Create a mortgage calculator using the pseudocode below:

Create a MortgageCalculator class  
 Create header for the main method   
 declare and initialize an array to hold 3 loan terms - 7, 15, 30 using

member initializer list  
 declare an array to hold current interest rates for 3 loan terms  
   
 display the message: "Welcome to Mortgage Calculator Program"  
 make a call to method getCurrentRates pass as arguments

the term of loan array, and current rate array

make a call to method getTermInformation pass as arguments

the term of loan array, and current rate array

assign the output of the getTermInformation to an integer variable called index  
   
 declare a reference to a Scanner object and call it input   
 display the message "Enter Principle Amount: "  
  
 read the principle amount from the keyboard and store in a variable principle  
   
 make a call to a method displayLoanSummary pass as arguments principle,   
 element of the term loan array using index above,   
 element of current rate array using index above  
   
 make a call to a method displayAmortTable pass as arguments principle,   
 element of the term of loan array using index above,   
 element of current rate array using index above

/\* This method will input the interest rates for each of the 7, 15 and 30 year terms 0f loans  
 and assign specific values to the elements of the rates array \*/

Create a method getCurrentRates which returns a void and accepts as input parameters

an array to hold terms of loan,

an array of rates of loan

declare a reference to a Scanner object and call it input  
 Create a Loop to iterate the number of elements in the terms of loan array which was

received as input parameter

display a message "Enter current rate of # of years"

--> # is the value of the terms of loan array

read the rate and assign to each of the element of the rate of loan array  
   
 /\* This method will display all the 3 choice and ask user to select an option from the 3 choices  
 it will then return the index of the option selected \*/   
 Create a method getTermInformation which returns an int and accepts as input parameters

an array of terms of loan

an array of rates of loan

declare an int variable to hold a choice

Loop while true  
 display a message "Welcome to XYZ Bank!"  
 Loop for the length of terms of loan array  
 display sequence number, element of the terms of loan array, and

element of the rates of loan array  
   
 declare a reference to a Scanner object and call it input  
 display the message "Select (1, 2 or 3) an option for the loan:"  
 read the option and assign to variable choice  
  
 if the choice is not 1, 2 or 3 then

displaying a message that the choice is not a valid option

continue to the loop

else break from the loop

return choice – 1

/\* This method will display the Summary of the loan including principle amount, the interest rate and term of the loan\*/  
Create a method to displayLoanSummary which accepts the principle amount, term of loan, rate of loan   
 display a message "Principle amount: $" outputting principle amount --> use printf   
 display a message "Int. Rate(%): " outputting rate of loan  
 display a message "Term of Loan (Months): " outputting the term multiplied by 12

ÏÏ Welcome to Mortgage Calculator Program  
ÏÏÏ  
Enter current rate for 7 years: 5.35  
Enter current rate for 15 years: 5.50  
Enter current rate for 30 years: 5,75  
ÏÏ  
Welcome to XYZ Bank!  
1: 7 years - 5.35%  
2: 15 years - 1.00%  
3: 30 years - 1.00%

Select (1, 2 or 3) an option for the loan:1  
Enter Principal Amount: 100000

Summary of Loan  
Principal Amount: 100000.00  
Int. Rate(%): 5.35  
Term of Loan (Months): 84

/\* This method will display the amortization table to show the monthly activity of the loan \*/  
Create a method displayAmortTable which accepts as input parameters

principle,

term of loan,

rate of loan

declare an integer variable to hold the term of loan month  
 declare double variables to hold monthly Interest rate,

monthly mortgage payment,

total mortgage

declare double variables to hold total interest,

total amount due,

interest paid,

principle amount paid

assign term of loan (input parameter) multiplied by 12 to variable term of loan month   
assign rate of loan (input parameter) divided by (12 \* 100) to variable monthly interest rate   
  
assign principle \* (monthly interest rate / (1-Math.pow((1+monthly interest rate), - term of loan month))) to variable monthly mortgage payment   
  
assign monthly mortgage payment multiplied by term of loan (input parameter) to variable total mortgage

assign total mortgage minus principle to variable total interest   
assign principle to variable total amount due   
   
display the header of the amortization table "Payment", "Principle", "Interest", "Balance" using "\n%-10s%-16s%-16s%-16s\n"

Loop for number of terms of loan month  
 assign total amount due \* monthly interest rate to variable interest paid   
 if interest paid is less than or equal to zero  
 assign zero to variable interest paid   
   
 assign monthly mortgage payment minus interest paid to variable

principle amount paid

assign total amount due minus principle amount paid to variable

total amount due  
  
 if total amount due is less than zero  
 assign zero to variable total amount due   
  
 display sequence, principle amount paid, interest paid, total amount due

using "%-10d$%-15.2f$%-15.2f$%-15.2f\n"

Payment Principle Interest Balance   
1 $984.06 $445.83 $99015.94   
2 $988.45 $441.45 $98027.49   
3 $992.86 $437.04 $97034.63   
4 $997.28 $432.61 $96037.35   
5 $1001.73 $428.17 $95035.62   
6 $1006.19 $423.70 $94029.43   
7 $1010.68 $419.21 $93018.75   
8 $1015.19 $414.71 $92003.56   
9 $1019.71 $410.18 $90983.85   
10 $1024.26 $405.64 $89959.59   
11 $1028.83 $401.07 $88930.76   
12 $1033.41 $396.48 $87897.35   
13 $1038.02 $391.88 $86859.33   
14 $1042.65 $387.25 $85816.68   
15 $1047.30 $382.60 $84769.39   
purposefully excluded rows 16 – 74, but those should be part of your output  
75 $1367.68 $62.21 $12586.81   
76 $1373.78 $56.12 $11213.03   
77 $1379.90 $49.99 $9833.13   
78 $1386.06 $43.84 $8447.07   
79 $1392.24 $37.66 $7054.84   
80 $1398.44 $31.45 $5656.40   
81 $1404.68 $25.22 $4251.72   
82 $1410.94 $18.96 $2840.78   
83 $1417.23 $12.67 $1423.55   
84 $1423.55 $6.35 $0.00

Each program should include comments that explain what each block of code is doing. Additionally, the programs should compile without errors, and run with the results described in the exercise. The following deductions will be made from each exercise if any of the following is incorrect or missing:

* Proper formatting [5 points]
* Proper names for classes and variables  [5 points]
* Comments [5 point]
* Program doesn't compile [ 10 points]
* Source code (java file) missing [ 10 points]
* Executable (class file) missing [10 points]
* Missing loop where a loop was required [5 points]

**TURN IN TO BLACKBOARD (i.e. submit)**:

A single zip file (**Project3.zip**) containing the following:

Source Code files – i.e. **MortgageCalculator.java,**

Class files (i.e. Java bytecode) - i.e. **MortgageCalculator.class,**