15. час: Учитавање табеларно представљених података из спољашњих извора

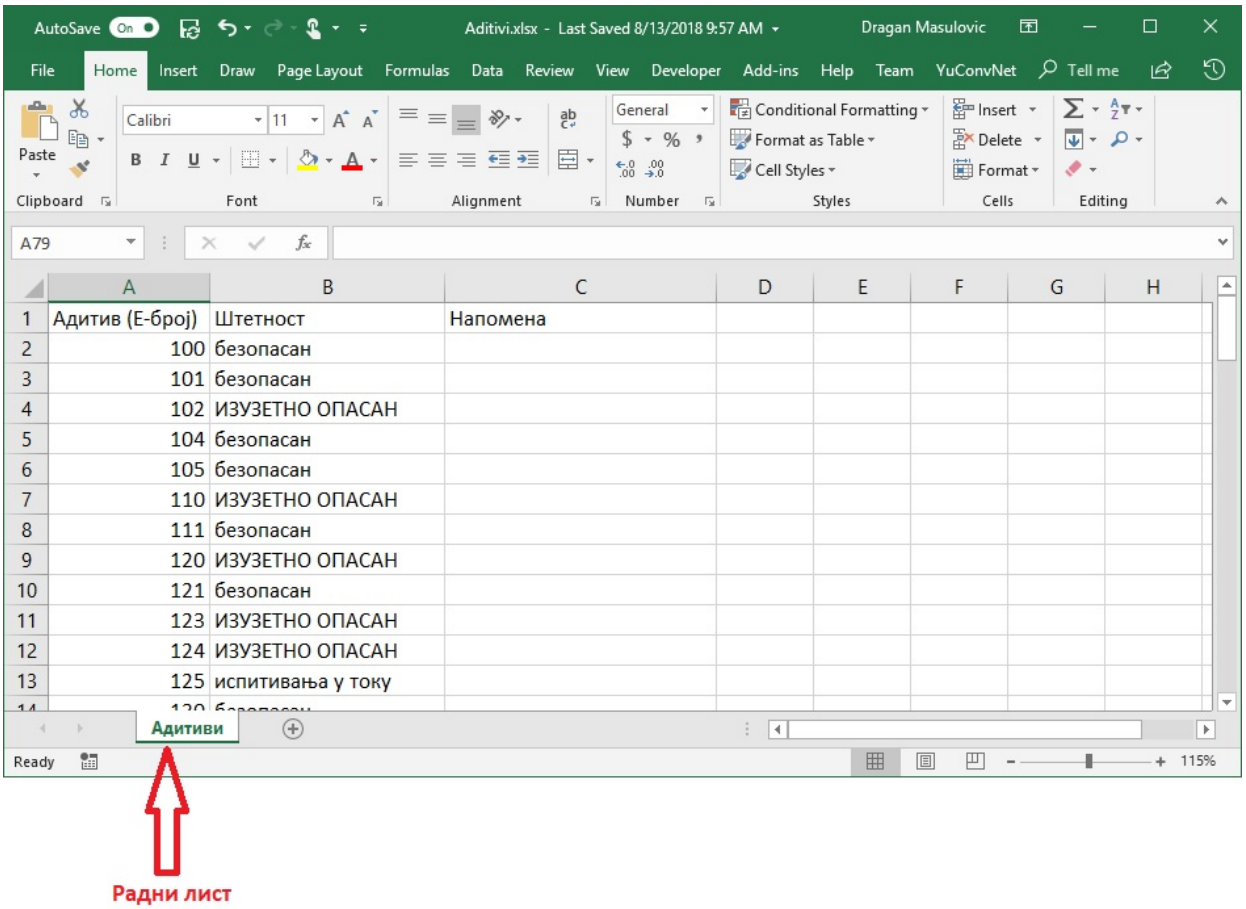
На овом часу ћемо говорити о:

1. Ексел датотеке
2. Он-лајн ресурси
3. Зашто Џупајтер а не Ексел?

15.1. Учитавање података из локалних Ексел датотека

Мајкрософтов Ексел (*Microsoft Excel*) представља један од најраспрострањенијих софтверских производа за обраду табеларно представљених података. (На крају овог часа рећи ћемо неколико речи о томе зашто смо се ми и поред тога одлучили за Џупајтер.) Библиотека *pandas* зато има функцију која може да учита податке представљене Ексел табелом.

Структура Ексел документа је релативно сложена јер у једном документу може да се налази више табела. Један Ексел документ се, зато, састоји из неколико *радних листова* (енгл. *work sheets*):



	A	B	C
1	Адитив (Е-број)	Штетност	Напомена
2	100	безопасан	
3	101	безопасан	
4	102	ИЗУЗЕТНО ОПАСАН	
5	104	безопасан	
6	105	безопасан	
7	110	ИЗУЗЕТНО ОПАСАН	
8	111	безопасан	
9	120	ИЗУЗЕТНО ОПАСАН	
10	121	безопасан	
11	123	ИЗУЗЕТНО ОПАСАН	
12	124	ИЗУЗЕТНО ОПАСАН	
13	125	испитивања у току	

па функцији за учитавање Ексле табеле поред имена датотеке треба дати и име радног листа са кога се учитава табела.

Уколико се не наведе име раднлог листа, функција ће учитати табелу из првог радног листа на који наиђе.

У следећој ћелији ћемо из датотеке *Aditivi.xlsx* која се налази у фолдеру *podaci* учитати табелу из (јединог) радног листа "Адитиви":

In [1]:
```
# изврши ову ћелију
import pandas as pd
aditivi = pd.read_excel("podaci/Aditivi.xlsx", sheet_name="Адитиви")
```

Ова датотека садржи податке о адитивима, што су супстанце које се користе у индустрији. Неки од њих се користе и у индустрији хране. (Подаци су преузети из уџбеника биологије за 8. разред.)

Ево првих неколико редова ове табеле:

In [6]:
```
# изврши ову ћелију
aditivi.head(15)
```

Out[6]:

	Адитив (Е-број)	Штетност	Напомена
0	100	безопасан	NaN
1	101	безопасан	NaN
2	102	ИЗУЗЕТНО ОПАСАН	NaN
3	104	безопасан	NaN
4	105	безопасан	NaN
5	110	ИЗУЗЕТНО ОПАСАН	NaN
6	111	безопасан	NaN
7	120	ИЗУЗЕТНО ОПАСАН	NaN
8	121	безопасан	NaN
9	123	ИЗУЗЕТНО ОПАСАН	NaN
10	124	ИЗУЗЕТНО ОПАСАН	NaN
11	125	испитивања у току	NaN
12	130	безопасан	NaN
13	131	штетан	може изазвати рак
14	132	безопасан	NaN

Видимо да су ћелије које су биле празне у Ексел табели овде добиле специјалну вредност *NaN* што је скраћеница од *not a number* (енгл. "није број"). Ово је специјална вредност која се користи да се открију потенцијалне грешеке које могу да настану приликом учитавања великих табела. У нашем случају празне ћелије у колони "Напомена" и треба да остану празне, па ћемо табелу учитати поново, с тим да ћемо "замолити Пајтон да искључи вештачку интелигенцију":

In [3]:
```
# изврши ову ћелију
aditivi = pd.read_excel("podaci/Aditivi.xlsx", sheet_name="Адитиви", na_filter=Fa
aditivi.head(15)
```

Out[3]:

	Адитив (Е-број)	Штетност	Напомена
0	100	безопасан	
1	101	безопасан	
2	102	ИЗУЗЕТНО ОПАСАН	
3	104	безопасан	
4	105	безопасан	
5	110	ИЗУЗЕТНО ОПАСАН	
6	111	безопасан	
7	120	ИЗУЗЕТНО ОПАСАН	
8	121	безопасан	
9	123	ИЗУЗЕТНО ОПАСАН	
10	124	ИЗУЗЕТНО ОПАСАН	
11	125	испитивања у току	
12	130	безопасан	
13	131	штетан	може изазвати рак
14	132	безопасан	

Аргумент `na_filter=False` каже функцији `read_excel` да празне ћелије остану празне и да у њих не уноси вредност *NaN*.

Направићемо сада фреквенцијску анализу ове табеле на основу штетности адитива.

In [9]:
```
# изврши ову ћелију
aditivi["Штетност"].value_counts()
```

Out[9]:
```
штетан               33
безопасан            29
испитивања у току    10
ИЗУЗЕТНО ОПАСАН       5
Name: Штетност, dtype: int64
```

Профилтрираћемо табелу да бисмо излистали адитиве који могу изазвати рак.

In [7]:
```
# изврши ову ћелију
aditivi[aditivi.Напомена == "може изазвати рак"]
```

Out[7]:

	Адитив (Е-број)	Штетност	Напомена
13	131	штетан	може изазвати рак
17	142	штетан	може изазвати рак
28	210	штетан	може изазвати рак
29	211	штетан	може изазвати рак
30	213	штетан	може изазвати рак
31	214	штетан	може изазвати рак
32	215	штетан	може изазвати рак
33	216	штетан	може изазвати рак
34	217	штетан	може изазвати рак
45	239	штетан	може изазвати рак
55	330	штетан	може изазвати рак

За крај, излистаћемо адитиве који су изузетно опасни или могу изазвати рак.

In [10]:
```
# изврши ову ћелију
aditivi[(aditivi.Напомена == "може изазвати рак") | (aditivi.Штетност == "ИЗУЗЕТН
```

Out[10]:

	Адитив (Е-број)	Штетност	Напомена
2	102	ИЗУЗЕТНО ОПАСАН	
5	110	ИЗУЗЕТНО ОПАСАН	
7	120	ИЗУЗЕТНО ОПАСАН	
9	123	ИЗУЗЕТНО ОПАСАН	
10	124	ИЗУЗЕТНО ОПАСАН	
13	131	штетан	може изазвати рак
17	142	штетан	може изазвати рак
28	210	штетан	може изазвати рак
29	211	штетан	може изазвати рак
30	213	штетан	може изазвати рак
31	214	штетан	може изазвати рак
32	215	штетан	може изазвати рак
33	216	штетан	може изазвати рак
34	217	штетан	може изазвати рак
45	239	штетан	може изазвати рак
55	330	штетан	може изазвати рак

15.2. Учитавање података из удаљених ресурса

Могуће је преузети и податке са удањених ресурса без потребе да се они прво пребаце на локалну машину. Да бисмо приступили податку који се налази на некој другој машини потребно је да обе машине имају приступ Интернету и да знамо тачну локацију податка на удањеној машини. Тачна локација било ког ресурса на Интернету је описана његовим *URL*-ом (од енгл. *Universal Resource Locator*, што значи "Универзални локатор ресурса").

На адреси

```
https://raw.githubusercontent.com/cs109/2014_data/master/countries.csv
```

се налази јавно доступан списак свих држава на свету. Ову табелу можемо лако учитати наредбом `read_csv` :

In [3]:
```
# изврши ову ћелију
drzave = pd.read_csv("https://raw.githubusercontent.com/cs109/2014_data/master/co
drzave.head(5)
```

Out[3]:

	Country	Region
0	Algeria	AFRICA
1	Angola	AFRICA
2	Benin	AFRICA
3	Botswana	AFRICA
4	Burkina	AFRICA

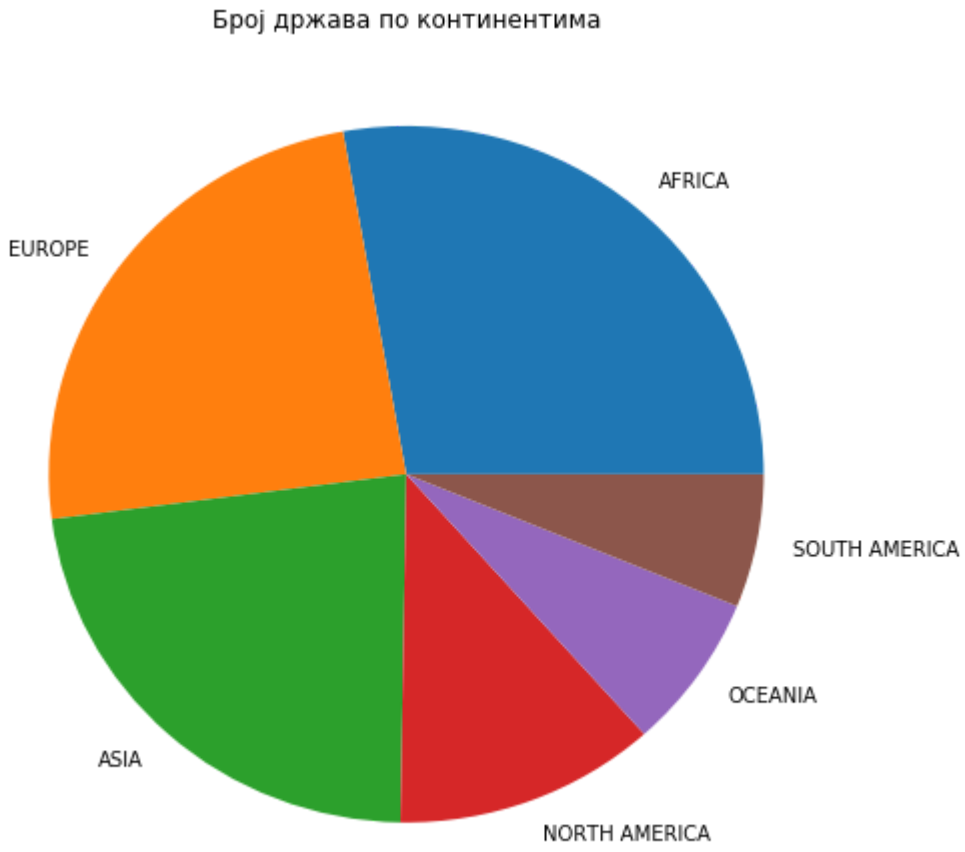
Број држава по континентима можемо видети овако:

In [4]:
```
# изврши ову ћелију
drzave["Region"].value_counts()
```

Out[4]:
```
AFRICA           54
EUROPE           47
ASIA             44
NORTH AMERICA    23
OCEANIA          14
SOUTH AMERICA    12
Name: Region, dtype: int64
```

Прикажимо број држава по континентима секторским дијаграмом:

In [9]:
```
# изврши ову ћелију
import matplotlib.pyplot as plt
po_kontinentima = drzave["Region"].value_counts()
plt.figure(figsize=(8,8))
plt.pie(po_kontinentima.values, labels=po_kontinentima.index)
plt.title("Број држава по континентима")
plt.show()
plt.close()
```



Помоћу наредбе `read_html` може се прочитати и табела директно из *HTML* кода неке веб странице. Рецимо, следећа наредба чита списак свих федералних јединица Сједињених Америчких Држава са одговарајуће странице Википедије:

In [41]:
```
# изврши ову ћелију
US = pd.read_html("https://simple.wikipedia.org/wiki/List_of_U.S._states", header
```

Наредба `read_html` враћа релативно сложену структуру, али табела коју желимо да видимо је прва у тој структури. Зато иза наредбе следи конструкт `[0]` који враћа прву компоненту сложене структуре. Аргумент `header=0` значи да прву врсту треба узети за заглавље табеле. Ево како изгледа табела:

In [42]:
```
# изврши ову ћелију
US
```

Out[42]:

	Sl no.	Abbreviation	State Name	Capital	Became a State
0	1	AL	Alabama	Montgomery	December 14, 1819
1	2	AK	Alaska	Juneau	January 3, 1959
2	3	AZ	Arizona	Phoenix	February 14, 1912
3	4	AR	Arkansas	Little Rock	June 15, 1836
4	5	CA	California	Sacramento	September 9, 1850
5	6	CO	Colorado	Denver	August 1, 1876
6	7	CT	Connecticut	Hartford	January 9, 1788
7	8	DE	Delaware	Dover	December 7, 1787
8	9	FL	Florida	Tallahassee	March 3, 1845
9	10	GA	Georgia	Atlanta	January 2, 1788
10	11	HI	Hawaii	Honolulu	August 21, 1959
11	12	ID	Idaho	Boise	July 3, 1890
12	13	IL	Illinois	Springfield	December 3, 1818
13	14	IN	Indiana	Indianapolis	December 11, 1816
14	15	IA	Iowa	Des Moines	December 28, 1846
15	16	KS	Kansas	Topeka	January 29, 1861
16	17	KY	Kentucky	Frankfort	June 1, 1792
17	18	LA	Louisiana	Baton Rouge	April 30, 1812
18	19	ME	Maine	Augusta	March 15, 1820
19	20	MD	Maryland	Annapolis	April 28, 1788
20	21	MA	Massachusetts	Boston	February 6, 1788
21	22	MI	Michigan	Lansing	January 26, 1837
22	23	MN	Minnesota	Saint Paul	May 11, 1858
23	24	MS	Mississippi	Jackson	December 10, 1817
24	25	MO	Missouri	Jefferson City	August 10, 1821
25	26	MT	Montana	Helena	November 8, 1889
26	27	NE	Nebraska	Lincoln	March 1, 1867
27	28	NV	Nevada	Carson City	October 31, 1864
28	29	NH	New Hampshire	Concord	June 21, 1788
29	30	NJ	New Jersey	Trenton	December 18, 1787
30	31	NM	New Mexico	Santa Fe	January 6, 1912
31	32	NY	New York	Albany	July 26, 1788
32	33	NC	North Carolina	Raleigh	November 21, 1789
33	34	ND	North Dakota	Bismarck	November 2, 1889

	Sl no.	Abbreviation	State Name	Capital	Became a State
34	35	OH	Ohio	Columbus	March 1, 1803
35	36	OK	Oklahoma	Oklahoma City	November 16, 1907
36	37	OR	Oregon	Salem	February 14, 1859
37	38	PA	Pennsylvania	Harrisburg	December 12, 1787
38	39	RI	Rhode Island	Providence	May 19, 1790
39	40	SC	South Carolina	Columbia	May 23, 1788
40	41	SD	South Dakota	Pierre	November 2, 1889
41	42	TN	Tennessee	Nashville	June 1, 1796
42	43	TX	Texas	Austin	December 29, 1845
43	44	UT	Utah	Salt Lake City	January 4, 1896
44	45	VT	Vermont	Montpelier	March 4, 1791
45	46	VA	Virginia	Richmond	June 25, 1788
46	47	WA	Washington	Olympia	November 11, 1889
47	48	WV	West Virginia	Charleston	June 20, 1863
48	49	WI	Wisconsin	Madison	May 29, 1848
49	50	WY	Wyoming	Cheyenne	July 10, 1890

15.3. Зашто Џупајтер, а не Ексел

Мајкрософтов Ексел (*Microsoft Excel*) представља један од најраспрострањенијих софтверских производа за обраду табеларно представљених података, па се природно намеће питање зашто овај курс није организован око Ексела. Разлога има много, а навешћемо три најважнија.

Цена. За разлику од Ексела који је комерцијални производ и који мора да се купи да би могао легално да се користи, Пајтон, све његове библиотеке и Џупајтер (као радно окружење за Пајтон) су *бесплатни*. Свако може без икакве накнаде да инсталира Пајтон и Џупајтер и да их користи за личне потребе и за образовне потребе.

Флексибилност. Пајтон долази са веома великим бројем библиотека које су развијане за потребе ефикасне обраде великих количина података. Све те библиотеке су доступне из Џупајтера. Ако се за коју годину појави нека нова библиотека која нуди нове могућности, можемо је лако и брзо увести у Џупајтер и користити.

Слично Џупајтеру, Ексел подржава писање мањих програмских фрагмената, али у програмском језику *Visual Basic for Applications*. За разлику од Џупајтера, нове функционалности се не дистрибуирају кроз библиотеке функција које се просто додају систему, већ свака нова функционалност изискује инсталацију нове верзије програма.

Континуитет. На крају, Пајтон смо већ учили претходне две године. Док би увођење у *Visual Basic for Applications* трајало дуже и тиме би се изгубило на континуитету, окружење засновано на Пајтону као што је Џупајтер омогућује да се одмах постави фокус на обраду и визуелизацију података.

15.4. Задаци

Задатак 1. У табели `podaci/SO2.xlsx` налазе се резултати мерења концентрације сумпор-диоксида у 2017. години у неким градовима Србије. Табела има четири колоне:

- МернаСтаница = Мерна станица
- СГВ = Средња годишња вредност у микрограмима по кубном метру
- БД125 = Број дана са више од 125 микрограма по кубном метру
- МДВ = Максимална дневна вредност у микрограмима по кубном метру

(а) Учитати ову табелу у структуру података *DataFrame*.

(б) Сортирати подаке по колони МДВ и приказати вредности у овој колони линијским дијаграмом.

(в) Издвојити из табеле оне редове код којих је вредност у колони БД125 већа од 0.

Задатак 2. На адреси

```
https://raw.githubusercontent.com/resbaz/r-novice-gapminder-files/master/
data/gapminder-FiveYearData.csv
```

се налази јавно доступна табела са списком држава света и неким параметрима економског развоја тих држава праћеним у интервалима од 5 година.

Табела има следеће колоне:

- country = држава
- year = година на коју се односе подаци
- pop = број становника (енгл. *population*)
- continent = континент
- lifeExp = очекивани животни век у годинама (енгл. *life expextancy*)
- gdpPercap = БДП по глави становника у америчким доларима (енгл. *GDP per capitem*)

(а) Учитати ову табелу у структуру података *DataFrame*.

(б) У нову табелу издвојити податке који се односе на Србију (Упутство: `tabela[tabela.country == "Serbia"]`)

(в) Приказати линијским дијаграмом како се мењао очекивани животни век грађана Србије за године за које постоје подаци у табели.

(г) Приказати хистограмом како се мењао БДП по глави становника Србије за године за које постоје подаци у табели.

In [ ]: