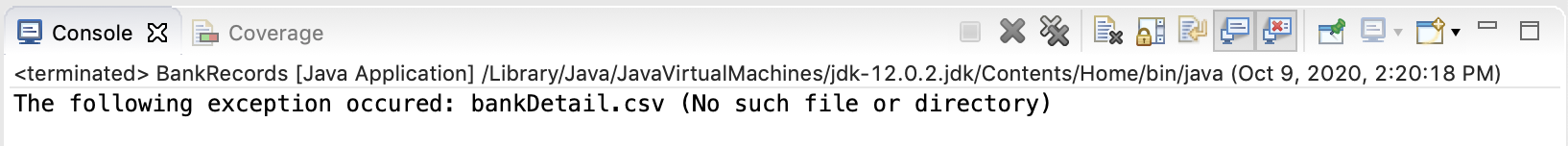
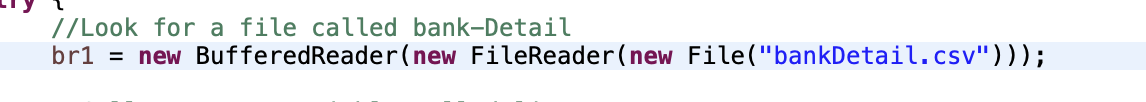
Snapshot 1: First 25 records printed in neat format

Table

Description automatically generated

Snapshot 2: Testing the try catch block to see the error catching for when a file is not found:



**Source Code**

**Client.java**

/\*

\* Cheryl Gardner

\* 09/29/2020

\* Lab 02-411

\* Purpose: To name the three methods that the bank needs to process

\* the names are: readData(), processData(), and printData()

\*/

//Declare a new abstract class called Client that can be extended to the other program

**public** **abstract** **class** Client{

//read all of the details of the file with all the information

**public** **abstract** **void** readData();

//process and store all of the information of each variable in arrays

**public** **abstract** **void** processData();

//print all of the data that was read and processed to the user

**public** **abstract** **void** printData();

}

**BankRecords.java**

/\*

\* Cheryl Gardner

\* 09/29/2020

\* ITMD411-Lab 2

\* Purpose: To write a program that will read, process and store the

\* data in variables and the print it very nicely in columns back to the

\* user

\*/

//Import the packages that will be used throughout the program

**import** java.io.\*;

**import** java.util.\*;

//Declare a new class called BankRecords that calls on the other class Client for the methods

**public** **class** BankRecords **extends** Client {

//Declare all of the variables needed

**private** String id;

**private** **int** age;

**private** String sex;

**private** String region;

**private** **double** income;

**private** String married;

**private** **int** children;

**private** String car;

**private** String save\_act;

**private** String current\_act;

**private** String mortgage;

**private** String pep;

//Declare a new array to import all of the data and then create a new ArrayList

**static** BankRecords *robjs*[] = **new** BankRecords[600];

**static** ArrayList<List<String>> *array1* = **new** ArrayList<>();

//Define the readData method that will be used to read all of the data from bank-Detail

**public** **void** readData() {

//Create a new Buffered Reader to go through the file

BufferedReader br1;

//Declare a new try block to do if the file is defined and there

**try** {

//Look for a file called bank-Detail

br1 = **new** BufferedReader(**new** FileReader(**new** File("bank-Detail.csv")));

//Call on a new variable called line1

String line1;

//Keep reading the file and adding to a list until they get to the very end

**while** ((line1 = br1.readLine()) != **null**) {

*array1*.add(Arrays.*asList*(line1.split(",")));

}

//Close the buffered reader

br1.close();

processData();

}

//If the file isn't present print the exception to the user

**catch** (Exception e){

System.***out***.println("The following exception occured: " + e.getMessage());

}

}

//Define the next abstract method called processData to go through all of the data

**public** **void** processData() {

//Declare a new variable to keep track of the count

**int** cag=0;

//Add all of the data into an array and get the data from each row and add it to variables to display

**for** (List<String> rowData: *array1*) {

*robjs*[cag] = **new** BankRecords();

*robjs*[cag].setterId(rowData.get(0));

*robjs*[cag].setterAge(Integer.*parseInt*(rowData.get(1)));

*robjs*[cag].setterSex(rowData.get(2));

*robjs*[cag].setterRegion(rowData.get(3));

*robjs*[cag].setterIncome(Double.*parseDouble*(rowData.get(4)));

*robjs*[cag].setterMarried(rowData.get(5));

*robjs*[cag].setterChildren(Integer.*parseInt*(rowData.get(6)));

*robjs*[cag].setterCar(rowData.get(7));

*robjs*[cag].setterSave\_Act(rowData.get(8));

*robjs*[cag].setterCurrent\_Act(rowData.get(9));

*robjs*[cag].setterMortgage(rowData.get(10));

*robjs*[cag].setterPep(rowData.get(11));

cag++;

}

//Call the printData() method

printData();

}

//Define the last abstract method to print all of the data for the user

**public** **void** printData() {

//Print the headings for each data field

System.***out***.println("ID\t\tAGE\t\tSEX\t\tREGION\t\tINCOME\t\tMORTGAGE");

//Keep going through and printing the data for the first 25 data fields

**for**(**int** i=0; i<25; i++) {

//Print all of the different fields

System.***out***.print(*robjs*[i].getterId()+"\t\t"+*robjs*[i].getterAge()+"\t\t"+*robjs*[i].getterSex()+"\t\t");

//Depending on the length of the region, print either one or two spaces after printing the region name

**if**(*robjs*[i].getterRegion().contentEquals("INNER\_CITY") || *robjs*[i].getterRegion().contentEquals("SUBURBAN")) {

System.***out***.print(*robjs*[i].getterRegion()+"\t");

}

**else** {

System.***out***.print(*robjs*[i].getterRegion()+"\t\t");

}

//Convert the int of Income to a String

String length = ""+ *robjs*[i].getterIncome();

//Depending on the length of the income, print either one or two spaces after the income

**if**(length.length() == 8) {

System.***out***.print(*robjs*[i].getterIncome()+"\t"+*robjs*[i].getterMortgage());

}

**else** {

System.***out***.print(*robjs*[i].getterIncome()+"\t\t"+*robjs*[i].getterMortgage());

}

//Print a new line after each so that all of the data is not on one line

System.***out***.println("");

}

}

//Declare all of the getter and setter methods that allow for the program to get all of the variables

**public** String getterId() {

**return** id;

}

**public** **void** setterId(String id) {

**this**.id = id;

}

**public** **int** getterAge() {

**return** age;

}

**public** **void** setterAge(**int** age) {

**this**.age = age;

}

**public** String getterSex() {

**return** sex;

}

**public** **void** setterSex(String sex) {

**this**.sex = sex;

}

**public** String getterRegion() {

**return** region;

}

**public** **void** setterRegion(String region) {

**this**.region = region;

}

**public** **double** getterIncome() {

**return** income;

}

**public** **void** setterIncome(**double** income) {

**this**.income = income;

}

**public** String getterMarried() {

**return** married;

}

**public** **void** setterMarried(String married) {

**this**.married = married;

}

**public** **int** getterChildren(){

**return** children;

}

**public** **void** setterChildren(**int** children) {

**this**.children = children;

}

**public** String getterCar() {

**return** car;

}

**public** **void** setterCar(String car) {

**this**.car = car;

}

**public** String getterSave\_Act() {

**return** save\_act;

}

**public** **void** setterSave\_Act(String save\_act) {

**this**.save\_act = save\_act;

}

**public** String getterCurrent\_Act() {

**return** current\_act;

}

**public** **void** setterCurrent\_Act(String current\_act) {

**this**.current\_act = current\_act;

}

**public** String getterMortgage() {

**return** mortgage;

}

**public** **void** setterMortgage(String mortgage) {

**this**.mortgage = mortgage;

}

**public** String getterPep() {

**return** pep;

}

**public** **void** setterPep(String pep) {

**this**.pep = pep;

}

//Call the main method that calls the BankRecords program

**public** **static** **void** main(String[] args) {

BankRecords brs = **new** BankRecords();

//Call the readData method to start the program

brs.readData();

}

}