**SourceCode**

Importing Library File For Java Class

Declaring A Scanner class Call it **scan** To Take Input From Keyword

Declaring A String To Store File Name Namely **FileName** And Assign A Null Value to It

Prompting The User To Enter The File Name

Taking The Input For The File Name

Creating the File Object To Read The File

Declaring A variable Of Scanner Type call it **ReadFile**

Declaring The Integer Type Variable namely **NumberOfBook** And Assign Value to It

Declaring A String Type Array To Store 50 String Value call it **SplitText**

Declaring A String Type Array To Store 50 String Value call it **BookTitle**

Declaring A String Type Array To Store 50 String Value call it **BookAuthor**

Declaring A String Type Array To Store 50 String Value call it **BookPublisher**

Declaring A Float Type Array To Store 50 Float Value call it **BookPrice**

Declaring A Integer Type Array To Store 50 Integer Value call it **BookPage**

Declaring A String Type Array To Store 50 String Value call it **BookISBN**

Declaring a Format To Align The Text call it **Format**

Declaring a Integer Type Variable call It **InvalidDelimeter**

Declaring a Integer Type Variable call It **InvalidData**

Using Exceptional Handling for File Exception

Try Begin

Searching the File

Declaring The Sting type Variable call It Line

Reading the Text File Using the While Loop

While Loop Begin

Assigning the Red Text To The Line Variable

Declaring A Delimiter to Split The Text File

If Condition Begin when SplitText.length is Smaller Than 6 or SplitText.length is Greater Than

Displaying message For The Wrong Delimiter in the Text File

Incrementing the InvalidDelimeter

If Condition End

Else Begin

Escaping the White Space From The Text File

If Begin when First SplitText Has the Null Value

Assigning the Value to The BookTitle

Incrementing the Number Of Invalid Data

If Ends

Else Begin

Assigning the Value of SplitText[0] to The BookTitle

Else End

If Begin when Second SplitText Has The Null Value

Assigning the Value to The BookAuthor

Incrementing the Value Of Invalid Data

If End

Else Begin

Assigning the Value of SplitText[1] to The BookAuthor

Else End

If Begin when Third SplitText Has The Null Value

Assigning the Value to The BookPublisher

Incrementing The Value Of Invalid Data

If Ends

Else Begin

Assigning the Value of SplitText[2] to The BookPublisher

Else Ends

Try Begin TO Check The Valid Data Red From The TextFile For BookPrice

Assigning the Value Of SplitText[3] to The BookPrice If Valid

Catch Begin For Number Formate Exception

Assigning the Value 0 To The BookPrice If Invalid Data is Red

Incrementing The Value Of Invalid Data

Catch Ends

Try Begin

Check The Valid Data Red From The TextFile For BookPage

Assigning the Value Of SplitText[4] to The BookPage If Valid

Catch Begin to Check The Number Formate Exception

Assigning the Value 0 To The BookPage If Invalid Data is Red

Incrementing The Value Of Invalid Data

Catch End

if Condition Begin When the SplitText Has The Null Value

Assigning the Value to The BookISBN

Incrementing The Value Of Invalid Data

If Condition End

Else Begin

Assigning the Value to The BookISBN

Else End

Else End

Try End

Catch Begin to Check The Input Output Exception

Displaying The Value if File Not Found

Program Exit If File Not Found

Catch Ends

While Ends

Displaying the Number Of Invalid Data

Displaying the Number Of Invalid Delimiter

Declaring the Integer Type Variable call it **NumberOfBookShown**

While Begin if Number Of Book To Be Shown is Smaller Than The Number Of Book Red From The File

Using The \n TO Align The Text Properly

Displaying the BookTitle

Displaying the BookPublisher

If Condition Begin When BookPrice Has 0 The Value

Displaying the Value for BookPrice

If Ends

Else Ends

Displaying the Value For the BookPrice

Else Ends

If Condition Begin When BookPage Has 0 The Value

Displaying the Value Of the BookPage

If Condition Ends

Else Begin

Displaying the BookPage

Else Ends

Displaying the Value For the Book ISBN

Incrementing the Number Of book Shown

Declaring the Integer Variable call it **TotalItem** and Assign 0 to it

Declaring the String type Variable to Take **AuthorName** and Assign Null Value To It

Prompting the User To Enter the Author Name

Taking the Value For the BookAuthor

While Loop Started

Loop Begin When Total Item Is Smaller Than The Length Of The BookAuthor

If Begin

Aligning the Data

Displaying the Value Of Book Title For Particular Author

Displaying the Value Of Book Title For Particular Author

Displaying the Value Of Book Title For Particular Author

Displaying the Value Of Book Title For Particular Author

Displaying the Value Of Book Title For Particular Author

Displaying the Value Of Book Title For Particular Author

If Condition Ends

Incrementing the Value For the Total Item Shown

While Ends

**JavaCode**

//Importing Library File For Java Class

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.Scanner;

**public** **class** Assessment {

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

//Declaring A Scanner class To Take Input From Keyword

@SuppressWarnings("resource")

Scanner scan= **new** Scanner(System.***in***);

//Declaring A String To Store File Name

String FileName=**null**;

//Prompting The User To Enter The File Name

System.***out***.print("Enter The File Name");

//Taking The Input For The File Name

FileName=scan.nextLine();

//Creating the File Object

File FileObject= **new** File(FileName);

//Declaring A variable Of Scanner Type

Scanner Readfile;

//Declaring The Integer Type Variable to Store the Number Of Book

**int** NumberOfBook=0;

//Declaring A String Type Array to Store the SplitText From The File

String []SplitText=**new** String[50];

//Declaring A String Type Array to Store the BookTitle From The File

String BookTitle[]=**new** String[50];

//Declaring A String Type Array to Store the BookAuthor From The File

String BookAuthor[]=**new** String[50];

//Declaring A String Type Array to Store the BookPublisher From The File

String BookPublisher[]=**new** String[50];

//Declaring A String Type Array to Store the BookPrice From The File

**float** BookPrice[]=**new** **float**[50];

//Declaring A String Type Array to Store the BookPage From The File

**int** BookPage[]=**new** **int**[50];

//Declaring A String Type Array to Store the BookISBN From The File

String BookISBN[]=**new** String[50];

//Declaring a Format to Align The Red Text

String Format="%-20s %-20s%n";

**int** InvalidDelimeter=0;

**int** InvalidData=0;

//Using Exceptional Handling For File Exception

**try** {

Readfile= **new** Scanner(FileObject);

//Declaring The Sting type Variable

String Line;

//Reading the Text File Using the While Loop

**while**(Readfile.hasNext()){

//Assigning the Red Text To The Line Variable

Line=Readfile.nextLine();

//Declaring A Delimiter to Split THe Text File

SplitText=Line.split("-");

//If Condition Begin when SplitText.lengthis Smaller Than 6 or SplitText.length is Grater Than 6

**if**(SplitText.length<6 || SplitText.length>6)

{

//Displaying message For The Wrong Delimiter in the Text File

System.***out***.println("\nWrong Delimmiter used or missing delimiter\n");

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

InvalidDelimeter++;

}

//If Condition End

//Else Begin

**else** {

//ESCAPING THE WHITE SPACE FROM THE TEXT FILE

**if** (SplitText.length ==1 && SplitText[0]=="")

**continue**;

//VALIDATING THE TITLE OF THE BOOK FROM THE TEXT FILE

//If Begin when First SplitText Has The Null Value

**if**(SplitText[0]=="") {

//Assigning the Value to The BookTitle

BookTitle[NumberOfBook]="[Book Title Missing]";

InvalidData++;

}

//If Ends

//Else Begin

**else** {

//Assigning the Value to The BookTitle

BookTitle[NumberOfBook]=SplitText[0].trim();

}

//VALIDATING THE BOOK AUTHOR FROM THE TEXT FILE

//If Begin when Second SplitText Has The Null Value

**if**(SplitText[1]=="") {

//Assigning the Value to The BookAuthor

BookAuthor[NumberOfBook]="[Author Name Missing]";

InvalidData++;

}

//If End

//Else Begin

**else** {

//Assigning the Value to The BookAuthor

BookAuthor[NumberOfBook]=SplitText[1].trim();

}

//VALIDATING THE BOOK PUBLISHER FROM THE TEXT FILE

//If Begin when Third SplitText Has The Null Value

**if**(SplitText[2]=="") {

//Assigning the Value to The BookPublisher

BookPublisher[NumberOfBook]="[Book Publisher Missing]";

InvalidData++;

}

//If Ends

//Else Begin

**else** {

//Assigning the Value to The BookPublisher

BookPublisher[NumberOfBook]=SplitText[2].trim();

}

//Else Ends

//Try Begin TO Check The Valid Data Red From The TextFile For BookPrice

**try** {

// Assigning the Value Of SplitText[3] to The BookPrice If Valid

BookPrice[NumberOfBook]=Float.*parseFloat*(SplitText[3].trim());

}**catch**(NumberFormatException e) {

//AssigniNg the Value O To The BookPrice If Invalid Data is Red

BookPrice[NumberOfBook]=0;

InvalidData++;

}

//Try Begin TO Check The Valid Data Red From The TextFile For BookPage

**try** {

// Assigning the Value Of SplitText[4] to The BookPage If Valid

BookPage[NumberOfBook]=Integer.*parseInt*(SplitText[4].trim());

}**catch**(NumberFormatException e) {

//AssigniNg the Value O To The BookPage If Invalid Data is Red

BookPage[NumberOfBook]=0;

InvalidData++;

}

//VALIDATING THE ISBN NUMBER FROM THE TEXT FILE

//if Condition Begin When the SplitText Has The Null Value

**if**(SplitText[5]=="") {

//Assigning the Value to The BookISBN

BookISBN[NumberOfBook]="[Book ISBN Missing]";

InvalidData++;

}

//If Condition End

// Else Begin

**else** {

//Assigning the Value to The BookISBN

BookISBN[NumberOfBook]= SplitText[5].trim();

}

//Else End

NumberOfBook++;

}

//Else End

}

//Try End

}**catch**(IOException e){

//Displaying The Value if File Not Found

System.***out***.println("File Not Found !!!");

System.*exit*(0);

};

//While Ends

System.***out***.println("Total Number Of Invalid Data "+InvalidData);

System.***out***.println("Total Number Of Invalid Delimeter "+InvalidDelimeter);

//Declaring the Integer Type Variable to Store the Number Of Book Shown

**int** NumberOfBookShown=0;

//While Begin if Number Of Book To Be Shown is Smaller Than The Number Of Book Red From The File

**while**(NumberOfBookShown<NumberOfBook){

//Using THe |n TO Align The Text Properly

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

//Displaying the BookTitle

System.***out***.printf(Format,"Title :",BookTitle[NumberOfBookShown]);

//Displaying the BookAuthor

System.***out***.printf(Format,"Author :",BookAuthor[NumberOfBookShown]);

//Displaying the BookPublisher

System.***out***.printf(Format,"Publisher :",BookPublisher[NumberOfBookShown]);

//If Condition Begin When BookPrice Has 0 The Value

**if** (BookPrice[NumberOfBookShown]==0) {

//Displaying the Value for BookPrice

System.***out***.printf(Format,"Price :","Book Price Missing Or Not A Valid Numeric Number");

}

//If Ends

//Else Ends

**else** {

//Displaying the Value For the BookPrice

System.***out***.printf(Format,"Price :",BookPrice[NumberOfBookShown]);

}

//Else Ends

//If Condition Begin When BookPage Has 0 The Value

**if** (BookPage[NumberOfBookShown]==0) {

//Displaying the Value Of the BookPage

System.***out***.printf(Format,"Page :","Book Page Missing Or Not A Valid Numeric Number");

}

//If Condition Ends

**else** {

//Displaying the BookPage

System.***out***.printf(Format,"Page :",BookPage[NumberOfBookShown]);

}

//Else Ends

//Displaying the Value For the Book ISBN

System.***out***.printf(Format,"ISBN :",BookISBN[NumberOfBookShown]);

System.***out***.printf("\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_\n\n");

NumberOfBookShown++;

}

//Declaring the INteger Variable to Store the Total Number Of Item Shown For the Particular Author

**int** TotalItem=0;

//Declaring the String type Variable to Take AuthorName

String AuthorName="" ;

//Prompting the User To Enter the Author Name

System.***out***.println(" Enter The Name Of The Author TO View His/Her Book");

//Taking the Value For the BookAuthor

AuthorName =scan.nextLine();

//While Loop Started

**while**(TotalItem<BookAuthor.length){

//If Condition Begin If the AuthorName Is Equal To Any OF THe BookAuthor Name

**if**(AuthorName.equals(BookAuthor[TotalItem])) {

//Aligning the Date

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

//Displaying the Value Of Book Title For Particular Author

System.***out***.printf(Format,"Title :",BookTitle[TotalItem]);

//Displaying the Value Of Book Title For Particular Author

System.***out***.printf(Format,"Author :",BookAuthor[TotalItem]);

//Displaying the Value Of Book Title For Particular Author

System.***out***.printf(Format,"Publisher :",BookPublisher[TotalItem]);

//Displaying the Value Of Book Title For Particular Author

System.***out***.printf(Format,"Price :",BookPrice[TotalItem]);

//Displaying the Value Of Book Title For Particular Author

System.***out***.printf(Format,"Page :",BookPage[TotalItem]);

//Displaying the Value Of Book Title For Particular Author

System.***out***.printf(Format,"ISBN :",BookISBN[TotalItem]);

System.***out***.printf("\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_\n\n");

}

//If Condition Ends

//Incrementing the Value For the Total Item Shown

TotalItem++;

}

//While Ends

}

}