```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
```

Part 1

i)

```
# I have used sample cashflows. But you have to construct the entire cashflows for sleepin
df = pd.read_excel('cashflows_sample.xlsx', header = 0)
df = df.iloc[range(1,len(df))]  # Remove the first NaN value
r = 7.55/100
```

ii)

```
df["discount_factor"] = 1/(1+r)**df.index
df
```

	Year	Period	CF per $100	discount_factor
1	1994	1	7.55	0.929800
2	1995	2	7.55	0.864528
3	1996	3	7.55	0.803838
4	1997	4	7.55	0.747409
5	1998	5	7.55	0.694941
...	...	...	...	...
96	2089	96	7.55	0.000923
97	2090	97	7.55	0.000859
98	2091	98	7.55	0.000798
99	2092	99	7.55	0.000742
100	2093	100	107.55	0.000690

100 rows × 4 columns

```
bondvalue = np.dot(df.iloc[:,2], df.iloc[:,3])  # Calculate the bond value using the disco
print(f'The value of the bond per 100$ face value is {bondvalue:.2f}')
```

```
The value of the bond per 100$ face value is 100.00
```

iii)

```
r2 = 8.55/100
df["discount_factor2"] = 1/(1+r2)**df.index     # Fill in the updated discount factor
df
```

	Year	Period	CF per $100	discount_factor	discount_factor2
1	1994	1	7.55	0.929800	0.921234
2	1995	2	7.55	0.864528	0.848673
3	1996	3	7.55	0.803838	0.781827
4	1997	4	7.55	0.747409	0.720246
5	1998	5	7.55	0.694941	0.663515
...	...	...	...	...	...
96	2089	96	7.55	0.000923	0.000380
97	2090	97	7.55	0.000859	0.000350
98	2091	98	7.55	0.000798	0.000322
99	2092	99	7.55	0.000742	0.000297
100	2093	100	107.55	0.000690	0.000274

100 rows × 5 columns

```
bondvalue2 = np.dot(df.iloc[:,2], df.iloc[:,4])  # Fill in the bond value
print(f'The value of the bond wtth 8.55% is {bondvalue2:.2f}')
```

```
The value of the bond wtth 8.55% is 88.31
```

iv)

```
r3 = 6.55/100
df["discount_factor3"] = 1/(1+r3)**df.index     # Fill in the updated discount factor
df
```

	Year	Period	CF per $100	discount_factor	discount_factor2	discount_factor3
1	1994	1	7.55	0.929800	0.921234	0.938527
2	1995	2	7.55	0.864528	0.848673	0.880832
3	1996	3	7.55	0.803838	0.781827	0.826684
4	1997	4	7.55	0.747409	0.720246	0.775865
5	1998	5	7.55	0.694941	0.663515	0.728170
...	...	...	...	...	...	...
96	2089	96	7.55	0.000923	0.000380	0.002264

```
bondvalue3 = np.dot(df.iloc[:,2], df.iloc[:,5])   # Fill in the bond value
print(f'The value of the bond wtth 6.55% is {bondvalue3:.2f}')
```

```
The value of the bond wtth 6.55% is 115.24
```

v) Disney bond

```
sleepCF = df.iloc[:, [2]]
nap_CF = df.iloc[:10, [2]]  # Cash flows of Napping beauties assuming only 5 years. You ha
nap_CF.iloc[9] = nap_CF.iloc[9] + 100
print(sleepCF)
print(nap_CF)
```

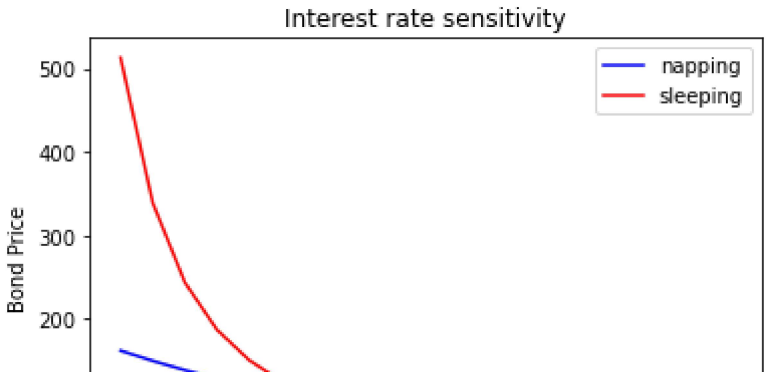
```
     CF per $100
1           7.55
2           7.55
3           7.55
4           7.55
5           7.55
..           ...
96          7.55
97          7.55
98          7.55
99          7.55
100       107.55

[100 rows x 1 columns]
    CF per $100
1          7.55
2          7.55
3          7.55
4          7.55
5          7.55
6          7.55
7          7.55
8          7.55
9          7.55
10       107.55
```

vi)

```
# Define bond price function
def bondprice(CF, r):
    CF['discount_factor'] = 1/(1+r/100)**(CF.index.values)
    CF['PV'] = CF.iloc[:,0]*CF.iloc[:,1]     # Complete this function
    price = CF['PV'].sum()
    return price


# Calculate the bond prices
sleepprice = np.zeros(20)
napprice = np.zeros(20)
for i in range(1, 21):
    sleepprice[i-1] = bondprice(sleepCF,i) # Complete the argument
    napprice[i-1] = bondprice(nap_CF,i)
```

```
/var/folders/dc/n6cypfsj2ml0wt4c_p9wzzgh0000gn/T/ipykernel_48159/2875711872.py:3: Set
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  CF['discount_factor'] = 1/(1+r/100)**(CF.index.values)
/var/folders/dc/n6cypfsj2ml0wt4c_p9wzzgh0000gn/T/ipykernel_48159/2875711872.py:4: Set
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  CF['PV'] = CF.iloc[:,0]*CF.iloc[:,1]     # Complete this function
/var/folders/dc/n6cypfsj2ml0wt4c_p9wzzgh0000gn/T/ipykernel_48159/2875711872.py:3: Set
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  CF['discount_factor'] = 1/(1+r/100)**(CF.index.values)
/var/folders/dc/n6cypfsj2ml0wt4c_p9wzzgh0000gn/T/ipykernel_48159/2875711872.py:4: Set
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  CF['PV'] = CF.iloc[:,0]*CF.iloc[:,1]     # Complete this function
```

```
plt.title("Interest rate sensitivity")
plt.xlabel("Interest Rate")
plt.ylabel("Bond Price")
plt.plot(range(1,21),napprice, 'b', label = "napping")
plt.plot(range(1,21),sleepprice, 'r', label = "sleeping")
plt.legend(loc="upper right")
plt.show()
```



sleepCF

	CF per $100	discount_factor	PV
1	7.55	8.333333e-01	6.291667e+00
2	7.55	6.944444e-01	5.243056e+00
3	7.55	5.787037e-01	4.369213e+00
4	7.55	4.822531e-01	3.641011e+00
5	7.55	4.018776e-01	3.034176e+00
...	...	...	...
96	7.55	2.503804e-08	1.890372e-07
97	7.55	2.086504e-08	1.575310e-07
98	7.55	1.738753e-08	1.312758e-07
99	7.55	1.448961e-08	1.093965e-07
100	107.55	1.207467e-08	1.298631e-06

100 rows × 3 columns

nap_CF

```
     CF per $100  discount_factor         PV
```

vii)

The prices of these two bonds are the same when the interest rate is 7.55%.

viii)

Napping Beauty is worth more when the interest rate is above 7.55% because the PV of the largest cash flow(at time 10) is large.

ix)

Sleeping Beauty is worth more when the interest rate is below 7.55% because this bond has 100 cash flows and the PV of the last CF is larger if the interest rate is smaller.

x)

```
def duration(CF, r):
    CF['discount_factor'] = 1/((1+r/100)**(CF.index.values))
    CF['PV'] = CF["CF per $100"]*CF['discount_factor']
    CF['weights'] = CF['PV']/(CF['PV'].sum())
    CF['time'] = CF.index.values
    duration = np.dot(CF['weights'],CF['time'])                    # Complete the durati
    return duration


macaulay_duartion_sleep = duration(sleepCF,7.55)           # Complete the arguments
macaulay_duartion_nap = duration(nap_CF,7.55)
MD_sleep = macaulay_duartion_sleep/(1+0.0755)                # Compute modified duration u
MD_nap = macaulay_duartion_nap/(1+0.0755)
delta_price_sleep = - MD_sleep*0.01
delta_price_nap = - MD_nap*0.01
print(f'The Macaulay Duration of Sleeping Beauties is {macaulay_duartion_sleep: .2f}')
print(f'The Macaulay Duration of Napping Beauties is {macaulay_duartion_nap: .2f}')
print(f'The price change of Sleeping Beauties is {delta_price_sleep: .2f}')
print(f'The price change of Napping Beauties is {delta_price_nap: .2f}')
```

```
The Macaulay Duration of Sleeping Beauties is  14.24
The Macaulay Duration of Napping Beauties is  7.37
The price change of Sleeping Beauties is -0.13
The price change of Napping Beauties is -0.07
/var/folders/dc/n6cypfsj2ml0wt4c_p9wzzgh0000gn/T/ipykernel_48159/737374224.py:2: Set
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  CF['discount_factor'] = 1/((1+r/100)**(CF.index.values))
/var/folders/dc/n6cypfsj2ml0wt4c_p9wzzgh0000gn/T/ipykernel_48159/737374224.py:3: Set
A value is trying to be set on a copy of a slice from a DataFrame.
```

```
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  CF['PV'] = CF["CF per $100"]*CF['discount_factor']
/var/folders/dc/n6cypfsj2ml0wt4c_p9wzzgh0000gn/T/ipykernel_48159/737374224.py:4: Sett
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  CF['weights'] = CF['PV']/(CF['PV'].sum())
/var/folders/dc/n6cypfsj2ml0wt4c_p9wzzgh0000gn/T/ipykernel_48159/737374224.py:5: Sett
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
  CF['time'] = CF.index.values
```

The sensitivity of Sleeping Beauty is larger than that of Napping Beauty.

xi)

Firstly, the Disney bonds have longer maturity than 30-year U.S Treasury bonds so they are more risky and there is a higher yield. Secondly, the bonds issued by Disney have more default risk than bonds issued by government so investors require a higher return.

xii a

Since the company can recall the bonds when interest rates decline, the bondholder will lose out on the stream of high interest payments relative to the market and will have to reinvest his money at a reduced rate, which reduces the attractiveness of the bonds on the date of issuance for the bondholder.

xii b

Considering that the Disney Company can benefit from interest rate decreases and refinance the bonds at a cheaper interest rate by calling back the bonds, management would add a call option in the bond's conditions. Even in a low interest environment, the Company will pay bondholders a higher coupon rate if the option is not present.

xii c

Due to the non-callability of the $150 million in 100 year bonds issued by Coca-Cola, their yield was lower than Disney's. Due to the bonds' non-callability, Coca-Cola is unable to take advantage of any potential future falls in interest rates. Because the rates are set for the next 100 years, the return offered is lower to offset the losses from declining interest rates.

xiii

Disney management decided to issue unusually long bonds because they anticipate interest rates will be higher than those currently in effect for a very long time. The decision was made to finance their debt by locking in today's reduced interest rates.

In my capacity as a top Disney executive, I would refrain from issuing 100-year bonds in light of the current financial situation and the sky-high interest rates. Till inflation is under control, the Fed intends to increase rates even further. As a result, Disney would incur a significant annual interest payment if it issued bonds right now. Before releasing 100-year bonds, I would hold off until the Fed starts lowering rates and they are almost at their lowest level.

xiv

Recent issuers of Centenary Bonds include: 1)Israel: Issued its first 100 year bond in 2020 to fund stimulus to limit the damage caused by the coronavirus pandemic. 2)Peru: Issued its first 100 year bond in 2020 (Total Offering: $1 billion) to deal with economic contraction and political crisis. These nations have issued bonds with a 100-year maturity in order to lock in lower interest rates for their planned spending needs over the coming century. Even if interest rates increase in the future, these nations will continue to pay the low rate that was specified when the bonds were issued. These nations think that interest rates have reached their lowest point, or are very close to it, and will eventually rise.

xv a

Maturity Date: 07/15/2093 Offering Date: 07/15/1993

xv b

Yes the bond is callable. Call date 07/15/2023

xv c

Coupon Rate: 7.550 % Yield: 6.710 %

xv d

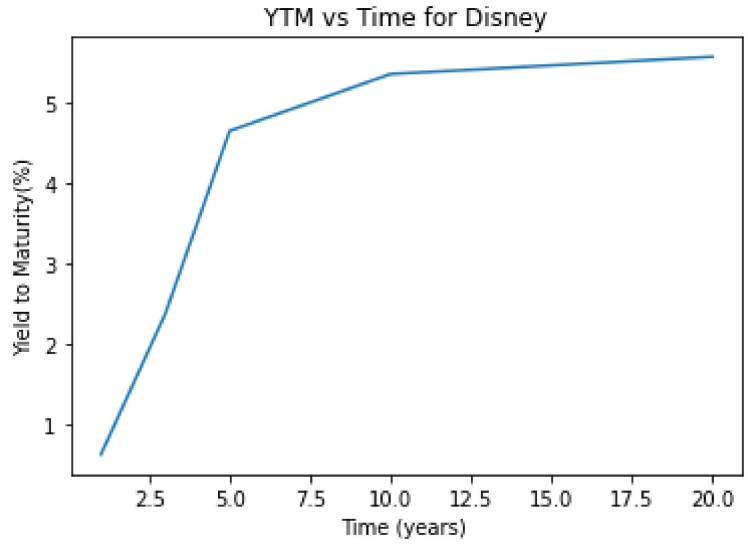
Due to the difference in interest rates between when the bond was issued and now, the bond's yield, which is 6.710%, is lower than the coupon rate.

The bond's yield decreased from the yield at the time of issuance to nearly equal the current market rate as the value of the longest benchmark interest rate increased. However, it is still

higher than 3.81% because it carries some credit risk in comparison to the risk-free US Treasury Bonds.

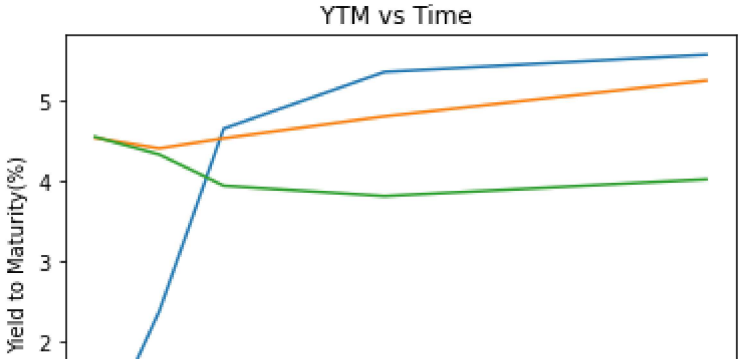
xvi

```
period= [1,3,5,10,20]
bond_yields_disney= [0.622,2.373,4.653,5.361,5.576]
plt.title("YTM vs Time for Disney")
plt.xlabel("Time (years) ")
plt.ylabel("Yield to Maturity(%)")
plt.plot(period,bond_yields_disney)
plt.show()
```



xvii

```
bond_yields_berkshire= [4.538,4.406,4.532,4.807,5.253]
bond_yields_treasury= [4.55,4.33,3.94,3.81,4.02]
plt.title("YTM vs Time ")
plt.xlabel("Time (years) ")
plt.ylabel("Yield to Maturity(%)")
plt.plot(period,bond_yields_disney,label="disney")
plt.plot(period,bond_yields_berkshire,label="berkshire")
plt.plot(period,bond_yields_treasury,label="treasury")
plt.legend()
plt.show()
```



The slope of Disney's yields is the highest, and they vary with time. For time periods longer than five years, the yields are noticeably greater than Berkshire Hathaways', demonstrating that Disney is a riskier business than Berkshire and must thus offer a larger yield. Compared to Disney, Berkshire is a fairly safe investment. Both yield curves are greater than the Treasury Yield Curve, demonstrating that Treasury's are the safest investment and serving as the benchmark YTM.ss