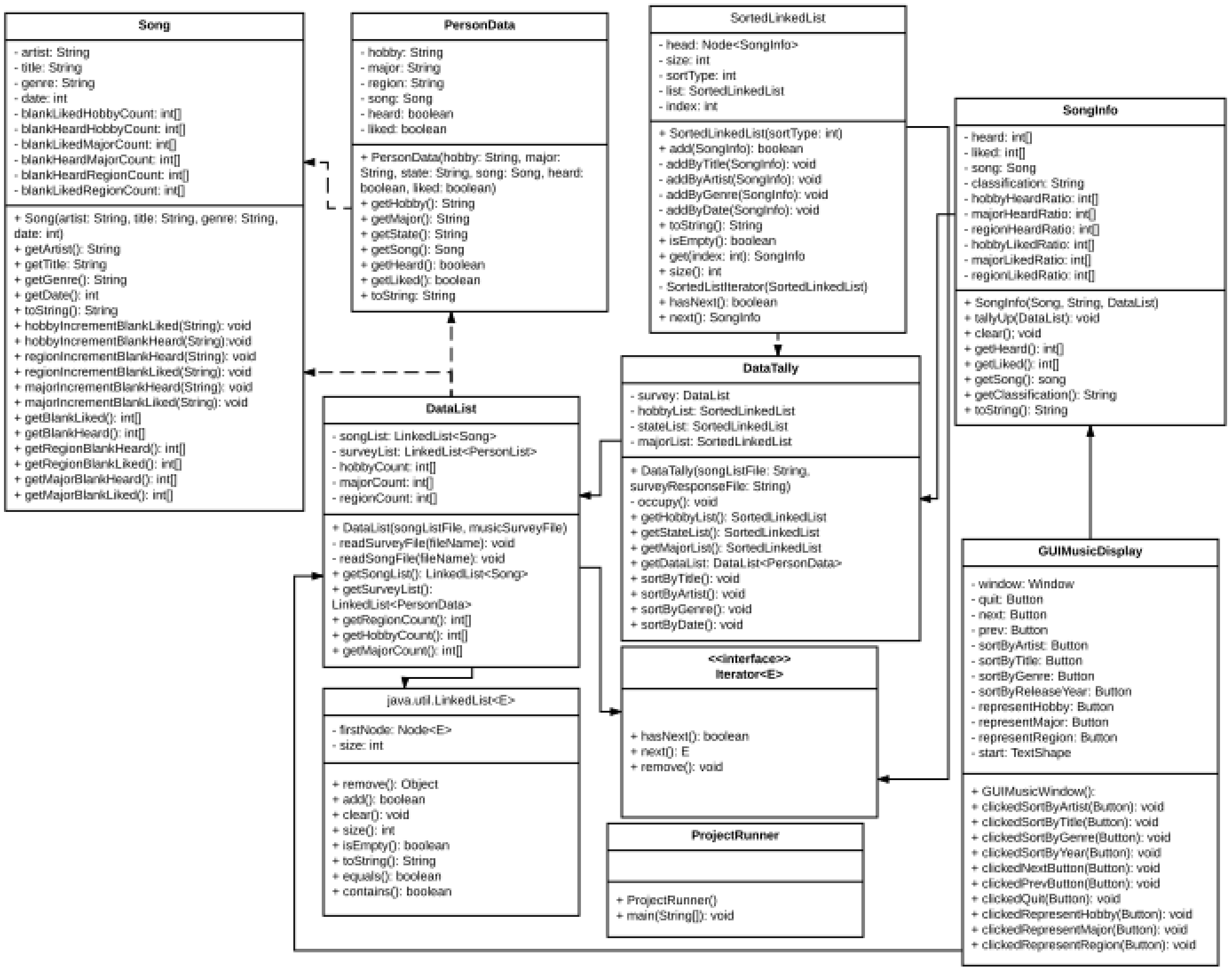
Group 35
Group Members:
Jack Dong – jack3664
Kevin Jiang – kjiang
Tam Phan – tphan25

<u>Updated UML</u>:


Song
- artist: String
- title: String
- genre: String
- date: int
- blankLikedHobbyCount: int[]
- blankHeardHobbyCount: int[]
- blankLikedMajorCount: int[]
- blankHeardMajorCount: int[]
- blankHeardRegionCount: int[]
- blankLikedRegionCount: int[]
+ Song(artist: String, title: String, genre: String, date: int)
+ getArtist(): String
+ getTitle: String
+ getGenre(): String
+ getDate(): int
+ toString(): String
+ hobbyIncrementBlankLiked(String): void
+ hobbyIncrementBlankHeard(String):void
+ regionIncrementBlankHeard(String): void
+ regionIncrementBlankLiked(String): void
+ majorIncrementBlankHeard(String): void
+ majorIncrementBlankLiked(String): void
+ getBlankLiked(): int[]
+ getBlankHeard(): int[]
+ getRegionBlankHeard(): int[]
+ getRegionBlankLiked(): int[]
+ getMajorBlankHeard(): int[]
+ getMajorBlankLiked(): int[]
PersonData
- hobby: String
- major: String
- region: String
- song: Song
- heard: boolean
- liked: boolean
+ PersonData(hobby: String, major: String, state: String, song: Song, heard: boolean, liked: boolean)
+ getHobby(): String
+ getMajor(): String
+ getState(): String
+ getSong(): Song
+ getHeard(): boolean
+ getLiked(): boolean
+ toString: String
SortedLinkedList
- head: Node<SongInfo>
- size: int
- sortType: int
- list: SortedLinkedList
- index: int
+ SortedLinkedList(sortType: int)
+ add(SongInfo): boolean
- addByTitle(SongInfo): void
- addByArtist(SongInfo): void
- addByGenre(SongInfo): void
- addByDate(SongInfo): void
+ toString(): String
+ isEmpty(): boolean
+ get(index: int): SongInfo
+ size(): int
- SortedListIterator(SortedLinkedList)
+ hasNext(): boolean
+ next(): SongInfo
SongInfo
- heard: int[]
- liked: int[]
- song: Song
- classification: String
- hobbyHeardRatio: int[]
- majorHeardRatio: int[]
- regionHeardRatio: int[]
- hobbyLikedRatio: int[]
- majorLikedRatio: int[]
- regionLikedRatio: int[]
+ SongInfo(Song, String, DataList)
+ tallyUp(DataList): void
+ clear(); void
+ getHeard(): int[]
+ getLiked(): int[]
+ getSong(): song
+ getClassification(): String
+ toString(): String
DataList
- songList: LinkedList<Song>
- surveyList: LinkedList<PersonList>
- hobbyCount: int[]
- majorCount: int[]
- regionCount: int[]
+ DataList(songListFile, musicSurveyFile)
- readSurveyFile(fileName): void
- readSongFile(fileName): void
+ getSongList(): LinkedList<Song>
+ getSurveyList(): LinkedList<PersonData>
+ getRegionCount(): int[]
+ getHobbyCount(): int[]
+ getMajorCount(): int[]
DataTally
- survey: DataList
- hobbyList: SortedLinkedList
- stateList: SortedLinkedList
- majorList: SortedLinkedList
+ DataTally(songListFile: String, surveyResponseFile: String)
- occupy(): void
+ getHobbyList(): SortedLinkedList
+ getStateList(): SortedLinkedList
+ getMajorList(): SortedLinkedList
+ getDataList: DataList<PersonData>
+ sortByTitle(): void
+ sortByArtist(): void
+ sortByGenre(): void
+ sortByDate(): void
java.util.LinkedList<E>
- firstNode: Node<E>
- size: int
+ remove(): Object
+ add(): boolean
+ clear(): void
+ size(): int
+ isEmpty(): boolean
+ toString(): String
+ equals(): boolean
+ contains(): boolean
<<interface>>
Iterator<E>
+ hasNext(): boolean
+ next(): E
+ remove(): void
ProjectRunner
+ ProjectRunner()
+ main(String[]): void
GUIMusicDisplay
- window: Window
- quit: Button
- next: Button
- prev: Button
- sortByArtist: Button
- sortByTitle: Button
- sortByGenre: Button
- sortByReleaseYear: Button
- representHobby: Button
- representMajor: Button
- representRegion: Button
- start: TextShape
+ GUIMusicWindow():
+ clickedSortByArtist(Button): void
+ clickedSortByTitle(Button): void
+ clickedSortByGenre(Button): void
+ clickedSortByYear(Button): void
+ clickedNextButton(Button): void
+ clickedPrevButton(Button): void
+ clickedQuit(Button): void
+ clickedRepresentHobby(Button): void
+ clickedRepresentMajor(Button): void
+ clickedRepresentRegion(Button): void

Group 35
Group Members:
Jack Dong – jack3664
Kevin Jiang – kjiang
Tam Phan – tphan25

This UML shows the updated one that we created based on the old one from the intermediate submission. We changed some methods in classes such as DataList. DataList will still read the files using readSongFile() and readSurveyFile() and keep a singly linked list of Songs and PersonData. The class also keeps count of total number of people with specific hobbies, state, and major in order to obtain a percentage when used together with the SongInfo/DataTally class. PersonData stores information about a students response to a specific song and whether or not he/she has heard/liked it. The Song class stores information about each song including the artist, title, date, and genre. SongInfo class stores the number of heard/liked in arrays associated with each song based on each classification such as hobby, major, and region. The tallyUp() method in the SongInfo class is what tallies up the results using information from the DataList class. Ww also created a new SortedLinkedList class in order to hold all the songs in order based on what was required: by title, artist, date, and genre. The increment methods in the song class is called whenever heard or liked is empty when the files are read in the DataList class. The GUIMusicDisplay and ProjectRunner classes are self explanatory.