FM 9528 Banking Analytics

Coursework 1

Student ID: 251121253

Word Count: 1648

Question 1

Part a)

- ✦ **Total Bond Price: $ 52,605,441,714**

- ✦ Additional Information:

1. All Bond price in the datasets are in cents
2. Discounts Rates are Average Yield of Government of Canada Marketable Bonds as of Sep 26th, 2019 (Bank of Canada, 2019)

1-3 Years	3-5 years	5-10 years	More than 10 years
1.58%	1.43%	1.37%	1.52%

- ✦ Methodology to Calculate Bond Prices (Pseudocode) :

Load Bond Dataset;
For each Bond in Dataset:

$$NumberofPayment = Time2Maturity$$

$$Coupon = YearlyRate \cdot FaceValue$$

$$DiscountRate = AnnualAverageYield$$

$$FaceValue = Principle$$

Using Numpy's function "pv", we have:

NormalBondPrice = -numpy.pv(DiscountRate, NumberofPayment, Coupon, FaceValue)

$$Zero\ Coupon\ Bond = \frac{FaceV}{(1+r)^n}$$

$$BondPrice = BondsHeld \cdot SingleBondPrice$$

$$TotalBondPrice = \sum BondPrice$$

Part b)

✦ The Provision and Capital Requirements Results:

	Provision ($)	F-IRB Capital Requirement($)
Bonds	9,858,989,107	20,092,672,282
Mortgages	512,264,202	755,794,845
Other Retail Loan	25,753,739	18,518,438

✦ Assumptions & Additional Information:

3. All bonds are considered as subordinate cooperate bonds: LGDs were considered as 75% in the Foundational Approach.
4. The LGD of Retail Loans are considered 100
5. According to current Basel III Accord, PD floor (0.03%) and LGD floors (10%) applies through all the calculations
6. All Mortgages are considered as residential mortgages
7. Mortgages' EAD are considered as 0 when (Unpaid Principle - Collateral) < 0

✦ Methodology of Calculation:

Load DataSet:
For each Bond, Mortgage, Loan in Data Set:

EAD of Bond = Current Bond Price
EAD of Mortgages = Unpaid Principle - Collateral
EAD of Loans = Unpaid Principle
where

$$UnpaidPrinciple = OriginalBalance \cdot (1 + PeriodRate)^{npaid} - Pmt \cdot [\frac{(1 + PeriodRate)^{npaid} - 1}{PeriodRate}]$$

PD = Max(Given PD, PD floor)
LGD = Max(Given LGD, LGD floor)

Calculation of Provision:

$$Provision = PD \cdot LGD \cdot EAD$$

Calculation of Capital Requirement (K)

$$K = LGD \cdot \left\{ N\left(\sqrt{\frac{1}{1-R}} \cdot N^{-1}(PD) + \sqrt{\frac{R}{1-R}} \cdot N^{-1}(0.999) \right) - PD \right\} \left(\frac{1+(M-2.5)b}{1-1.5b} \right)$$

where

$$R \text{ of Bond } = 0.12\left(\frac{1-e^{-50PD}}{1-e^{-50}}\right) + 0.24\left(1-\frac{1-e^{-50PD}}{1-e^{-50}}\right)$$

M of Bond = Time2Maturity

b of Bond = $(0.11852 - 0.05478 ln(PD))^2$

R of Mortgage = 0.15

$$R \text{ of Other Retail } = 0.03\left(\frac{1-e^{-35PD}}{1-e^{-35}}\right) + 0.16\left(1-\frac{1-e^{-35PD}}{1-e^{-35}}\right)$$

M of Mortgages & Other Retail Loan = 1

Total Provision for each products = $\sum Provision$ *in each business line*

Total Capital Requirement for each product = $\sum K \cdot EAD \cdot 12.5 \cdot 11.5\,\%$ *in each business line*

Part c)

✦ The Results

	Standardized Approach	IRB Approach
Total RWA ($)	44,443,757,320	181,452,048,401
Total Regulatory Capital ($)	5,111,032,091	20,866,985,566

✦ Methodology of Calculation:

Load Bond, Mortgage, Loan from Dataset;
For each Bond, Mortgage and Loan in Dataset:

$$IRB\_RWA = 12.5 \cdot k \cdot EAD$$

where k & EAD are parameters calculated from part b)

$$SA\_RWA = riskweight \cdot EAD$$

where standard risk weight is assigned according to chapter 3 of OSFI (Office of the Superintendent of Financial Institutions, 2018)
(See Appendix for specific risk weights)

Under Each Approach:

$$Total\ RWA = \sum BondRWA + \sum MortgageRWA + \sum LoanRWA$$

$$Total\ Regulatory\ Capital = Total\ RWA \cdot 11.5\%$$

where 11.5% is the capital adequacy requirement from OSFI

✦ Comparison with RBC's Q3 Pillar 3 Report (Royal Bank of Canada, 2019):

According to RBC's Q3 Pillar 3 Report, the reported capital requirement for credit risk exposure is $27,416 million, which is approximately 1.31 times of our higher capital requirement $20,867 calculated under F-IRB Approach. In addition to that, the entire RBC has a regulatory capital of $ 40,854 million; therefore the size of our bank should be a medium-sized bank considering RBC ranks No.1 in term of assets value in Canada.

Question 2

Part a)

The following is a brief summary of the Basel IV (Basel Committee on Banking Supervision, 2017)

✦ Changes in Treatment of the Credit Risk

- Standardized Approach (SA)

  Compare to Basel III, the risk weights assigned to different classes of asset are recalibrated with more risk sensitivity and robustness in Basel IV. For example: the updated SA assigns a risk weight of 75% or 35% to residential mortgages depends on whether the mortgage' Loan to Value Ratio exceeds 80% or not. Compare to the flat rate of 35% in Basel III, the new weight measures risk more accurately.

- The Internal Rating-Based Approach (IRB)

  Basel IV removes the option of using the A-IRB Approach to estimate the parameters of exposures on large-to-mid sized corporates and financial institutions from Basel III. It also updates floors for the inputs in IRB Approach with specifications on the characteristics of assets. The PD floor was lifted from 3bp to 5 bp (10 bp for QRRE revolvers), and the LGD now has various floors depends on the types of assets.

✦ The Treatment of the Operational Risk

Basel IV unifies the original four alternative approaches of measuring operational risk into one Standardized Measurement Approach (SMA). It uses a much more straightforward formula to calculate the operational risks :"Operational risk capital = Business Indicator Component x Internal Loss Multiplier."

✦ The Leverage Frame Work

Basel IV introduces a leverage ratio buffer for Global Systemically Important Banks (G-SIBS) as an add-on buffer. It also refines the leverage ratio exposure measure to ensure that off-balance exposures are measured consistently with the new SA in credit risks.

✦ The New Capital Output Floor

Basel IV replaces the old output floor, which requires banks' capital output to be at least 80% of relevant Basel I capital requirement with a new standard. The new floor sets as the maximum of the appropriate RWA calculated under regulation permission and 72.5% RWA calculated using the new Standardized Approach.

Part b) Potential Impact of Changes of Parameters in Basel IV

✦ Changes in Treatment of Credit Risk

The recalibrated risk weights in SA will help banks to measure their credit risk in a more risk sensitive manner. At the same time, banks who took advantage of A-IRB to lower their credit risk need to add more Common Equity Tier1 (CET1) due to the elimination of the approach for measuring exposures of large-to-medium sized corporates and financial institutions. With these two changes, it may encourage more banks to adopt the Standardized Approach to measure credit risks. In addition to that, Since PD and LGD are the two major parameters for calculating capital requirements under IRB, the updates in floors of PD and LGD will lead variations on banks' capital requirements depends on the structures of their credit products.

✦Changes in Operational Risk

Banks will lose flexibility in measuring their operational risk due to the reform of the measurement approaches. However, since SMA (the new method) is the only option left to them, it will increase the comparability of measuring operational risk among banks.

✦Changes in Leverage Frame Work

The newly introduced leverage ratio buffer would generally impose another layer of CET1 on top of the current CET1 level for G-SIBS.

✦New Capital Output Floor

The new output floor (minimum of 72.5% RWA calculated in SA) will force banks who took advantage of using IRB Approaches to add more CET1 to meet the regulation requirements. According to a report from KPMG (2018), it estimates that almost 75% of the banks in Europe would have decreases in their CET1 ratios after implementing the output floor. Facing this situation, banks will probably decrease their credit risk exposure to reduce the additional regulatory capital it has to hold.

✦The Most Affected Canadian Banks and Financial Institutions

In general, the transition to Basel IV will have a limited impact on Canadian banks and financial institutions because OSFI has set rules to keep them stay ahead in the course of banking regulations. By the end of 2018, all the Canadian financial institutions that use the A-IRB Approach had already implemented an output floor, which is 75% of RWA calculated under SA of Basel II (ScotiaBank, 2019). Although SA parameters may shift up under Basel IV, with the current output floor as a preparation, the new output floor will not cause a drastic decrease in the CET1 ratio for Canadian banks. If we have to pick one institution that will be affected the most, Bank of Nova Scotia (Scotiabank) may be the one. While Scotiabank has the third largest RWA in Canada, it mainly uses the A-IRB Approach to measure its credit risk exposure (ScotiaBank, 2019). As Basel IV updates the output floor, the constraints on A-IRB, and the leverage ratio buffer, Scotiabank's risk measurement could face a relatively large change compare to other institutions who mainly use F-IRB or SA to measure credit risks (i.e. RBC and BMO).

Therefore, considering the size of Scotiabank, it may need to add more CET1 or reduce its risk exposure to meet the new standards in Basel IV.

Part c) Evaluate the Proposal of BCBS

The major criticism of Basel IV is that it will force banks to raise more regulatory capital based on the current level which is already high, and it will create incentives for banks to reduce their exposure to certain borrowers or financial products (Wass.S, 2017). This could make it difficult for small-to-medium entities to access financial products and services and causing a slowdown of economic growth. On the other hand, Basel IV strengthens the global banking system by introducing more risk-sensitive parameters, unifying measurement approaches, and reducing modelling risk created from the A-IRB Approach. It is an upgrade patch of Basel III to prevent the “domino effect” that happened in the most recent financial crisis.

In my opinion, the benefit of adopting Basel IV overweighs the negative impact it brings. Although Basel IV will further lift banks’ regulatory capital, the size of impact varies by bank types, structure, and jurisdiction regions. Canadian banks, for example, could be less affected comparing to European banks due to the strict regulations imposed by OSFI. Therefore, Basel IV is not a nightmare for every bank or economics in the world. Given that there are 8 more years to fully implement Basel IV, the negative impact could be mitigated or reduced by strategic adjustments such as moving assets off balance sheet or changing the composition of business through the timeline (Mckinsey & Company, 2017). BCBS updated the parameters and methodologies in Basel IV to put banks in a more conservative position for preparing and defending the next financial crisis. Especially when recent global economic growth has slowed down, but trade wars keep going on, banks have to watch out for extremely adverse situations. Because of this, it is reasonable for Basel IV forces banks to operate conservatively. In conclusion, I believe the stable global banking system created by Basel IV overweighs the partial negative impact it brings.

Word Count: 1648

References

1. Bank of Canada. (2019, September 26). Selected Bond Yields. Retrieved September 26, 2019, from https://www.bankofcanada.ca/rates/interest-rates/canadian-bonds/.

2. Basel Committee on Banking Supervision. (2017). High-level summary of Basel III reforms. Retrieved from https://www.bis.org/bcbs/publ/d424_hlsummary.pdf

3. KPMG. (2018, April). Basel 4: The Way Ahead. Retrieved from https://assets.kpmg/content/dam/kpmg/xx/pdf/2018/05/basel-4-credit-risk-irb-approach.pdf

4. Mckinsey & Company . (2017, November). Bring Basel IV to Focus . Retrieved October 10, 2019, from https://www.mckinsey.com/business-functions/risk/our-insights/bringing-basel-iv-into-focus.

5. Office of the Superintendent of Financial Institutions. (2018, November). Capital Adequacy Requirements (CAR) Chapter 3 – Credit Risk – Standardized Approach. Retrieved October 10, 2019, from http://www.osfi-bsif.gc.ca/eng/fi-if/rg-ro/gdn-ort/gl-ld/pages/car18_chpt6.aspx.

6. Royal Bank of Canada . (2019). Royal Bank of Canada Pillar 3 Report . *Royal Bank of Canada Pillar 3 Report* . Retrieved from http://www.rbc.com/investorrelations/pdf/19Q3pillar3.pdf

7. Scotiabank. (2019). SUPPLEMENTARY REGULATORY CAPITAL DISCLOSURES. Retrieved from https://www.scotiabank.com/content/dam/scotiabank/corporate/quarterly-reports/2019/q3/2019Q3-Supplementary-Regulatory-Capital-Disclosures.pdf

8. Wass, S. (2017, September 5). Basel IV: Good or bad news for trade finance. Retrieved October 10, 2019, from https://www.gtreview.com/magazine/volume-15issue-5/basel-iv-good-bad-news-trade-finance/.

Appendix

Colab Link

https://colab.research.google.com/drive/1kJVibArE7uHwQLN2ykwNEXF7Pw9MQgjN

1.0 Initiate the Data File

```
import numpy as np
!gdown https://drive.google.com/uc?id=1dkEHvr6w6-4o4aTebCLwT6k0kI59tY0l
import pandas as pd
Bankdata = pd.ExcelFile('/content/Dataset.xlsx')
Bond = pd.read_excel(Bankdata, 'Bond')
Mortgage = pd.read_excel(Bankdata, 'Mortgage')
Loan = pd.read_excel(Bankdata, 'RetailLoan')
```

```
Downloading...
From: https://drive.google.com/uc?id=1dkEHvr6w6-4o4aTebCLwT6k0kI59tY0l
To: /content/Dataset.xlsx
17.0MB [00:00, 46.6MB/s]
```

1.1 Calculate the Current Bond Price

1.1.1 Functions to Calculate the Bond Price

$$BondPrice = Coupon \cdot \frac{1-(1+r)^{-n}}{r} + \frac{FaceV}{(1+r)^n}$$

where

$$Coupon = YearlyRate \cdot FaceValue$$

$$r = DiscountRate$$

$$n = Time2Maturity$$

$$FaceV = Principle$$

**Notice that for zero coupon bonds, "Remainning Coupon" = 0 **

$$ZeroCouponBond = \frac{FaceV}{(1+r)^n}$$

```
# Function to determine discout rate

def discount_deter (Maturity):
  #As of September 26th, 2019
  short_term_rate = 1.58/100  #1 to 3 years
  medium_term_rate = 1.43/100 #3 to 5 years
  relative_long_term_rate = 1.37/100 #5 to 10 years
  long_term_rate = 1.52/100 #over 10 years
  if Maturity <= 3:
    disr =  short_term_rate
  elif (Maturity > 3) and (Maturity <= 5):
    disr =  medium_term_rate
  elif (Maturity >5) and (Maturity <= 10):
    disr =   relative_long_term_rate
  else:
```

```
    disr =  long_term_rate

  return disr

# Function to calculate the bond prices

def Price_cal (npay,yrate,Time,FaceV):

  # Handle 0 coupon bond
  if npay == 0:
    disr = discount_deter(Time)
    output = FaceV/((1+disr) ** Time)

  # Handle normal coupon bond
  elif npay > 0:
    disr = discount_deter(Time)
    coupon = yrate/100* FaceV
    output = -np.pv(disr,Time,coupon,FaceV)
  else :
    print ('error')

  return output
```

1.1.2 Loop though the Data Set to Calculate the Bond Price

```
Stop = max(Bond.id)
CurrPrice = []
TotalPrice = []
for index, row in Bond.iterrows():
  if row['id'] <= Stop:
    price = Price_cal(row['CouponsRemaining'], row['YearlyRate'],
                      row['Time2Maturity'], row['Principal'])
    CurrPrice.append(price)
    # Since Professor announced that those prices are in cents
    TotalPrice.append(price*row['BondsHeld']/100)
  else:
    break


## Attach Saved result to data frame
Bond['CurrentPrice'] = CurrPrice
Bond['TotalPrice'] = TotalPrice
```

1.1.3 Extract the Calculation Result

```
print(sum(Bond['TotalPrice']))
```

```
52605441714.425255
```

1.2 Calculate the Provison & Capital Requirements in each product

In this section, we will loop through the dataset of Bond, Mortgages, and Other Retail Loans. during the iterations, we will calculate the Provision, RWA under Standarized Appraoch, RWA under IRB Apporach, Capital Requirement under Standarized Appraoch, Capital Requirement under IRB Apporach for each business line. Some of the calculation results will be used in Section 1.3

1.2.1 Initiate Constant Parameters and Functions that Required in the Calculation Process

Formulas below are the function we need to use to calculate the results

$$Provision = PD \cdot LGD \cdot EAD$$

- EAD of Bond = Current Bond Price
- EAD of Mortgages = Unpaid Principle - Collectral
- EAD of Loans = Unpaid Principle

Where

$$UnpaidPrinciple = OriginalBalance \cdot (1 + PeriodRate)^{npaid} - Pmt \cdot [\frac{(1 + PeriodRate)^{npaid} - 1}{PeriodRate}]$$

- PD = Max(Given PD, PD floor)
- LGD = Max(Given LGD, LGD floor)

Formulas for Calculating RWA and Capital Requirements:

$$K = LGD \cdot \left\{ N \left(\sqrt{\frac{1}{1-R}} \cdot N^{-1}(PD) + \sqrt{\frac{R}{1-R}} \cdot N^{-1}(0.999) \right) - PD \right\} \left(\frac{1 + (M - 2.5)b}{1 - 1.5b} \right)$$

Parameters for Mortgages

- Mortgages: $R = 0.15$

Parameters for Other Retail Loans

- Other retail: $R = 0.03 \left(\frac{1-e^{-35PD}}{1-e^{-35}} \right) + 0.16 \left(1 - \frac{1-e^{-35PD}}{1-e^{-35}} \right)$

Parameters for Bonds

- Bond exposures $R = 0.12 \left(\frac{1-e^{-50PD}}{1-e^{-50}} \right) + 0.24 \left(1 - \frac{1-e^{-50PD}}{1-e^{-50}} \right)$
- $b = (0.11852 - 0.05478 ln(PD))^2$

$$RWA\ of\ IRB = 12.5 \cdot K \cdot EAD$$

$$RWA\ of\ IRB = RiskWeight \cdot EAD$$

Risk Weight is assigned according to Chapter 3 OFSI

$$Capital\ Requirement\ = RWA * CapitalRequirementPercentage$$

For our bank

$$CapitalRequirementPercentage = 11.5\%$$

```
## According to the Foundational Approach,
## LGD for Bonds is 75% if we consider them all as subordinated bonds
## So we assign the LGD for coperate bonds in the loop as well
Bond['LGD'] = 0.75

## For Retail Loans, the LGD are considered as 100%
```

```
Loan['LGD'] = 1


## Assign PD and LGD Floor to each business line
## For Bonds and Retail Loans, since their default LGD is 75% abd 100%
## respectively; therefore, LGD floor only applies to Mortgages
## We will apply the LGD and PD floors in the process of calculation instead of
## alterning the data
PDfloor = 0.03/100
LGDfloor = 0.10



# Capital Requirement Percentage
cap_req_percent = 11.5 /100


#Formulas for calculate correaltion in different class of assets
#Retail Exposures:
import numpy as np
def other_Retail_R (PD):
  output = 0.03 * ( (1 - np.exp(-35 * PD)) / (1 - np.exp(-35)) )
  output += 0.16 * (1 - ( (1 - np.exp(-35 * PD)) / (1 - np.exp(-35)) ) )
  return output


#Corporate Exposures

def Coper_b (PD):
  output = (0.11852 - 0.05478 * np.log(PD))**2
  return output

def Coper_R (PD):
  output = 0.12 * ( (1 - np.exp(-50 * PD)) / (1 - np.exp(-50)) )
  output += 0.24 * (1 - ( (1 - np.exp(-50 * PD)) / (1 - np.exp(-50)) ) )
  return output

#Function to calculate capital requirements for Bonds:

def CapReq_Bond(LGD, PD, M):
  from scipy.stats import norm
  b = Coper_b (PD)
  R = Coper_R (PD)
  K = norm.cdf(np.sqrt( (1 - R) ** (-1) ) * norm.ppf(PD) +
        np.sqrt( R / (1 - R) ) * norm.ppf(0.999) ) - PD
  K *= LGD * ((1 + (M-2.5)*b)/(1-1.5*b))
  return(K)

#Function to calculate capital requirements for Mortgages:

def CapReq_Morg(LGD, PD):
  from scipy.stats import norm
  R = 0.15
  K = norm.cdf(np.sqrt( (1 - R) ** (-1) ) * norm.ppf(PD) +
               np.sqrt( R / (1 - R) ) * norm.ppf(0.999) ) - PD
  K *= LGD
  return (K)

#Function to calculate capital requirements for Other Retail Loan:

def CapReq_other_Retail(LGD, PD):
  from scipy.stats import norm
  R = other_Retail_R (PD)
  K = norm.cdf(np.sqrt( (1 - R) ** (-1) ) * norm.ppf(PD) +
               np.sqrt( R / (1 - R) ) * norm.ppf(0.999) ) - PD
  K *= LGD
  return (K)




########################################################
########################################################
##
```

```
##
##
##    Standarized Risk Weight Constant
##
##
#######################################################
#######################################################


#Function to Assign weight to according to bond's rating
# Weight For AAA to AA- : 20%
# Weight For A+ to A- : 50%
# Weight For BBB+ to BB- : 100%
# Weight For below B - : 150%
def Weight_assign(PD):
  if PD  <= 0.01:
    weight = 20/100
  elif (PD > 0.01) and (PD <= 0.12):
    weight = 50/100
  elif (PD > 0.12) and (PD <= 0.52):
    weight = 100/100
  else:
    weight = 150/100

  return weight


# Function to assign mortgages weights
# (assume all mortgages are residential mortgages)
# If LTV ratio < 80% : risk weight is 35%
# If LTV ratio > 80%: risk weight is 75%

def Weight_assign_Mortgage (loan, collateral):
  if loan/collateral < 0.80:
    weight = 0.35
  else:
    weight = 0.75

  return weight


# Weight for Other Retail Loans are a constant
# defined as below 75%
Weight_Retail_Loan = 0.75
```

1.2.2 Calculate Provison and Capital Requirements in the Bond Business Line

```
Stop1 = max(Bond.id)
#Initiate Empty List to record values
Bond_Provison = []
Bond_IRB_RWA = []
Bond_IRB_Cap = []
Bond_weight = []
Bond_SA_RWA = []
Bond_SA_Cap = []

for index, row in Bond.iterrows():
  if row['id'] <= Stop1:
    # LGD floor and PD floor applies here
    adjusted_PD = max(row['PD'],PDfloor)
    adjusted_LGD = max(LGDfloor,row['LGD'])
    Provison = row['TotalPrice']*adjusted_PD*adjusted_LGD
    Bond_Provison.append(Provison)

    #RWA in F-IRB
    Cap_Req = CapReq_Bond(adjusted_LGD, adjusted_PD, row['Time2Maturity'])
    Cap_Req = Cap_Req * row['TotalPrice']
    RWA_V = 12.5 * Cap_Req
    Bond_IRB_RWA.append(RWA_V)
```

```
        #Capital Requirement in F-IRB
        Final_Cap_Req = RWA_V * cap_req_percent
        Bond_IRB_Cap.append(Final_Cap_Req)

        #Assign weight
        weight_V = Weight_assign(row['PD'])
        Bond_weight.append(weight_V)

        #RWA in Standized Approach
        W_RWA_V = row['TotalPrice']  * weight_V
        Bond_SA_RWA.append(W_RWA_V)

        #Capital Requirement in SA
        Final_Cap_Req2 =  W_RWA_V * cap_req_percent
        Bond_SA_Cap.append (Final_Cap_Req2)


      else:
        break


    ## Attach Saved result to data frame
    Bond['Bond_Provison'] = Bond_Provison
    Bond['Capital_Requirement_IRB'] = Bond_IRB_Cap
    Bond['IRB_RWA'] = Bond_IRB_RWA
    Bond['weight'] = Bond_weight
    Bond['SA_RWA'] = Bond_SA_RWA
    Bond['Capital_Requirement_SA'] = Bond_SA_Cap
```

1.2.3 Calculate Provison and capital Requirements in the Mortgage Business Line

```
## Provision in the Mortgage Business Line


## Function to Calculate the Remainning Balance
def Rem_Bal (Principal, APR, Years, npaid):
    monthly_interest = APR/ 12
    payment_number = Years * 12

    #Calculate the Fixed Payment of Mortgage
    monthly_payment = Principal*(monthly_interest/(1-(1+monthly_interest)
                                               **(- payment_number)))


    #Calculate Unpaid Priciple
    rem_bal = Principal * (1 - ((1 + monthly_interest) ** npaid - 1) /
                           ((1 + monthly_interest) ** payment_number - 1))

    return rem_bal


## Calculation of Remaining Balance for Mortgages
Stop2 = max(Mortgage.ID)
Mortgage_Provison = []
EAD = []
Remaining_Bal = []
Mortgage_IRB_RWA = []
Mortgage_IRB_Cap= []
Mortgage_weight = []
Mortgage_SA_RWA = []
Mortgage_SA_Cap = []

for index, row in Mortgage.iterrows():
  if row['ID'] <= Stop2:
    # Calculate Remaining Balance
    Remaning_Bal_V = Rem_Bal(row['OriginalAmount'], row['Rate'],row['TermYears']
                             , row ['CurrentInstallment'])
    Remaining_Bal.append(Remaning_Bal_V)
    # Calculate the EAD
    # If the collectral exceeds remaning balance, the EAD should be 0
```

```
        # instead of negative
        EAD_V =  max(0, Remaning_Bal_V - row['Collateral_valuation'])
        EAD.append(EAD_V)
        # Calculate the Provison
        # Applies PD and LGD Floor

        adjusted_PD = max(row['PD'],PDfloor)
        adjusted_LGD = max(LGDfloor,row['LGD'])


        Provison = EAD_V * adjusted_PD * adjusted_LGD
        Mortgage_Provison.append(Provison)


        # Risk Weighted Assets in IRB
        Cap_Req = CapReq_Morg(adjusted_LGD,adjusted_PD)
        Cap_Req = Cap_Req * EAD_V
        RWA_V = 12.5 * Cap_Req
        Mortgage_IRB_RWA.append(RWA_V)

        # Capital Requirement in IRB
        Final_Cap_Req = RWA_V * cap_req_percent
        Mortgage_IRB_Cap.append(Final_Cap_Req)

        # Assign Weights to Mortgages
        Loan_V = row['OriginalAmount']
        Collateral_V = row['Collateral_valuation']
        weight_V = Weight_assign_Mortgage (Loan_V, Collateral_V)
        Mortgage_weight.append(weight_V)


        # Calculate RWA using SA
        W_RWA_V =  EAD_V * weight_V
        Mortgage_SA_RWA.append(W_RWA_V)

        # Capital Requirement in SA
        Final_Cap_Req2 = W_RWA_V * cap_req_percent
        Mortgage_SA_Cap.append(Final_Cap_Req2)


      else:
        break

    ## Attach Saved result to data frame
    Mortgage['EAD'] = EAD
    Mortgage['Unpaid_Bal'] = Remaining_Bal
    Mortgage['Provision'] = Mortgage_Provison
    Mortgage['Capital_Requirement_IRB'] = Mortgage_IRB_Cap
    Mortgage['IRB_RWA'] = Mortgage_IRB_RWA
    Mortgage['Weight'] = Mortgage_weight
    Mortgage['SA_RWA'] = Mortgage_SA_RWA
    Mortgage['Capital_Requirement_SA'] = Mortgage_SA_Cap
```

1.2.4 Calculate Provison and Capital Requirements in the Retail Loan Business Line

```
    ## Provision in the Retail Loan Line


    ## Function to Calculate the Remainning Balance for Loan
    ## We assumes the loan terms are in months
    def Rem_Bal_Loan (Principal, APR, terms, npaid):
        monthly_interest = APR/ 12
        payment_number = terms

        # Calculate fixed payment amount
        monthly_payment = Principal*(monthly_interest/(1-(1+monthly_interest)
                                                **(- payment_number)))
        # give payments made
        rem_bal = rem_bal = Principal * (1 - ((1 + monthly_interest) ** npaid - 1) /
                               ((1 + monthly_interest) ** payment_number - 1))

        return rem_bal
```

```
Stop3 = max(Loan.ID)
EAD = []
Loan_Provison = []
Loan_IRB_RWA = []
Loan_IRB_Cap = []
Loan_SA_RWA = []
Loan_SA_Cap = []

for index, row in Loan.iterrows():
  if row['ID'] <= Stop3:
    #Calculate EAD = outstanding balance
    EAD_V = Rem_Bal_Loan (row['OriginalAmount'],row['YearlyInterestRate'],
                          row['Term'],row['CurrentInstallment'])
    EAD.append(EAD_V)

    #Apply floors in LGD and PD
    adjusted_PD = max(PDfloor, row['PD'])
    adjusted_LGD = max(LGDfloor, row['LGD'])

    #Calculate the Provision
    provinsion = EAD_V * adjusted_LGD * adjusted_PD
    Loan_Provison.append(provinsion)

    #Calculate F-IRB RWA
    Cap_Req =  CapReq_other_Retail(adjusted_LGD,adjusted_PD)
    Cap_Req = Cap_Req * EAD_V
    RWA_V = 12.5 * Cap_Req
    Loan_IRB_RWA.append(RWA_V)


    #Calculate the IRB Capital Requirement
    Final_Cap_Req = RWA_V * cap_req_percent
    Loan_IRB_Cap.append(Final_Cap_Req)


    #Calculate Loan_weighed_RWA
    #Weight_Retail_Loan is a predefined constant 75%
    W_RWA_V = EAD_V * Weight_Retail_Loan
    Loan_SA_RWA.append(W_RWA_V)


    #Calculate the SA Capital Requirement
    Final_Cap_Req2 = W_RWA_V * cap_req_percent
    Loan_SA_Cap.append(Final_Cap_Req2)


  else:
    break


## Attach Saved result to data frame
Loan['EAD'] = EAD
Loan['Provision'] = Loan_Provison
Loan['IRB_RWA'] = Loan_IRB_RWA
Loan['SA_RWA'] = Loan_SA_RWA
Loan['Capital_Requirement_SA'] = Loan_SA_Cap
Loan['Capital_Requirement_IRB'] = Loan_IRB_Cap
```

1.2.5 Extract the Calculation Result

```
#Extract Provision and Capital Requirement under F-IRB Approach for Bond
```

```
print (sum(Bond['Bond_Provison']))
print (sum(Bond['Capital_Requirement_IRB']))


#Extract Provision and Capital Requirement under F-IRB Approach for Mortgages
print(sum(Mortgage['Provision']))
print(sum(Mortgage['Capital_Requirement_IRB']))



#Extract Provision and Capital Requirement under F-IRB Approach for Loans
print(sum(Loan['Provision']))
print(sum(Loan['Capital_Requirement_IRB']))
```

```
9858989107.772062
20092672282.9325
512264202.8959211
755794845.1596057
25753739.259888496
18518438.115086857
```

1.3 Calculate The RWA under IRB Approach and Standerized Approach

Because all th result was already calculated through the loop in Section 1.2, we just need to extract them out

```
##Calculate the Total RWA under IRB Approach
Total_Bond_IRB_RWA = sum(Bond.IRB_RWA)
Total_M_IRB_RWA = sum(Mortgage.IRB_RWA)
Total_Loan_IRB_RWA = sum(Loan.IRB_RWA)
Total_IRB_RWA = Total_Loan_IRB_RWA + Total_M_IRB_RWA + Total_Bond_IRB_RWA
print(Total_IRB_RWA)
print((Total_IRB_RWA* 11.5/100))

##Calculate the Total RWA under SA Approach
Total_Bond_SA_RWA = sum(Bond.SA_RWA)
Total_M_SA_RWA = sum(Mortgage.SA_RWA)
Total_Loan_SA_RWA = sum(Loan.SA_RWA)
Total_SA_RWA = Total_Bond_SA_RWA + Total_M_SA_RWA + Total_Loan_SA_RWA
print (Total_SA_RWA)
print((Total_SA_RWA * 11.5/100))
```

```
181452048401.8026
20866985566.2073
44443105660.24057
5110957150.927666
```