Likovno"]
ocene1 = ocene.set_index("Ime")
ocene1
```

Out[1]:

	Informatika	Engleski	Matematika	Fizika	Hemija	Likovno
Ime						
Ana	5	3	5	2	4	5
Bojan	5	5	5	5	5	5
Vlada	4	5	3	4	5	4
Gordana	5	5	5	5	5	5
Dejan	3	4	2	3	3	4
Đorđe	4	5	3	4	5	4
Elena	3	3	3	4	2	3
Žaklina	5	5	4	5	4	5
Zoran	4	5	4	4	3	5
Ivana	2	2	2	2	2	5
Jasna	3	4	5	4	5	5

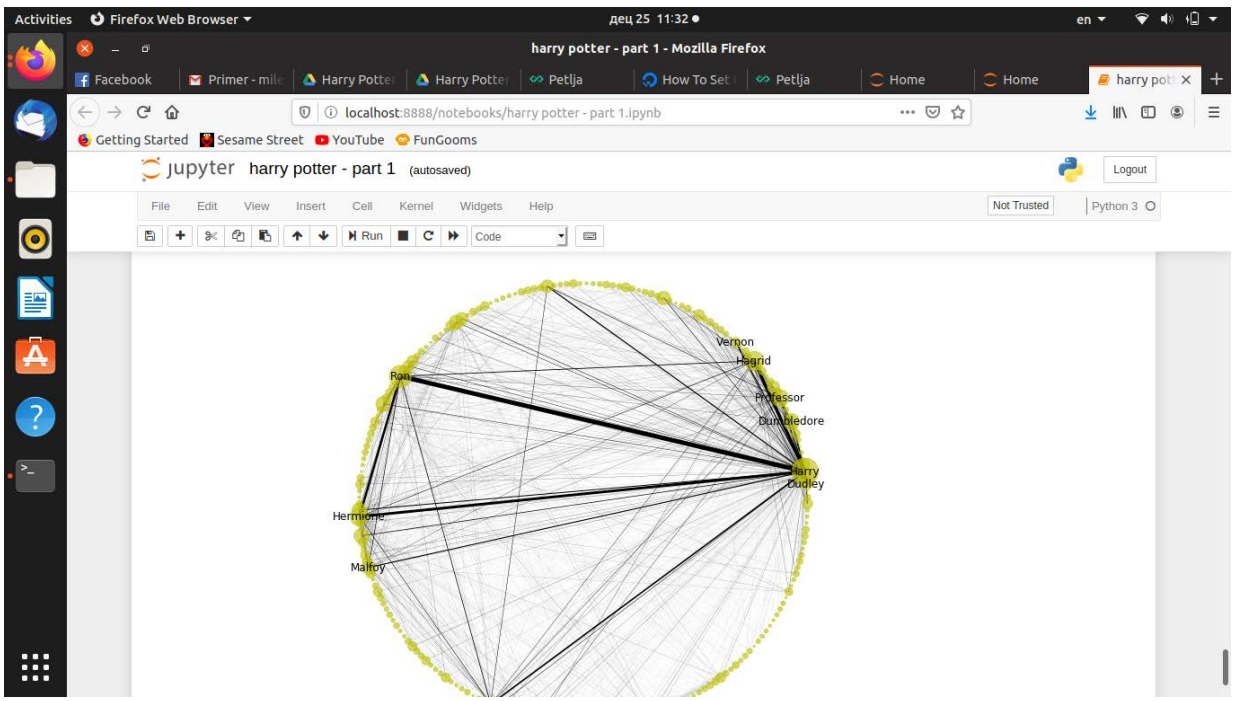
Обрађене податке често приказујемо дијаграмима када желимо да анализирамо резултате наше обраде.
За анализирање резултата обраде података користићемо добро позната математичка знања.

In [12]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(20,5))
plt.plot(stanovnistvo1.index, stanovnistvo1["М/Ж"])
plt.plot(stanovnistvo1.index, [1.0] * len(stanovnistvo1.index))
plt.title("Однос броја мушких и женских држављана Србије по годинама
старости\nпрема проценама броја становника на дан 31.12.2017")
plt.show()
plt.close()
```



АНАЛИЗА И ВИЗУЕЛНО ПРЕДСТАВЉАЊЕ ПОДАТАКА

Из скупа податак који представљају међусобне односе ликова у романима о добро познатом јунаку Харију потеру, можемо применом до сада научених својстава радне свеске и функција одређених библиотека прормског језика Python, да представимо међусобан однос главних јунака романа на следећи начин:



У оквиру предавања наставници ће самостално урадити неколико вежби на скуповима података који су отворени и који се могу преузети са интернета, а у циљу савладавања презентоване материје.