record management system using java programming

Introduction to Programming – 6G4Z1901



Jordan Prescott

Student ID - 20060725

Contents

[Summary 3](#_Toc71374565)

[Program Specification 3](#_Toc71374566)

[Program Design 3](#_Toc71374567)

[Main Menu 4](#_Toc71374568)

[[1] Create 4](#_Toc71374569)

[[2] Edit 4](#_Toc71374570)

[[3] Delete 5](#_Toc71374571)

[[4] Search 5](#_Toc71374572)

[[5] Display All 5](#_Toc71374573)

[[6] Decade 5](#_Toc71374574)

[[7] Shuffle 5](#_Toc71374575)

[[8] Exit 6](#_Toc71374576)

[Menu Tree’s 6](#_Toc71374577)

[Main Menu 6](#_Toc71374578)

[Create 7](#_Toc71374579)

[Edit 7](#_Toc71374580)

[Delete 8](#_Toc71374581)

[Search 9](#_Toc71374582)

[User Guide/ Interface Design 9](#_Toc71374583)

[Program Boot 9](#_Toc71374584)

[Program Start 9](#_Toc71374585)

[Create A Record 10](#_Toc71374586)

[Edit a Record 11](#_Toc71374587)

[Delete a Record 13](#_Toc71374588)

[Delete a single record 13](#_Toc71374589)

[Delete in Bulk 14](#_Toc71374590)

[Search 15](#_Toc71374591)

[ID 15](#_Toc71374592)

[Title 16](#_Toc71374593)

[Artist 17](#_Toc71374594)

[Year 18](#_Toc71374595)

[Decade 19](#_Toc71374596)

[Genre 20](#_Toc71374597)

[Duplicates 21](#_Toc71374598)

[Display All 22](#_Toc71374599)

[Decades 23](#_Toc71374600)

[Shuffle Pick 24](#_Toc71374601)

[Exit 25](#_Toc71374602)

[Exiting Functions 25](#_Toc71374603)

[Exit Program 25](#_Toc71374604)

[Function Path Codes 25](#_Toc71374605)

[Class Design 26](#_Toc71374606)

[Method/ Variable Design 26](#_Toc71374607)

[Method - Name 26](#_Toc71374608)

[Class - Program 26](#_Toc71374609)

[Method – Main 26](#_Toc71374610)

[Method – Write 28](#_Toc71374611)

[Class – Create 28](#_Toc71374612)

[Method – Create 28](#_Toc71374613)

[Method – createRecord 29](#_Toc71374614)

[Method - newId 29](#_Toc71374615)

[Method – thisYear 29](#_Toc71374616)

[Class – Edit 30](#_Toc71374617)

[Method – Edit 30](#_Toc71374618)

[Method – editRecord 30](#_Toc71374619)

[Method – deleteRecord 31](#_Toc71374620)

[Method – deleteInBulk 31](#_Toc71374621)

[Class – Display 32](#_Toc71374622)

[Method – Display 32](#_Toc71374623)

[Method – displayAll 32](#_Toc71374624)

[Method – shuffleByID 32](#_Toc71374625)

[Method – displayByDecade 33](#_Toc71374626)

[Method - displayByID 33](#_Toc71374627)

[Method - displayByTitle 34](#_Toc71374628)

[Method - displayByArtist 34](#_Toc71374629)

[Method - displayByYear 34](#_Toc71374630)

[Method - displayByDecade 35](#_Toc71374631)

[Method - displayByGenre 35](#_Toc71374632)

[Method – displayDuplicates 35](#_Toc71374633)

[Class – VinylRecord 36](#_Toc71374634)

[Method – VinyRecord 36](#_Toc71374635)

[Method – **get***Element* 36](#_Toc71374636)

[Method – **set***Element* 36](#_Toc71374637)

[Method – toString 37](#_Toc71374638)

[Version History 37](#_Toc71374639)

[Version 1.0.1 37](#_Toc71374640)

[Bugs Fixed 37](#_Toc71374641)

[Additional Features 37](#_Toc71374642)

[Version 1.0.2 37](#_Toc71374643)

[Bugs Fixed 37](#_Toc71374644)

[Additional Features 37](#_Toc71374645)

[Version 1.0.3 38](#_Toc71374646)

[Bugs Fixed 38](#_Toc71374647)

[Additional Features 38](#_Toc71374648)

[Version 1.0.3 Testing 38](#_Toc71374649)

# Summary

This program is a record management system designed to manage album/ vinyl collections using a Terminal User Interface (TUI). The user can create entries within the application, or they can load them using the CSV file. Each record has five characteristics, and they are ID, this is a unique identifier specific for each record. Title, this is the title of the album e.g., ‘Abbey Road’. Artist, this is the artist’s name that produced the album e.g. ‘The Beatles’. Year, this is the year the album was released e.g., ‘1969’ and finally the Genre of the record being the style of music e.g., ‘Rock’.

When records are in the program, they can then be managed using the functions built in. They are Create, Edit, Delete, Search, Display, Decade, and Shuffle. The programs main functions are the first five in this list and with the latter being extra features for user enjoyment.

# Program Specification

For users to manage records efficiently the system is required to provide multiple functions that can assist. The below are built into the system, for more information please find the section ‘Program Design’.

1. User Input – Allow the user to interact with the system.
2. Create – The ability to create records.
3. Edit - Change records details.
4. Searchable – Search for records using key criteria.
5. Display Information – Display record details in multiple ways.
6. Delete – Permanently remove records from system.

# Program Design

The program is divided into eight core functions that perform specific tasks that all span from the main menu. The main menu is displayed for users at start up to select the function they would like and when a task is completed or quit this is where the program is returned to other than Exit which will close the program. The programs overview is displayed in the below diagram, here you can see all functions spanning the main menu.

The diagram is split into functions colored Red and Blue. Red means they are a parent function and have child operations/ interactions needed to complete their purpose. Blue means parent only and when selected will require no further interactions to display or perform what is required. With each Parent/Child function a menu is displayed to the TUI stating the user will need key information to perform the task and if they do not have this, they can exit their current state to find this info.

Chart, box and whisker chart

Description automatically generated

## Main Menu

The main menu is the anchor point of this program, this is placing the system starts after boot and where it returns after a function is completed or quit. This displays all options to the TUI for the user to them select by inputting the corresponding number with the option. When selected the program will then drop into the function’s submenu (If Parent/ Child) or the data/operation (If Parent Only) will be displayed/performed.

## [1] Create

The create function when completed and all input is entered correctly creates a new record. When selected a submenu is displayed giving the user the option to continue and create the record or exit this state. When continued it will then display to the TUI what information the system requires from the user to create a record and validates this when inputted. When completed indication that the record has successfully been created to the TUI.

## [2] Edit

The edit function allows the user to modify certain details of an album. When selected a submenu is displayed to the TUI informing the user, they will need the ID number of the record they wish to edit and if they do not have this, they can exit here to find it. When continued the system will indicate its ready for the ID number from the user. This will then check for this ID and pull the records details, if no ID was matched a message is shown and the Edit function will return to the Main Menu.

If the ID entered matches a record the system will display the details records it will then ask user if they would like to modify an element of the record starting with Title. Once all elements that can be modified have been worked through the new details are presented and indicated that they are saved.

|  |  |
| --- | --- |
| Element | Can be Modified |
| ID | **N** |
| Title | **Y** |
| Artist | **Y** |
| Year | **Y** |
| Genre | **Y** |

## [3] Delete

The delete function permanently removes a record from the system. When selected a submenu is displayed to the TUI informing the user, they will need the ID number of the record they wish to edit and if they do not have this, they can exit here to find it. There are two delete functions the first deleting a single record and the second deleting in bulk.

When deleting a single record, the program will indicate its ready for the ID number from the user. This will then check for this ID and pull the records details, if no ID was matched a message is shown and the Delete function will return to the Main Menu. If the ID matches a record, it will then display the records details and confirm the user is happy to delete this. If the user commits this then the record is removed and if not, the system returns to the main menu.

When deleting in bulk the program will continue to prompt the user to enter an ID until they answer “N” for no. This will then look at the list of IDs the user has entered and run this through the existing list of records, if found they are deleted and indicated to the TUI for the user.

## [4] Search

The search function displays record information based on search criteria inputted from the user. When selected a submenu is displayed showing the criteria that the user used to search for the record. When an option has been chosen, a further submenu is displayed allowing giving the users to continue or quit if they do not have the correct information.

The searchable criteria can be categorized by Single or Multiple. This is the number of records that search can offer if the data is available. Please see below for categorized searches.

|  |  |
| --- | --- |
| Criteria | Single/ Multiple |
| ID | **S** |
| Title | **M** |
| Artist | **M** |
| Year | **M** |
| Genre | **M** |
| Duplicates | **M** |

## [5] Display All

The display all function displays every records detail to the TUI. When selected the user can see all record information at their own time. The system will pause and wait for the user to enter before returning to the main menu.

## [6] Decade

The decade function creates a timeline of all the records on system. When selected this will display all records on the system in each decade they were released. The information shown here is Title and Artist only placed under each decade header. The user can view this at their own time as the system will pause and wait for the user to enter before returning to the main menu.

## [7] Shuffle

The shuffle function will display a randomly chosen records details to the TUI. When selected the program will display ‘What will you listen to today’ followed by displaying “.” five times with a pause in between to add suspense for the user. After this the randomly generated record will be displayed and the system will return to the main menu.

## [8] Exit

The exit function will display the CSV export path and close the program. When selected the path of the export will be displayed to the TUI and the program will move to the closing menu. Once here it will display that the system is saving all records and then that it is now closed.

# Menu Tree’s

## Main Menu

A picture containing text

Description automatically generated

## Create

Diagram

Description automatically generated

## Edit

Diagram

Description automatically generated

## Delete

Diagram

Description automatically generated

## Search

Chart

Description automatically generated with low confidence

# User Guide/ Interface Design

## Program Boot

When the program starts the start, menu is displayed.

Text

Description automatically generated

## Program Start

When the program moves from the boot phase the main menu will be displayed. This is the anchor point of the program and where it will return when a function is completed or quit.

Text

Description automatically generated

## Create A Record

1. First choose option ‘[1] Create’ by entering ‘1’.
2. When you enter the below submenu choose option ‘[1] Create Album’.

Text

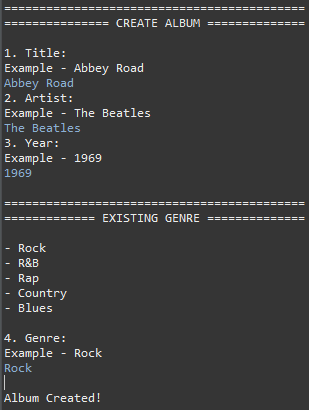
Description automatically generated

1. You will now be in the function of creating a record and you will need to complete this to return to the main menu. Enter the record detail in each section and press enter to commit it.
   1. Title
   2. Artist
   3. Year
   4. Genre

Text

Description automatically generated

1. After this the system will confirm your album is now saved.



1. When the album is created you, the program will go back to the Create submenu, here you can create another or save and exit your new records

Text

Description automatically generated

## Edit a Record

1. First choose option ‘[2] Edit’ by entering ‘2’.
2. When you enter the below submenu choose ‘[1] Edit Record’ to enter the ID of the record.

Text

Description automatically generated

1. You will then need to enter your record ID like below. You are now in the function and will need to complete this to return to the main menu.

Text

Description automatically generated

1. The current record details will be displayed, and you will be asked if you would like to edit this element. To change an element details, enter ‘Y’ like the example below.

Elements:

* 1. Title
  2. Artist
  3. Year
  4. Genre

Text

Description automatically generated

1. You will then be redirected to the main menu.

## Delete a Record

### Delete a single record

1. First choose ‘[3] Delete’ from main menu by entering ‘3’.
2. When you enter the below submenu choose ‘[1] Delete Record’ to enter the ID of the record.

Text

Description automatically generated

1. You will then need to enter your record ID like below. You are now in the function and will need to complete this to return to the main menu.

Text

Description automatically generated

1. The details of your chosen record will be displayed and displays a message to confirm you are sure you would like to delete this record.

Text

Description automatically generated

1. You will have confirmation the record is deleted and returned to the main menu.

### Delete in Bulk

1. First choose ‘[3] Delete’ from main menu by entering ‘3’.
2. When you enter the below submenu choose ‘[2] Delete in Bulk’ to enter the ID of the records.

Text

Description automatically generated

1. You will need to complete this function to return to the main menu. If you do not wish to continue enter “N” straight away.

When in the below you are prompted to enter an ID of the record enter “Y” to do this.

Text

Description automatically generated

1. Enter the ID you want to delete, you will then be promoted if you would like to delete another ID example below.

Text

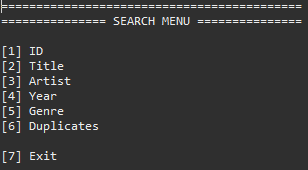
Description automatically generated

1. When you are done enter “N” and the program will confirm the albums are deleted.

## Search

### ID

1. First choose ‘[4] Search’ from the main menu by entering ‘4’
2. Then choose ‘[1] ID’ by entering ‘1’ in the below submenu.



1. In the below menu choose ‘[1] Enter ID’ by entering ‘1’.

Text

Description automatically generated

1. You will need to enter the ID of the record you are searching for. You are now in the function and you will need to complete this to return to the main menu.

Text

Description automatically generated

1. The records details will then be displayed, and you will be returned to the main menu.

Text

Description automatically generated

### Title

1. First choose ‘[4] Search’ from the main menu by entering ‘4’
2. Then choose ‘[2] Title’ by entering ‘2’ in the below submenu.

Graphical user interface, text

Description automatically generated

1. In the below menu choose ‘[1] Enter Title’ by entering ‘1’.

Text

Description automatically generated

1. You will need to enter the Title of the record you are searching for. You are now in the function and you will need to complete this to return to the main menu.

Text

Description automatically generated

1. The records details will then be displayed, and you will be returned to the main menu.

Text

Description automatically generated

### Artist

1. First choose ‘[4] Search’ from the main menu by entering ‘4’
2. Then choose ‘[3] Artist’ by entering ‘3’ in the below submenu.

Graphical user interface, text

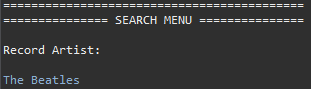
Description automatically generated

1. In the below menu choose ‘[1] Enter Artist’ by entering ‘1’.

Text

Description automatically generated

1. You will need to enter the Artist of the record you are searching for. You are now in the function and you will need to complete this to return to the main menu.



1. The records details will then be displayed, and you will be returned to the main menu. If you have multiple records from this artist, you will see all entries.

Text

Description automatically generated

### Year

1. First choose ‘[4] Search’ from the main menu by entering ‘4’
2. Then choose ‘[4] Year’ by entering ‘4’ in the below submenu.

Graphical user interface, text

Description automatically generated

1. In the below menu choose ‘[1] Enter Year’ by entering ‘1’.

Text

Description automatically generated

1. You will need to enter the Year of the record you are searching for. You are now in the function and you will need to complete this to return to the main menu.

Graphical user interface, text

Description automatically generated

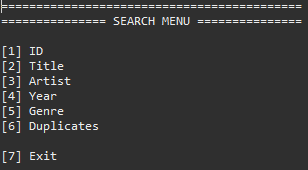
1. The records details will then be displayed, and you will be returned to the main menu. If you have multiple records from this year, you will see all entries.

Text

Description automatically generated

### Decade

1. First choose ‘[4] Search’ from the main menu by entering ‘4’
2. Then choose ‘[4] Year’ by entering ‘4’ in the below submenu.



1. In the below menu choose ‘[2] Enter Decade by entering ‘1’.

Text

Description automatically generated

1. You will need to enter the Decade of the record you are searching for. You are now in the function and you will need to complete this to return to the main menu.

Text

Description automatically generated

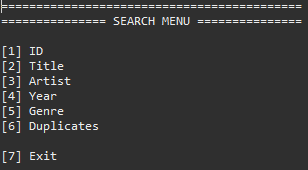
1. The records details will then be displayed, and you will be returned to the main menu. If you have multiple records from this decade, you will see all entries.

Text

Description automatically generated

### Genre

1. First choose ‘[4] Search’ from the main menu by entering ‘4’
2. Then choose ‘[5] Genre’ by entering ‘5’ in the below submenu.



1. In the below menu choose ‘[1] Enter Genre’ by entering ‘1’.

Text

Description automatically generated

1. You will need to enter the Genre of the record you are searching for. You are now in the function and you will need to complete this to return to the main menu.

Text

Description automatically generated

1. The records details will then be displayed, and you will be returned to the main menu. If you have multiple records from this genre, you will see all entries.

Text

Description automatically generated

### Duplicates

1. First choose ‘[4] Search’ from the main menu by entering ‘4’
2. Then choose ‘[6] Duplicates’ by entering ‘6’ in the below submenu.

Graphical user interface, text

Description automatically generated

1. This will then display all duplicate records. User will need to press enter to return to the main menu.

Text

Description automatically generated

## Display All

1. Choose ‘[5] Display All’ from the main menu by entering ‘5’.
2. All albums will they be displayed.

Text

Description automatically generated

1. Press enters to return to the main menu.

## Decades

1. Choose ‘[6] Decades’ from the main menu by entering ‘6’.
2. All albums in a timeline output will be shown in what decade they are in.

Text

Description automatically generated

1. Press enters to return to the main menu.

## Shuffle Pick

1. Choose ‘[7] Shuffle Pick’ from the main menu by entering ‘7’.
2. The program will then display a message followed my five loading ‘.’.



1. The randomly picked record will then be displayed to the TUI

Text

Description automatically generated

1. You will then return to the main menu.

## Exit

### Exiting Functions

Throughout the program you will be able to exit before entering the function chosen. You will see a menu with option one to enter the function and option 2 to exit back to the main menu like the one below.

1. To exit from the function, choose ‘[2] Exit’ by entering ‘2’. Please note when you enter the function you will need to complete this to return to the main menu.

Text

Description automatically generated with medium confidence

### Exit Program

1. Choose ‘[8] Exit’ from the main menu by entering ‘8’.
2. The program will then display the CSV export path to the TUI for the user.

Graphical user interface, text

Description automatically generated

1. The program will them move to the Closing Page displaying the system is closing, saving user records, and then closed.

Text

Description automatically generated with medium confidence

# Function Path Codes

*All codes below start from the main menu in the program.*

|  |  |
| --- | --- |
| Function | Path Code |
| Create Record | 1-1 |
| Edit Record | 2-1 |
| Delete Record | 3-1 |
| Delete in Bulk | 3-2 |
| Search by ID | 4-1-1 |
| Search by Title | 4-2-1 |
| Search by Artist | 4-3-1 |
| Search by Year | 4-4-1 |
| Search by Decade | 4-4-2 |
| Search by Genre | 4-5-1 |
| Search by Duplicates | 4-6 |
| Display All | 5 |
| Decades | 6 |
| Shuffle Pick | 7 |
| Exit | 8 |

# Class Design

Graphical user interface, application

Description automatically generated

# Method/ Variable Design

**TEMPLATE HERE**

### Method - Name

|  |  |
| --- | --- |
| Purpose |  |
| Pseudocode |  |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
|  |  |

## Class - Program

### Method – Main

|  |  |
| --- | --- |
| Purpose | Starting point of the program, this will read in all existing records from the CSV and create instances of them. It is then link to all the functions the user would like to perform. The displaying of the menus will guide the user to the action they want to do and then when they confirm this is the correct action the program calls the correct function. After every function it will return to the main menu and if the user decides to exit it will enter the exit stage writing the records to the CSV and displaying the location. |
| Pseudocode | 1. First read the lines in the file path Data/VinylRecords. 2. With each line create a new instance in the array list. 3. The program will now have the existing records. 4. Start the program and show something to the TUI its starting. 5. Enter the main menu. 6. Take in user input for choice of submenu. 7. Display submenu options. 8. Take in user input for choice in submenu. 9. Take in some input to confirm details of record they wish to perform action on. 10. Call the correct method with the input taken from user. 11. After method completion/ exit return to the main menu. 12. Repeat the above unless exit is selected from main menu. 13. If exit is chosen to leave the do while loop for the main menu. 14. Write the array list to the CSV. 15. Display the location path to the CSV. 16. Display the program is closing. 17. End. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| Scanner read | Reads in the file |
| Scanner in | Stores users input for menu selection and some functions |
| Boolean exit | The condition for the do While loop that contains the main menu |
| Int userAnswer | Stores user’s choice from main menu |
| Int createMenu1 | Stores user’s choice from create menu |
| Boolean exitCreate | The condition used to control the loop for creating records |
| Create createRecord | Instance of create used to pass in the global array list |
| Int userAnswer | Stores the users answer for the create submenu |
| Int editMenu1 | Stores user’s choice from exit menu |
| Int recordID | Stores the integer of the ID the users’ inputs |
| Display displayRecordID | Instance of display to pass in the global array list |
| Edit editRecord | Instance of edit to pass in the global array list |
| Int deleteMenu1 | Stores users answer for delete menu |
| Display deleteRecordID | Instance of display to pass in the global array list |
| Edit deleteRecord | Instance of edit to pass in the global array list |
| Edit bulkDelete | Instance of edit to pass in the global array list |
| Display searchRecord | Instance of edit to pass in the global array list |
| Int searchMenu1 | Stores user’s choice to submenu |
| Int idSubMenu | Stores user’s choice to submenu |
| Display displayRecordID | Instance of display to pass the global array list |
| Int titleSubMenu | Stores user’s choice to submenu |
| String recordTitle | Stores the users title they are searching for |
| String recordTitle | Stores the user’s artist they are searching for |
| Int yearSubMenu | Stores user’s choice to submenu |
| Int recordYear | Stored outside of the try catch so its accessible to the function call. This stores the year they are searching for |
| String recordDecade | Stores the user’s decade they are searching for |
| Int genreSubMenu | Stores user choice for submenu |
| String recordGenre | Stores users input of genre they are searching for |
| Display displayDuplicates | Instance of Display to pass in global variable |
| Display decadesOfRecord | Instance of Display to pass in global variable |
| Display shuffleRecord | Instance of Display to pass in global variable |
| File file | File reader to take in Data/VinyRecords.csv |

### Method – Write

|  |  |
| --- | --- |
| Purpose | Writes all the all instances in the global array list into the csv file stored in Data/VinyRecods.csv |
| Pseudocode | 1. Open the file. 2. Write headers to the file for each variable of Viny Record. 3. Loop through all records and print the toString to the file. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| PrintWriter file | Creates print writer for editing the file |

## Class – Create

#### Global Variables

|  |  |
| --- | --- |
| Name | Purpose |
| ArrayList vinylsList | Stores instances of vinyl records |

### Method – Create

|  |  |
| --- | --- |
| Purpose | Constructor method to pass in the array list ready for use by internal methods. |
| Pseudocode | 1. Take in the global array list. 2. Store the list as a global variable here for methods to access. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| This.vinylRecords = vinylsList | Stores instances of vinyl records |

### Method – createRecord

|  |  |
| --- | --- |
| Purpose | Create an instance of record by taking in the criteria from the user. This should ask the user for specific data when all together is a record and then use this to create a record and store this in the global array list. |
| Pseudocode | 1. Generate a new record ID and store this. 2. Store user input for Title. 3. Store user input for Artist. 4. Store user input for Year. 5. Store user input for Genre. 6. Create new instance of Vinyl Record. 7. Add new instance to global array list. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| Int newId | Stores the generated new ID for the new record |
| String newTitle | Stores the input for title from user |
| String newArtist | Stores the input for artist from user |
| Int newYear | Stores the input for year from user |
| String newGenre | Stores the input for genre from user |
| Display displayRecordGenre | New instance to pass in global array |
| VinylRecord newVinylRecord | New instance of vinyl record |

### Method - newId

|  |  |
| --- | --- |
| Purpose | The purpose of this is to generate a new ID for the new record when creating. Record ID’s will increment by 1 starting at 1 so this record simply needs to collect the latest ID and continue the sequence. |
| Pseudocode | 1. Read all records. 2. Store each record ID in a list. 3. Look at the last ID stored in the list. 4. Add one to the last ID. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| ArrayList <Integer> albumIDs | Store’s integers of the IDs |

### Method – thisYear

|  |  |
| --- | --- |
| Purpose | Return the current year is YYYY format. |
| Pseudocode | 1. Take in today’s date. 2. Format the date to show the year as YYYY. 3. Return the year in correct format. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| Date currentDate | New instance of date taking in today’s date |
| SimpleDateFormat dateFormat | Formats the date to the style the program needs which is YYYY |
| Int n | Parses the formatted year into an integer |

## Class – Edit

#### Global Variables

|  |  |
| --- | --- |
| Name | Purpose |
| ArrayList vinylsList | Stores instances of vinyl records |

### Method – Edit

|  |  |
| --- | --- |
| Purpose | Constructor method to pass in the array list ready for use by internal methods. |
| Pseudocode | 1. Take in the global array list. 2. Store the list as a global variable here for methods to access. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| This.vinylRecords = vinylsList | Stores instances of vinyl records |

### Method – editRecord

|  |  |
| --- | --- |
| Purpose | Edit a records criterion, prompt the user if they would like to edit a certain piece of data about the record, if yes then take in the new data and adjust the record. |
| Pseudocode | 1. Take in a record ID. 2. Loop through the records. 3. If the ID matches prompt the user if they would like to edit below. 4. Title 5. Artist 6. Year 7. Genre 8. Update the records details to the new data. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| Int searchableID | Stores the ID passed to the method as a local variable for use |
| Scanner in | Takes in the scanner passed from the program for use in this method |
| String Title | Stores user answer which is the condition for the if statement |
| String Artist | Stores user answer which is the condition for the if statement |
| String Year | Stores user answer which is the condition for the if statement |
| String Genre | Stores user answer which is the condition for the if statement |

### Method – deleteRecord

|  |  |
| --- | --- |
| Purpose | Delete a record chosen by the user. Prompt the user to confirm they are happy before they delete the record. |
| Pseudocode | 1. Take in the ID of record. 2. Loop through the records. 3. If match is found confirm the user would like to delete this record. 4. If yes, then delete the record. 5. If no cancel deletion. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| Scanner in | Takes in the scanner passed from the program for use in this method |
| Int deleteID | Stores the ID passed to the method as a local variable for use |
| Int index | If the match is found the index of the record in the array list is stored here |

### Method – deleteInBulk

|  |  |
| --- | --- |
| Purpose | Delete record in bulk. Ask user for ID and that they would like to enter another until they do not. When they do not delete the record, they have chosen. |
| Pseudocode | 1. Ask user if they would like to delete a record. 2. If yes, ask for ID If no cancel the deletion. 3. Store the ID user enters in a list. 4. Repeat asking until they say no. 5. Then remove the IDs chosen. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| Scanner in | Stores the scanner passed by program |
| Boolean again | Controls the condition of the do While loop |
| ArrayList <Integer> idList | Stores the IDs the user enters |
| ArrayList <Integer> found | Stores the IDs that match what the user entered |
| String userAnswer | Stores the users answer if they would like to continue |

## Class – Display

#### Global Variables

|  |  |
| --- | --- |
| Name | Purpose |
| ArrayList vinylsList | Stores instances of vinyl records |

### Method – Display

|  |  |
| --- | --- |
| Purpose | Constructor method to pass in the array list ready for use by internal methods. |
| Pseudocode | 1. Take in the global array list. 2. Store the list as a global variable here for methods to access. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| This.vinylRecords = vinylsList | Stores instances of vinyl records |

### Method – displayAll

|  |  |
| --- | --- |
| Purpose | Displays all the existing records and their details in a list to the TUI. |
| Pseudocode | 1. Loop through each record. 2. Display the record to the TUI in a formatted way. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| N/A | N/A |

### Method – shuffleByID

|  |  |
| --- | --- |
| Purpose | Randomly chooses a record from the existing list and displays that records details to the TUI. |
| Pseudocode | 1. Loop through the records and store all the IDs in a list. 2. Randomly choose one of the IDs in list. 3. Return the ID of the randomly selected record. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| ArrayList <Integer> albumIDs | Stores record IDs |
| Int shuffleID | Stores the integer of the randomly generated between the size of the list. This will give the index of the random ID and it will then be returned. |

### Method – displayByDecade

|  |  |
| --- | --- |
| Purpose | Display a timeline of the existing records in their decade to the TUI over the past 100 years. |
| Pseudocode | 1. Loop through all records. 2. Look at the year they were released. 3. Check the 3rd character in this as this will determine the decade. 4. Store this in a list that will signify that decade. 5. Display the lists in a formatted way to the TUI giving a timeline. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| ArrayList <String> the1930s | Store’s title and artist of the record in this decade. |
| ArrayList <String> the 940s | Store’s title and artist of the record in this decade. |
| ArrayList <String> the1950s | Store’s title and artist of the record in this decade. |
| ArrayList <String> the1960s | Store’s title and artist of the record in this decade. |
| ArrayList <String> the1970s | Store’s title and artist of the record in this decade. |
| ArrayList <String> the1980s | Store’s title and artist of the record in this decade. |
| ArrayList <String> the1990s | Store’s title and artist of the record in this decade. |
| ArrayList <String> the2000s | Store’s title and artist of the record in this decade. |
| ArrayList <String> the2010s | Store’s title and artist of the record in this decade. |
| ArrayList <String> the2020s | Store’s title and artist of the record in this decade. |
| Int i | Stores the year of the record |
| String j | Makes the int of i to a String |
| Char [] charSplit | Splits the year into a character index |

### Method - displayByID

|  |  |
| --- | --- |
| Purpose | Takes an ID of a record and display the records details to the TUI. |
| Pseudocode | 1. Take the ID. 2. Loop through the records. 3. Display the details of the record matching the ID. 4. Display no records found if there is no match. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| Int i | Stores the ID to a local variable |
| Boolean found | Condition for the if statement if a record is found |

### Method - displayByTitle

|  |  |
| --- | --- |
| Purpose | Takes a Title of a record and display the records details to the TUI. |
| Pseudocode | 1. Take in the title. 2. Loop through the records. 3. Display the details of the record matching the Title. 4. Display no records found if there is no match. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| String i | Stores the Title to a local variable |
| Boolean found | Condition for the if statement if a record is found |

### Method - displayByArtist

|  |  |
| --- | --- |
| Purpose | Takes an Artist of a record and display the records details to the TUI. |
| Pseudocode | 1. Take in the artist. 2. Loop through the records. 3. Display the details of the record matching the Artist. 4. Display no records found if there is no match. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| String i | Stores the Artist to a local variable |
| ArrayList <String> artist | Stores records by this artist |

### Method - displayByYear

|  |  |
| --- | --- |
| Purpose | Takes a Year of a record and display the records details to the TUI. |
| Pseudocode | 1. Take in the Year. 2. Loop through the records. 3. Display the details of the record matching the Year. 4. Display no records found if there is no match. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| int i | Stores the Year to a local variable |
| ArrayList <String> years | Stores records by this artist |

### Method - displayByDecade

|  |  |
| --- | --- |
| Purpose | Takes a Decade and displays records details from this decade to the TUI. |
| Pseudocode | 1. Take in the Decade. 2. Loop through the records. 3. Display the details of the record matching the Decade. 4. Display no records found if there is no match. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| Char [] i | Stores the decade as a character array |
| ArrayList <String> decade | Stores Title and Artist of the record matching the decade |
| Int j | Stores the year of the record |
| String k | Changes the int j into a string |
| Char [] charsplit | Creates a character array of the string k |

### Method - displayByGenre

|  |  |
| --- | --- |
| Purpose | Takes a Genre of a record and display the records details to the TUI. |
| Pseudocode | 1. Take in the Genre. 2. Loop through the records. 3. Display the details of the record matching the Genre. 4. Display no records found if there is no match. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| String i | Stores the Genre to a local variable |
| ArrayList <String> genre | Stores records by this Genre |

### Method – displayDuplicates

|  |  |
| --- | --- |
| Purpose | Display any duplicate records IDs to the TUI for the user to resolve. |
| Pseudocode | 1. Loops through the records. 2. Takes the record and compare it against all others. 3. If details match other than the ID display this to the TUI. 4. If no duplicates found display this to the TUI. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| Boolean records | Condition for the if statement if no records are found |
| ArrayList <Integer> duplicates | Stores record ID’s that are duplicates |

## Class – VinylRecord

#### Global Variables

|  |  |
| --- | --- |
| Name | Purpose |
| Private int id | Stores as global variable for lower method to access |
| Private String | Stores as global variable for lower method to access |
| Private String | Stores as global variable for lower method to access |
| Private int year | Stores as global variable for lower method to access |
| Private String | Stores as global variable for lower method to access |

### Method – VinylRecord

|  |  |
| --- | --- |
| Purpose | Constructor method to pass in the variables needed to create object of VinylRecord |
| Pseudocode | 1. Take in the variables passed to the method. 2. Store the variables as a global variable here for methods to access. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| this.id | Takes in the value and stores to global variable |
| this.title | Takes in the value and stores to global variable |
| this.artist | Takes in the value and stores to global variable |
| this.year | Takes in the value and stores to global variable |
| this.genre | Takes in the value and stores to global variable |

### Method – **get***Element*

|  |  |
| --- | --- |
| Purpose | Get methods in this class are used to look at a records specific element and then pull that element for use in another class/method. |
| Pseudocode | 1. Return the records element requested. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| NA | NA |

### Method – **set***Element*

|  |  |
| --- | --- |
| Purpose | Set methods in this class are used to change a records specific element to a new value. |
| Pseudocode | 1. Set the records element requested to new value. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| NA | NA |

### Method – toString

|  |  |
| --- | --- |
| Purpose | When the class VinyRecord is called this is the default output it sends to the TUI. |
| Pseudocode | 1. Return the records details in a formatted way for CSV. |

#### Variables

|  |  |
| --- | --- |
| Name | Purpose |
| NA | NA |

# Version History

|  |  |
| --- | --- |
| Version No. | Date |
| 1.0.1 | 29/04/2021 |
| 1.0.2 | 01/05/2021 |
| 1.0.3 | 09/05/2021 |

## Version 1.0.1

### Bugs Fixed

* Scanner not cleared after an integer take in causing some values to be blank throwing exceptions – buffer cleared on these integers take in.
* Menus not clear when taking in input from the user – Changed the TUI to indicate better and be more user friendly.
* Multiple functions being called after completion of one – Breaks in switch cases were missing in some places but they are now in where needed.

### Additional Features

* Decades Function.
* Shuffle Pick.

## Version 1.0.2

### Bugs Fixed

* Decade’s function showing all decades even if no records exist in them – If statements introduced to only display if data exists in the array list.
* Error checking not working for year input in Create function – If statement was incorrect allowing additional years, adjusted the if statement has cleared it.

### Additional Features

* Duplicates Function.

## Version 1.0.3

### Bugs Fixed

* Start method was removed to handle exceptions better and main program code was moved to the main method.
* Duplicate function not displaying in testing – If statement to check criteria against one another was incorrect. This was written with || rather than && not completing properly.

### Additional Features

* Delete in Bulk.

# Version 1.0.3 Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Details | Expected Results | Result | PASS/FAIL |
| 1 | Program boots into main menu. | Start program and the main menu is displayed. | Started the program and the main menu was displayed. | PASS |
| 2 | Program reads all of existing records in CSV into the array list. | Start program and records are read into array list. I will use the [5] Display All to confirm. | Started program and used option 5. All records where displayed. | PASS |
| 3 | Reach **Create** function from main menu. | Input the function path from the Main Menu and then function starts. | Inputted path and function started. | **PASS** |
| 4 | **Create** function with correct input. | Input details the function needs. | Inputted correct details and a new record was **created**. | **PASS** |
| 5 | **Create** function with invalid input. | Inputted invalid format. | Program terminated the **create** function and displayed message stating invalid input. | **PASS** |
| 6 | Return from **Create** to Main Menu. | Select the exit option from the submenu. | Selected exit option and returned to Main Menu. | **PASS** |
| 7 | Reach **Edit** function from main menu. | Input the function path from the Main Menu and then function starts. | Inputted path and function started. | **PASS** |
| 8 | **Edit** function with correct input. | Input details the function needs. | Inputted correct details and a record was **edited**. | **PASS** |
| 9 | **Edit** function with invalid input. | Inputted invalid format. | Program terminated the **edit** function and displayed message stating invalid input. | **PASS** |
| 10 | Return from **Edit** to Main Menu. | After completion of function program returns to main menu. | Selected exit option and returned to Main Menu. | **PASS** |
| 11 | Return from **Edit** to Main Menu. | Select the exit option from the submenu. | After completion of function program was returned to main menu | **PASS** |
| 12 | Reach **Delete** function from main menu. | Input the function path from the Main Menu and then function starts. | Inputted path and function started. | **PASS** |
| 13 | **Delete** function with correct input. | Input details the function needs. | Inputted correct details and record was **deleted**. | **PASS** |
| 14 | **Delete** function with invalid input. | Inputted invalid format. | Program displayed message stating invalid input and prompted to enter again. | **PASS** |
| 15 | Return from **Delete** to Main Menu. | After completion of function program returns to main menu. | Selected exit option and returned to Main Menu. | **PASS** |
| 16 | Return from **Delete** to Main Menu. | Select the exit option from the submenu. | After completion of function program was returned to main menu | **PASS** |
| 17 | Reach **Delete in Bulk** function from main menu. | Input the function path from the Main Menu and then function starts. | Inputted path and function started. | **PASS** |
| 18 | **Delete in Bulk** function with correct input. | Input details the function needs. | Inputted correct details and record was **deleted in Bulk**. | **PASS** |
| 19 | **Delete in Bulk** function with invalid input. | Inputted invalid format. | Program terminated the **delete in Bulk** function and displayed message stating invalid input. | **PASS** |
| 20 | Return from **Delete in Bulk** to Main Menu. | After completion of function program returns to main menu. | Selected exit option and returned to Main Menu. | **PASS** |
| 21 | Return from **Delete in Bulk** to Main Menu. | Select the exit option from the submenu. | After completion of function program was returned to main menu | **PASS** |
| 22 | Reach **Search by ID** function from main menu. | Input the function path from the Main Menu and then function starts. | Inputted path and function started. | **PASS** |
| 23 | **Search by ID** function with correct input. | Input details the function needs. | Inputted correct details and record was displayed. | **PASS** |
| 24 | **Search by ID** function with invalid input. | Inputted invalid format. | Program displayed message stating invalid input and prompted to enter again. | **PASS** |
| 25 | Return from **Search by ID** to Main Menu. | After completion of function program returns to main menu | Selected exit option and returned to Main Menu. | **PASS** |
| 26 | Return from **Search by ID** to Main Menu. | Select the exit option from the submenu. | After completion of function program was returned to main menu | **PASS** |
| 27 | Reach **Search by Title** function from main menu. | Input the function path from the Main Menu and then function starts. | Inputted path and function started. | **PASS** |
| 28 | **Search by Title** function with correct input. | Input details the function needs. | Inputted correct details and record was displayed. | **PASS** |
| 29 | Return from **Search by Title** to Main Menu. | After completion of function program returns to main menu | Selected exit option and returned to Main Menu. | **PASS** |
| 30 | Return from **Search by Title** to Main Menu. | Select the exit option from the submenu. | After completion of function program was returned to main menu | **PASS** |
| 31 | Reach **Search by Artist** function from main menu. | Input the function path from the Main Menu and then function starts. | Inputted path and function started. | **PASS** |
| 32 | **Search by Artist** function with correct input. | Input details the function needs. | Inputted correct details and record was displayed. | **PASS** |
| 33 | Return from **Search by Artist** to Main Menu. | After completion of function program returns to main menu | Selected exit option and returned to Main Menu. | **PASS** |
| 34 | Return from **Search by Artist** to Main Menu. | Select the exit option from the submenu. | After completion of function program was returned to main menu | **PASS** |
| 35 | Reach **Search by Year** function from main menu. | Input the function path from the Main Menu and then function starts. | Inputted path and function started. | **PASS** |
| 36 | **Search by Year** function with correct input. | Input details the function needs. | Inputted correct details and records displayed. | **PASS** |
| 37 | **Search by Year** function with invalid input. | Inputted invalid format. | Program displayed message stating invalid input and prompted to enter again. | **PASS** |
| 38 | Return from **Search by Year** to Main Menu. | Select the exit option from the submenu. | Selected exit option and returned to Main Menu. | **PASS** |
| 39 | Return from **Search by Year** to Main Menu. | Select the exit option from the submenu. | After completion of function program was returned to main menu | **PASS** |
| 40 | Reach **Search by Decade** function from main menu. | Input the function path from the Main Menu and then function starts. | Inputted path and function started. | **PASS** |
| 41 | **Search by Decade** function with correct input. | Input details the function needs. | Inputted correct details and records displayed. | **PASS** |
| 42 | Return from **Search by Decade** to Main Menu. | Select the exit option from the submenu. | Selected exit option and returned to Main Menu. | **PASS** |
| 43 | Reach **Search by Genre** function from main menu. | Input the function path from the Main Menu and then function starts. | Inputted path and function started. | **PASS** |
| 44 | **Search by Genre** function with correct input. | Input details the function needs. | Inputted correct details and records displayed. | **PASS** |
| 45 | Return from **Search by Genre** to Main Menu. | After completion of function program returns to main menu | Selected exit option and returned to Main Menu. | **PASS** |
| 46 | Return from **Search by Genre** to Main Menu. | Select the exit option from the submenu. | After completion of function program was returned to main menu | **PASS** |
| 47 | Reach **Duplicate** function from main menu. | Input the function path from the Main Menu and then function starts. | Inputted path and function started. | **PASS** |
| 48 | Duplicate function works correctly. | Display duplicates if they exist. | Duplicates shown to the TUI. | **PASS** |
| 49 | Return from **Duplicate** to Main Menu. | After completion of function program returns to main menu | Selected exit option and returned to Main Menu. | **PASS** |
| 50 | Display All function works correctly. | All records are shown to the TUI. | All records were displayed to the TUI. | **PASS** |
| 51 | Return from **Display All** to Main Menu. | After completion of function program returns to main menu | Selected exit option and returned to Main Menu. | **PASS** |
| 52 | Decade’s function works correctly. | A timeline displayed with the records in there right decade. | Timeline was shown to the TUI with the records in decade. | **PASS** |
| 53 | Return from **Decades** to Main Menu. | After completion of function program returns to main menu | Selected exit option and returned to Main Menu. | **PASS** |
| 54 | Shuffle Pick function works correctly. | A randomly selected record is shown to the TUI. | Records was randomly selected and displayed to the TUI. | **PASS** |
| 55 | Return from **Shuffle Pick** to Main Menu. | After completion of function program returns to main menu | Selected exit option and returned to Main Menu. | **PASS** |
| 56 | Exit function works correctly. | The program to write the records to the file, display where the file can be reached and close the program. | Records where written, the file location was displayed, and program closed. | **PASS** |