```
/* Report.java - Prints a formatted table of Movie and Rental data.
   Author:   Brendan Kirby
   Module:   2
   Project:  1
   Problem Statement:  Uses classes Movie, Action, Drama, Comedy, and
                       Rental to generate a formatted table of the
                       information contained therein.
   Algorithm:

        DO
            PRINT questionPrompt
            READ response

            SWITCH (response)

                CASE 'N':
                CASE 'n':
                    isNotValid = false

                    PRINT tableHeader

                    FOR (int i = 0; i < rentals.length; i++)

                        IF (rentals[i].getMovie() instanceof Action)
                            genre = "Action"
                        ELSE IF (rentals[i].getMovie() instanceof Comedy)
                            genre = "Comedy"
                        ELSE IF (rentals[i].getMovie() instanceof Drama)
                            genre = "Drama"
                        ELSE IF (rentals[i].getMovie() instanceof Movie)
                            genre = "Movie"

                        PRINTF tableLine
                    BREAK

                CASE 'Y':
                CASE 'y':
                    isNotValid = false

                    PRINT  tableHeader

                    FOR (int i = 0; i < rentals.length; i++)

                        IF (rentals[i].getMovie() instanceof Action) THEN
                            genre = "Action"
                        ELSE IF (rentals[i].getMovie() instanceof Comedy) THEN
                            genre = "Comedy"
                        ELSE IF (rentals[i].getMovie() instanceof Drama) THEN
                            genre = "Drama"
                        ELSE IF (rentals[i].getMovie() instanceof Movie) THEN
                            genre = "Movie"
                        END IF

                        IF (rentals[i].getDaysLate() > 0) THEN
                            PRINTF tableLine
                        END IF
                    BREAK

                DEFAULT:
                    isNotValid = true
                    PRINT errorInvalidInput
                    BREAK

            END SWITCH

        WHILE (isNotValid)
```

```
        PRINT totalLateFees
*/

import java.util.Scanner;

public class Report {

    //method for formatting table output, concatenates enough spaces to fill the rest
    //of a given column on either the front or back end of the input string
    public static String fillSpace(String toFill, int totalSpace, boolean flipped) {

        int length, space;
        length = toFill.length();
        String emptySpace;

        emptySpace = "";
        space = (totalSpace - length);

        for (int i = 0; i < space; i++)
        {
            emptySpace = emptySpace + " ";
        }

        if (flipped) {

            return emptySpace + toFill;
        }
        else {

            return toFill + emptySpace;
        }
    }

    public static void main(String[] args) {

        //declaration
        Scanner key;
        char response;
        boolean isNotValid;
        Movie rental1, rental2, rental3, rental4, rental5, rental6;
        String inputTemp, lineFormat, column3, column4, column5, column6, genre;
        Rental[] rentals;

        //initialization
        rentals = new Rental[6];
        rental1 = new Movie("Bee Movie", 298765, "G");
        rental2 = new Movie("Planet Earth II", 102938, "G");
        rental3 = new Drama("Dramatic Lawyers", 654321, "PG-13");
        rental4 = new Drama("Dramatic Doctors", 123456, "PG-13");
        rental5 = new Action("Deadpool", 564738, "R");
        rental6 = new Comedy("23 Jump Street", 919293, "R");
        rentals[0] = new Rental(rental1, 7211, 3);
        rentals[1] = new Rental(rental2, 1993, 1);
        rentals[2] = new Rental(rental3, 5182, -3);
        rentals[3] = new Rental(rental4, 1729, 2);
        rentals[4] = new Rental(rental5, 3422, 7);
        rentals[5] = new Rental(rental6, 2112, 5);
        key = new Scanner(System.in);
        genre = "";
        lineFormat = "%6d"+"%7d"+"%-23s"+"%13s"+"%1s"+ "%1s" +"%-1s"+"%2d"+"%-1s"+"%5.2f";
        isNotValid = true;

```

```
131         //error checking user input
132         do {
133
134             //prompt & user input
135             System.out.print("Do you only want to see overdue movies? <y/n>: ");
136             inputTemp = key.next();
137             response = inputTemp.charAt(0);
138             System.out.print("\n\n\n");
139
140             //processing
141             switch (response) {
142
143                 case 'N':
144                 case 'n':
145                     isNotValid = false;
146
147                     System.out.print("Rental Customer                                  Movie     MPAA  Days    Late\n");
148                     System.out.print("  No.     ID        Movie Title         Class      ID     Rating Late    Fees\n");
149                     System.out.print("------ ------ -------------------- -------- ---------- ------ ----- --------\n");
150
151                     for (int i = 0; i < rentals.length; i++) {
152
153                         if (rentals[i].getMovie() instanceof Action) {
154
155                             genre = "Action";
156                         }
157                         else if (rentals[i].getMovie() instanceof Comedy) {
158
159                             genre = "Comedy";
160                         }
161                         else if (rentals[i].getMovie() instanceof Drama) {
162
163                             genre = "Drama";
164                         }
165                         else if (rentals[i].getMovie() instanceof Movie) {
166
167                             genre = "Movie";
168                         }
169
170                         column3 = rentals[i].getMovie().getTitle();
171                         column4 = genre;
172                         column5 = "" + rentals[i].getMovie().getIdNum();
173                         column6 = rentals[i].getMovie().getRating();
174
175                         System.out.printf(lineFormat,
176                                           (i+1),
177                                           rentals[i].getCustomerID(),
178                                           fillSpace(" " + column3, 24, false),
179                                           fillSpace(column4, 12, false),
180                                           column5,
181                                           " "
182                                           fillSpace(column6, 10, false),
183                                           rentals[i].getDaysLate(),
184                                           fillSpace("$", 4, true),
185                                           rentals[i].calcRentalLateFee());
186                         System.out.println();
187                     }
188                     break;
189
```

```
                case 'Y':
                case 'y':
                    isNotValid = false;

                    System.out.print("Rental Customer                                          Movie     MPAA  Days    Late\n");
                    System.out.print("  No.    ID        Movie Title          Class      ID     Rating Late    Fees\n");
                    System.out.print("------ ------ -------------------- -------- ---------- ------ ----- --------\n");

                    for (int i = 0; i < rentals.length; i++) {

                        if (rentals[i].getMovie() instanceof Action) {

                            genre = "Action";
                        }
                        else if (rentals[i].getMovie() instanceof Comedy) {

                            genre = "Comedy";
                        }
                        else if (rentals[i].getMovie() instanceof Drama) {

                            genre = "Drama";
                        }
                        else if (rentals[i].getMovie() instanceof Movie) {

                            genre = "Movie";
                        }

                        column3 = rentals[i].getMovie().getTitle();
                        column4 = genre;
                        column5 = "" + rentals[i].getMovie().getIdNum();
                        column6 = rentals[i].getMovie().getRating();

                        //only prints late rentals, per user input
                        if (rentals[i].getDaysLate() > 0) {

                            System.out.printf(lineFormat,
                                              (i+1),
                                              rentals[i].getCustomerID(),
                                              fillSpace("" + column3, 24, false),
                                              fillSpace(column4, 12, false),
                                              column5,
                                              " "
                                              fillSpace(column6, 10, false),
                                              rentals[i].getDaysLate(),
                                              fillSpace("$", 4, true),
                                              rentals[i].calcRentalLateFee());
                            System.out.println();
                        }
                    }
                    break;

                default:
                    System.out.println("Invalid input, try again.\n\n");
                    isNotValid = true;
                    break;
            }
        } while (isNotValid);

        //prints total late fees owed
        System.out.print("\n\n");
        System.out.printf("%73s"+"%5.2f", "Total late fees to collect: $ ", Rental.lateFeesOwed(rentals));
    }
}
```