Name: Viviyan Richards W

Roll no:205229133

Lab_Pandas_Indexing_and_Computations

0.1 Pandas Indexing and Selection Lab

0.2 Simple Series and DataFrames

Create a Series to store Temperature values for 1 week

In [82]:
```
import pandas as pd
```

In [2]:
```
 temperature_trichy = pd.Series([40.2, 39.8, 36.3, 39.1, 41.3, 32.9, 36.6])
```

In [3]:
```
# show temperature values
temperature_trichy
```

Out[3]:
```
0    40.2
1    39.8
2    36.3
3    39.1
4    41.3
5    32.9
6    36.6
dtype: float64
```

In [4]:
```
 # What is the weather on 2nd day?
temperature_trichy[1]
```

Out[4]:
```
39.8
```

In [5]:
```
# Find all days and temperature where temperature over 40.0 degree Celsius
temperature_trichy[temperature_trichy > 40.0]
```

Out[5]:
```
0    40.2
4    41.3
dtype: float64
```

```
In [7]: # Find only day, not also temperature
        temperature_trichy[temperature_trichy > 40.0].index

Out[7]: Int64Index([0, 4], dtype='int64')
```

0.2.1 Create a Dataframe for student details from List

```
In [9]:  students = [['DS01', 'Rex', '1msc'], ['DS02', 'peter', '2msc'], ['CS01', 'ann',
        df_stud = pd.DataFrame(students, columns=['rollno', 'name', 'class']) # row index
```

```
In [10]: df_stud
```

Out[10]:

	rollno	name	class
0	DS01	Rex	1msc
1	DS02	peter	2msc
2	CS01	ann	3bsc

Display all column names

```
In [11]:  df_stud.columns

Out[11]: Index(['rollno', 'name', 'class'], dtype='object')
```

Add a new column address

```
In [13]: address = ['Delhi', 'Bangalore', 'Chennai']
         df_stud['address'] = address
```

```
In [14]: df_stud
```

Out[14]:

	rollno	name	class	address
0	DS01	Rex	1msc	Delhi
1	DS02	peter	2msc	Bangalore
2	CS01	ann	3bsc	Chennai

0.2.2 Create a Dataframe for Phone book from Dictionary

In [86]:
```
phonebook = {'name':['rex','sam','peter'],'phone':['9942002764', '9932176542','98
df_phonebook = pd.DataFrame(phonebook, columns=['name','mobile', 'email'])
```

In [87]:
```
df_phonebook
```

Out[87]:

	name	mobile	email
0	rex	NaN	rex@abc.com
1	sam	NaN	sam@xyz.com
2	peter	NaN	ann@bhc.com

0.3 Exploratory Data Analysis on Video Game Review Dataset

Import ign.csv dataset

In [20]:
```
reviews = pd.read_csv("ign.csv")
```

Show top-5 rows

In [21]:
```
reviews.head()
```

Out[21]:

	Unnamed: 0	score_phrase	title	url	platform	score	genre	edito
0	0	Amazing	LittleBigPlanet PS Vita	/games/littlebigplanet-vita/vita-98907	PlayStation Vita	9.0	Platformer	
1	1	Amazing	LittleBigPlanet PS Vita -- Marvel Super Hero E...	/games/littlebigplanet-ps-vita-marvel-super-he...	PlayStation Vita	9.0	Platformer	
2	2	Great	Splice: Tree of Life	/games/splice/ipad-141070	iPad	8.5	Puzzle	
3	3	Great	NHL 13	/games/nhl-13/xbox-360-128182	Xbox 360	8.5	Sports	
4	4	Great	NHL 13	/games/nhl-13/ps3-128181	PlayStation 3	8.5	Sports	

Show bottom 3 rows

In [22]:
```
reviews.tail(3)
```

Out[22]:

	Unnamed: 0	score_phrase	title	url	platform	score	genre	editors
18622	18622	Mediocre	Star Ocean: Integrity and Faithlessness	/games/star-ocean-5/ps4-20035681	PlayStation 4	5.8	RPG	
18623	18623	Masterpiece	Inside	/games/inside-playdead/xbox-one-121435	Xbox One	10.0	Adventure	
18624	18624	Masterpiece	Inside	/games/inside-playdead/pc-20055740	PC	10.0	Adventure	

How many rows and columns here?

In [23]:
```
reviews.shape
```

Out[23]:
```
(18625, 11)
```

What are the datatypes?

In [24]:
```
reviews.dtypes
```

Out[24]:
```
Unnamed: 0          int64
score_phrase       object
title              object
url                object
platform           object
score             float64
genre              object
editors_choice     object
release_year        int64
release_month       int64
release_day         int64
dtype: object
```

0.3.1 Selecting Columns

#### Select a single column, say title

In [25]:
```
reviews['title'].tail()
```

Out[25]:
```
18620                  Tokyo Mirage Sessions #FE
18621           LEGO Star Wars: The Force Awakens
18622    Star Ocean: Integrity and Faithlessness
18623                                     Inside
18624                                     Inside
Name: title, dtype: object
```

Select multiple columns, title and genre

In [27]:
```
reviews[['title', 'genre']].head(10)
```

Out[27]:

	title	genre
0	LittleBigPlanet PS Vita	Platformer
1	LittleBigPlanet PS Vita -- Marvel Super Hero E...	Platformer
2	Splice: Tree of Life	Puzzle
3	NHL 13	Sports
4	NHL 13	Sports
5	Total War Battles: Shogun	Strategy
6	Double Dragon: Neon	Fighting
7	Guild Wars 2	RPG
8	Double Dragon: Neon	Fighting
9	Total War Battles: Shogun	Strategy

0.3.2 Selection using Positions

Select top-5 rows and all columns, same as head()

In [28]:
```
reviews.iloc[:5, :]
```

Out[28]:

	Unnamed: 0	score_phrase	title	url	platform	score	genre	edito
0	0	Amazing	LittleBigPlanet PS Vita	/games/littlebigplanet-vita/vita-98907	PlayStation Vita	9.0	Platformer	
1	1	Amazing	LittleBigPlanet PS Vita -- Marvel Super Hero E...	/games/littlebigplanet-ps-vita-marvel-super-he...	PlayStation Vita	9.0	Platformer	
2	2	Great	Splice: Tree of Life	/games/splice/ipad-141070	iPad	8.5	Puzzle	
3	3	Great	NHL 13	/games/nhl-13/xbox-360-128182	Xbox 360	8.5	Sports	
4	4	Great	NHL 13	/games/nhl-13/ps3-128181	PlayStation 3	8.5	Sports	

Select rows from position 5 onwards, and columns from position 5 onwards.

In [94]:
```
reviews.iloc[5:, 5:]
```

Out[94]:

	genre	editors_choice	release_year	release_month	release_day
5	Strategy	N	2012	9	11
6	Fighting	N	2012	9	11
7	RPG	Y	2012	9	11
8	Fighting	N	2012	9	11
9	Strategy	N	2012	9	11
10	Fighting	N	2012	9	11
11	Fighting	N	2012	9	11
12	NaN	N	2012	9	10
13	Action, Adventure	Y	2012	9	7
14	Action, Adventure	Y	2012	9	7
15	Adventure	N	2012	9	6
16	Adventure	N	2012	9	6
17	Action	N	2012	9	5
18	Action, Adventure	N	2012	9	3
19	Fighting	N	2012	9	3
20	Fighting	N	2012	9	3
21	RPG	N	2012	8	31
22	RPG	N	2012	8	31
23	RPG	N	2012	8	31
24	Action, RPG	Y	2012	8	31
25	Shooter	N	2012	8	30
26	Action, RPG	Y	2012	8	30
27	Shooter	N	2012	8	29
28	Adventure	N	2012	8	29
29	Action, RPG	N	2012	8	28
30	Adventure	Y	2012	8	28
31	RPG	Y	2012	10	4
32	Platformer	N	2012	10	4
33	RPG	Y	2012	10	3
34	Action	N	2012	10	3
...	...	...	...	...	...
18595	Action	N	2016	7	16
18596	Action, Adventure	N	2016	7	14
18597	Shooter, Adventure	N	2016	7	13
18598	Action	N	2016	7	13

	genre	editors_choice	release_year	release_month	release_day
18599	Battle	N	2016	7	13
18600	Shooter	N	2016	8	19
18601	Shooter	N	2016	8	19
18602	Shooter	N	2016	8	19
18603	Platformer	N	2016	8	18
18604	Sports	N	2016	8	17
18605	Adventure	N	2016	8	16
18606	Strategy	N	2016	8	4
18607	Battle	N	2016	7	13
18608	Racing, Action	N	2016	7	13
18609	Action	N	2016	7	12
18610	Adventure	N	2016	7	12
18611	Shooter	N	2016	7	6
18612	Puzzle	N	2016	7	6
18613	Strategy	N	2016	7	1
18614	Adventure	Y	2016	6	29
18615	Adventure	Y	2016	6	29
18616	Adventure	N	2016	8	2
18617	Adventure	N	2016	8	2
18618	Action	Y	2016	7	28
18619	Puzzle, Action	N	2016	7	28
18620	RPG	N	2016	6	29
18621	Action, Adventure	Y	2016	6	29
18622	RPG	N	2016	6	28
18623	Adventure	Y	2016	6	28
18624	Adventure	Y	2016	6	28

18620 rows × 5 columns

Select the first column, and all of the rows for the column

In [96]:
```
reviews.iloc[:, :1]
```

Out[96]:

	score_phrase
0	Amazing
1	Amazing
2	Great
3	Great
4	Great
5	Good
6	Awful
7	Amazing
8	Awful
9	Good
10	Good
11	Good
12	Good
13	Amazing
14	Amazing
15	Okay
16	Okay
17	Great
18	Mediocre
19	Good
20	Good
21	Good
22	Good
23	Good
24	Amazing
25	Good
26	Amazing
27	Good
28	Great
29	Okay
...	...
18595	Bad
18596	Okay
18597	Bad
18598	Okay

	score_phrase
18599	Good
18600	Good
18601	Good
18602	Good
18603	Good
18604	Great
18605	Okay
18606	Okay
18607	Good
18608	Mediocre
18609	Great
18610	Okay
18611	Mediocre
18612	Good
18613	Great
18614	Amazing
18615	Amazing
18616	Good
18617	Great
18618	Amazing
18619	Good
18620	Good
18621	Amazing
18622	Mediocre
18623	Masterpiece
18624	Masterpiece

18625 rows × 1 columns

the 10th row, and all of the columns for that row

In [97]:
```
reviews.iloc[10:, :]
```

Out[97]:

	score_phrase	title	url	platform	score	genre	editor
10	Good	Tekken Tag Tournament 2	/games/tekken-tag-tournament-2/ps3-124584	PlayStation 3	7.5	Fighting	
11	Good	Tekken Tag Tournament 2	/games/tekken-tag-tournament-2/xbox-360-124581	Xbox 360	7.5	Fighting	
12	Good	Wild Blood	/games/wild-blood/iphone-139363	iPhone	7.0	NaN	
13	Amazing	Mark of the Ninja	/games/mark-of-the-ninja-135615/xbox-360-129276	Xbox 360	9.0	Action, Adventure	
14	Amazing	Mark of the Ninja	/games/mark-of-the-ninja-135615/pc-143761	PC	9.0	Action, Adventure	
15	Okay	Home: A Unique Horror Adventure	/games/home-a-unique-horror-adventure/mac-2001...	Macintosh	6.5	Adventure	
		Home: A	/games/home-a-unique-				

First column is not useful. So remove it

In [29]:
```
reviews = reviews.iloc[:,1:]
reviews.head()
```

Out[29]:

	score_phrase	title	url	platform	score	genre	editors_choice
0	Amazing	LittleBigPlanet PS Vita	/games/littlebigplanet-vita/vita-98907	PlayStation Vita	9.0	Platformer	Y
1	Amazing	LittleBigPlanet PS Vita -- Marvel Super Hero E...	/games/littlebigplanet-ps-vita-marvel-super-he...	PlayStation Vita	9.0	Platformer	Y
2	Great	Splice: Tree of Life	/games/splice/ipad-141070	iPad	8.5	Puzzle	N
3	Great	NHL 13	/games/nhl-13/xbox-360-128182	Xbox 360	8.5	Sports	N
4	Great	NHL 13	/games/nhl-13/ps3-128181	PlayStation 3	8.5	Sports	N

0.3.3 Selection using Row and Column Labels

We have already created students dataframe as below. Let us access name column with loc()

In [30]:
```
 students = [['DS01', 'Rex', '1msc'], ['DS02', 'peter', '2msc'], ['CS01', 'ann',
df_stud = pd.DataFrame(students, columns=['rollno', 'name', 'class'])
```

In [31]:
```
df_stud
```

Out[31]:

	rollno	name	class
0	DS01	Rex	1msc
1	DS02	peter	2msc
2	CS01	ann	3bsc

In [32]:
```
df_stud.loc[:, "name"]
```

Out[32]:
```
0      Rex
1    peter
2      ann
Name: name, dtype: object
```

Let us come back to our reviews. Display the first five rows of reviews using the loc method

In [33]:
```
reviews.loc[0:5,:] # here 0:5 are labels, not integer values
```

Out[33]:

	score_phrase	title	url	platform	score	genre	editors_choice
0	Amazing	LittleBigPlanet PS Vita	/games/littlebigplanet-vita/vita-98907	PlayStation Vita	9.0	Platformer	Y
1	Amazing	LittleBigPlanet PS Vita -- Marvel Super Hero E...	/games/littlebigplanet-ps-vita-marvel-super-he...	PlayStation Vita	9.0	Platformer	Y
2	Great	Splice: Tree of Life	/games/splice/ipad-141070	iPad	8.5	Puzzle	N
3	Great	NHL 13	/games/nhl-13/xbox-360-128182	Xbox 360	8.5	Sports	N
4	Great	NHL 13	/games/nhl-13/ps3-128181	PlayStation 3	8.5	Sports	N
5	Good	Total War Battles: Shogun	/games/total-war-battles-shogun/mac-142565	Macintosh	7.0	Strategy	N

Select score_phrase column

In [35]:
```
#reviews.loc[:, 'score_phrase']
reviews.loc[:, 'score_phrase'].head()
```

Out[35]:
```
0    Amazing
1    Amazing
2      Great
3      Great
4      Great
Name: score_phrase, dtype: object
```

In [36]:
```
reviews['score_phrase'].head(10) # it is also same
```

Out[36]:
```
0    Amazing
1    Amazing
2      Great
3      Great
4      Great
5       Good
6      Awful
7    Amazing
8      Awful
9       Good
Name: score_phrase, dtype: object
```

Select rows from 5 to 15

In [37]:
```
some_reviews = reviews.iloc[5:15,]
some_reviews.head()
```

Out[37]:

	score_phrase	title	url	platform	score	genre	editors_choice	release_year
5	Good	Total War Battles: Shogun	/games/total-war-battles-shogun/mac-142565	Macintosh	7.0	Strategy	N	2012
6	Awful	Double Dragon: Neon	/games/double-dragon-neon/xbox-360-131320	Xbox 360	3.0	Fighting	N	2012
7	Amazing	Guild Wars 2	/games/guild-wars-2/pc-896298	PC	9.0	RPG	Y	2012
8	Awful	Double Dragon: Neon	/games/double-dragon-neon/ps3-131321	PlayStation 3	3.0	Fighting	N	2012
9	Good	Total War Battles: Shogun	/games/total-war-battles-shogun/pc-142564	PC	7.0	Strategy	N	2012

Select scores of first 3 rows from score_phrase

df

In [38]:
```
some_reviews.loc[5:7, 'score'] # scores of first 3 rows
```

Out[38]:
```
5    7.0
6    3.0
7    9.0
Name: score, dtype: float64
```

Select “score”, “genre”, and “release_year” columns

In [39]:
```
 #reviews[["score", "genre", "release_year"]]
reviews[["score", "genre", "release_year"]].head()
```

Out[39]:

	score	genre	release_year
0	9.0	Platformer	2012
1	9.0	Platformer	2012
2	8.5	Puzzle	2012
3	8.5	Sports	2012
4	8.5	Sports	2012

In [40]:
```
# every column is a Series
type(reviews["score"])
```

Out[40]:
```
pandas.core.series.Series
```

0.3.4 Aggregate Columns

Find mean value of score column

In [41]:
```
reviews['score'].mean()
```

Out[41]:
```
6.950459060402666
```

Find mean value of all numeric columns

In [44]:
```
reviews.mean()
```

Out[44]:
```
score                6.950459
release_year      2006.515329
release_month        7.138470
release_day         15.603866
dtype: float64
```

Find mean value for each numeric column

In [45]:
```
 reviews.mean(axis=0)
```

Out[45]:
```
score                6.950459
release_year      2006.515329
release_month        7.138470
release_day         15.603866
dtype: float64
```

Find mean value for each row containing numeric values

In [46]:
```
#reviews.mean(axis=1)
reviews.mean(axis=1).head()
```

Out[46]:
```
0    510.500
1    510.500
2    510.375
3    510.125
4    510.125
dtype: float64
```

Find lowest, highest, median, standard deviation of a column

In [47]:
```
reviews['score'].median()
```

Out[47]:
```
7.3
```

In [48]:
```
reviews['score'].max()
```

Out[48]:
```
10.0
```

In [49]:
```
reviews['score'].std()
```

Out[49]:
```
1.7117358608045874
```

In [50]:
```
reviews['score'].count()
```

Out[50]:
```
18625
```

Describe() gives summary

In [51]:
```
reviews.describe() #gives summary of all numeric columns by default
```

Out[51]:

	score	release_year	release_month	release_day
count	18625.000000	18625.000000	18625.00000	18625.000000
mean	6.950459	2006.515329	7.13847	15.603866
std	1.711736	4.587529	3.47671	8.690128
min	0.500000	1970.000000	1.00000	1.000000
25%	6.000000	2003.000000	4.00000	8.000000
50%	7.300000	2007.000000	8.00000	16.000000
75%	8.200000	2010.000000	10.00000	23.000000
max	10.000000	2016.000000	12.00000	31.000000

check if review score has any correlation with other columns

In [52]:
```
reviews.corr()
```

Out[52]:

	score	release_year	release_month	release_day
score	1.000000	0.062716	0.007632	0.020079
release_year	0.062716	1.000000	-0.115515	0.016867
release_month	0.007632	-0.115515	1.000000	-0.067964
release_day	0.020079	0.016867	-0.067964	1.000000

Review score has no correlation with other features. So, release timing doesn’t linearly

vrelate to review score

0.3.5 Math Operations on DF columns

Divide score by 2

In [53]:
```
#reviews['score'] / 2
(reviews['score'] / 2).head()
```

Out[53]:
```
0    4.50
1    4.50
2    4.25
3    4.25
4    4.25
Name: score, dtype: float64
```

0.3.6 Boolean Indexing in Pandas

Select all video games whose review score > 7

In [55]:
```
score_filter = reviews["score"] > 7
#score_filter
score_filter.head()
```

Out[55]:
```
0    True
1    True
2    True
3    True
4    True
Name: score, dtype: bool
```

In [56]:
```
filtered_reviews = reviews[score_filter]
filtered_reviews.head()
```

Out[56]:

	score_phrase	title	url	platform	score	genre	editors_choice
0	Amazing	LittleBigPlanet PS Vita	/games/littlebigplanet-vita/vita-98907	PlayStation Vita	9.0	Platformer	Y
1	Amazing	LittleBigPlanet PS Vita -- Marvel Super Hero E...	/games/littlebigplanet-ps-vita-marvel-super-he...	PlayStation Vita	9.0	Platformer	Y
2	Great	Splice: Tree of Life	/games/splice/ipad-141070	iPad	8.5	Puzzle	N
3	Great	NHL 13	/games/nhl-13/xbox-360-128182	Xbox 360	8.5	Sports	N
4	Great	NHL 13	/games/nhl-13/ps3-128181	PlayStation 3	8.5	Sports	N

Show filtered_reviews shape and titles

In [57]:
```
filtered_reviews.shape
```
Out[57]:
```
(9800, 10)
```

In [58]:
```
filtered_reviews['title'].head(10)
```
Out[58]:
```
0                                 LittleBigPlanet PS Vita
1      LittleBigPlanet PS Vita -- Marvel Super Hero E...
2                                    Splice: Tree of Life
3                                                  NHL 13
4                                                  NHL 13
7                                             Guild Wars 2
10                              Tekken Tag Tournament 2
11                              Tekken Tag Tournament 2
13                                    Mark of the Ninja
14                                    Mark of the Ninja
Name: title, dtype: object
```

Find games released for the Xbox One platform that have a score of more than 7

In [60]:
```
xbox_one_filter = (reviews["score"] > 7) & (reviews["platform"] == "Xbox One")
filtered_reviews2 = reviews[xbox_one_filter]
filtered_reviews2.head()
```
Out[60]:

	score_phrase	title	url	platform	score	genre	editors_choice	release_
17137	Amazing	Gone Home	/games/gone-home/xbox-one-20014361	Xbox One	9.5	Simulation	Y	2
17197	Amazing	Rayman Legends	/games/rayman-legends/xbox-one-20008449	Xbox One	9.5	Platformer	Y	2
17295	Amazing	LEGO Marvel Super Heroes	/games/lego-marvel-super-heroes/xbox-one-20000826	Xbox One	9.0	Action	Y	2
17313	Great	Dead Rising 3	/games/dead-rising-3/xbox-one-124306	Xbox One	8.3	Action	N	2
17317	Great	Killer Instinct	/games/killer-instinct-2013/xbox-one-20000538	Xbox One	8.4	Fighting	N	2

In [61]:
```
filtered_reviews2.shape
```
Out[61]:
```
(140, 10)
```

How many video games are ‘Action’ genre?

In [63]:
```
action_reviews = reviews[reviews['genre'] == 'Action']
```

In [64]:
```
action_reviews.head()
```

Out[64]:

	score_phrase	title	url	platform	score	genre	editors_choice	release_year
17	Great	Avengers Initiative	/games/avengers-initiative/iphone-141579	iPhone	8.0	Action	N	2012
34	Good	War of the Roses	/games/war-of-the-roses-140577/pc-115849	PC	7.3	Action	N	2012
45	Amazing	Bad Piggies	/games/bad-piggies/iphone-141455	iPhone	9.2	Action	Y	2012
49	Okay	Demon's Score	/games/demons-score/iphone-118050	iPhone	6.9	Action	N	2012
69	Great	Hotline Miami	/games/hotline-miami/pc-139657	PC	8.8	Action	Y	2012

In [65]:
```
action_reviews.shape
```

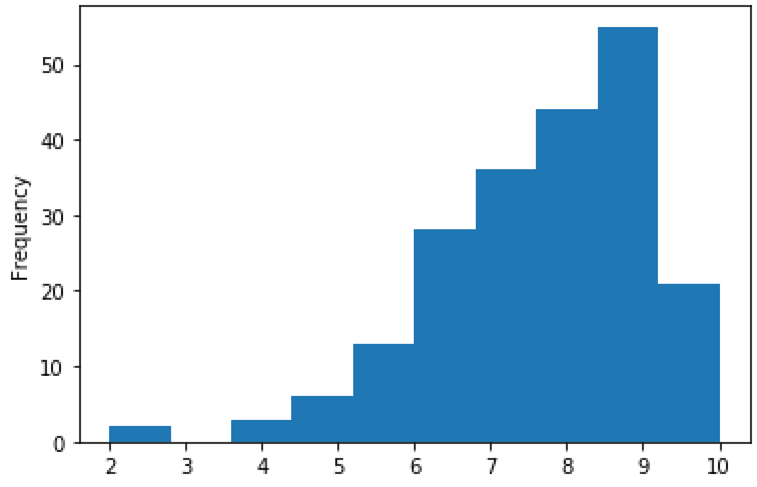
Out[65]:
```
(3797, 10)
```

0.3.7 Plot Review Ratings of two Play Stations and Compare Which one has more ratings?

Now that we know how to filter, we can create plots to observe the review distribution for the Xbox One vs the review distribution for the PlayStation 4. This will help us figure out which console has better games. We can do this via a histogram, which will plot the frequencies for different score ranges.

In [66]:
```
import matplotlib.pyplot as plt
%matplotlib inline
reviews[reviews["platform"] == "Xbox One"]["score"].plot(kind="hist")
```

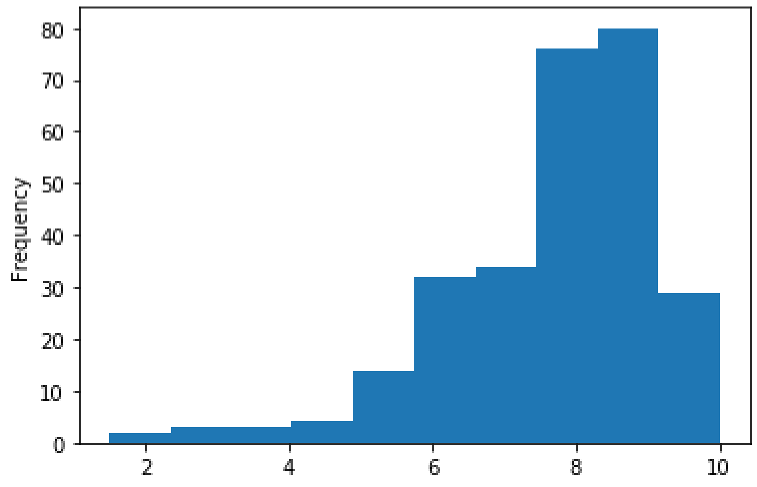
Out[66]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1f390418320>
```



Plot for Play Station4

In [68]:
```
reviews[reviews["platform"] == "PlayStation 4"]["score"].plot(kind="hist")
```

Out[68]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1f3924f3630>
```



In [ ]:
```
```