**Table of Contents**

PR2 Project Design

A. Class Diagram 3

B. Event Trace Diagram 4

Scenario 1: Startup 4

Scenario 2: Shutdown 4

Scenario 3: Error Handling-Close of Application (Lift) 5

Scenario 4: Error Handling-Close of Application (WOD) 5

Scenario 5: Error Handling-Close of Application (Notes) 6

Scenario 6: Error Handling-Close of Application (1RM) 6

Scenario 7: Error Handling-Close of Application (Time) 7

Scenario 8: Load Lift 7

Scenario 9: Load Lift and Change 1RM 8

Scenario 10: Load WOD 8

Scenario 11: Load WOD and Change Time 9

Scenario 12: Load WOD and Change Note 9

Scenario 13: Full use of application 10

C. Pseudocode 11

1. Main Subsystem 11

2. Lift Subsystem 11

3. WOD Subsystem 12

4. Data File Subsystem 12

5. List of Classes Utilized by the Subsystems 13

5a. WOD.class 13

5b. Lift.class 13

5c. fromTextFile Class 13

D. Unresolved Risks 15

Table of Figures

Figure 1: Class Diagram 3

Figure 2: Scenario 1 Event Trace Diagram 4

Figure 3: Scenario 2 Event Trace Diagram 4

Figure 4: Scenario 3 Event Trace Diagram 5

Figure 5: Scenario 4 Event Trace Diagram 5

Figure 6: Scenario 5 Event Trace Diagram 6

Figure 7: Scenario 6 Event Trace Diagram 6

Figure 8: Scenario 7 Event Trace Diagram 7

Figure 9: Scenario 8 Event Trace Diagram 7

Figure 10: Scenario 9 Event Trace Diagram 8

Figure 11: Scenario 10 Event Trace Diagram 8

Figure 12: Scenario 11 Event Trace Diagram 9

Figure 13: Scenario 12 Event Trace Diagram 9

Figure 14: Scenario 13 Event Trace Diagram 10

***PR2 Project Design***

A. Class Diagram:

|  |
| --- |
| **Main** |
| WOD:arraylist |
| Lifts:arraylist |
| selectedWOD:WOD |
| selectedLift:Lift |
| WodSys() |
| LiftSys() |
| toBackUp() |
| toTextFile() |
| fromTextFile() |

|  |
| --- |
| **fromTextFile** |
| liftData:ArrayList |
| wodData:ArrayList |
| toTextFile() |
| toBackupFile() |

|  |
| --- |
| **WOD** |
| wodName:String |
| hour:String |
| min:String |
| sec:String |
| content:String |
| notes:String |
| url:String |
| WOD() |
| WOD(String wodName, String  hour, String min, String sec) |
| getDescription(wodName) |
| getNotes(wodName) |
| setNotes(wodName) |
| getUrl(wodName) |
| getHour(wodName) |
| setHour(wodName) |
| getMin(wodName) |
| setMin(wodName) |
| getSec(wodName) |
| setSec(wodName) |
| newTime(timeInput) |
| newNotes(noteInput) |
| toString() |

|  |
| --- |
| **Lift** |
| liftName:String |
| oneRepMax:String |
| Lift() |
| Lift(liftName, 1RM) |
| Get1RM(liftName) |
| set1RM(liftName) |
| liftToString() |

|  |
| --- |
| **LiftSys(selectedLift)** |
| liftName:String |
| oneRepMax:String |
| Increments(oneRepMax) |
| changeIneRM(chgInput) |

|  |
| --- |
| **WodSys(selectedWOD)** |
| wodName:String |
| hour:String |
| min:String |
| sec:String |
| content:String |
| notes:String |
| url:String |
| newNotes() |
| newTime() |



Figure 1: Class Diagram

B. Event Trace Diagrams

Scenario 1: Startup

Description: **User** opens application.

Precondition: Computer is on and updated.

Post-condition: **PR2** is up and working.

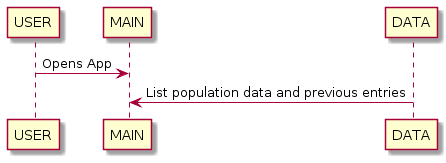


Figure 2: Scenario 1

Scenario 2: Shutdown

Description: User closes **PR2** without error, permitting data to be stored for persistence.

Precondition: **PR2** is up and running.

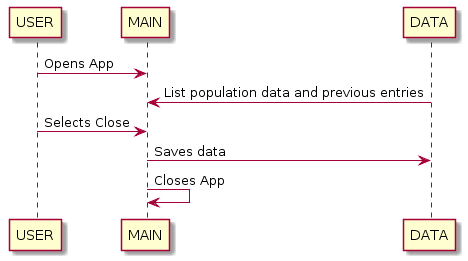
Post-condition: **PR2** is closed and data persistence is maintained.

Figure 3: Scenario 2

Scenario 3: Error Handling-Close of Application (Lift)

Description: User attempts to close **PR2** prior to data being stored properly.

Precondition: **PR2** is up and running, new **1RM** data has been entered

Post-condition: Exit is blocked until new data is stored properly.

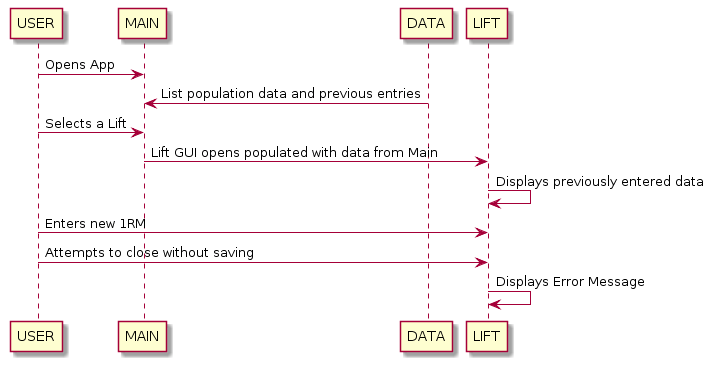


Figure 4: Scenario 3

Scenario 4: Error Handling-Close of Application (WOD)

Description: User attempts to close **PR2** prior to data being stored properly.

Precondition: **PR2** is up and running, new **Time** data has been entered.

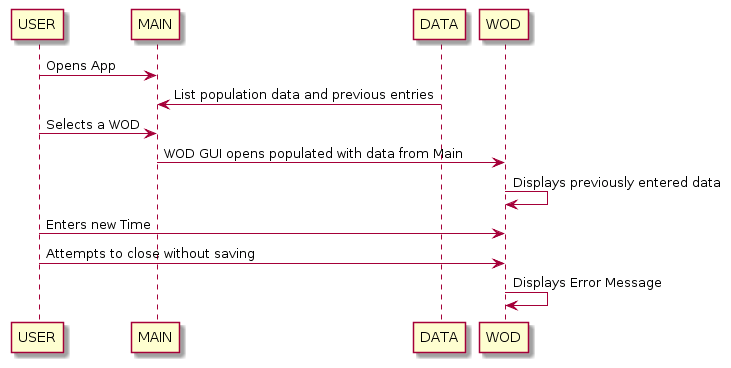
Post-condition: Exit is blocked until new data is stored properly.

Figure 5: Scenario 4

Scenario 5: Error Handling-Close of Application (Notes)

Description: User attempts to close **PR2** prior to data being stored properly.

Precondition: **PR2** is up and running, new **Note** data has been entered.

Post-condition: Exit is blocked until new data is stored properly.

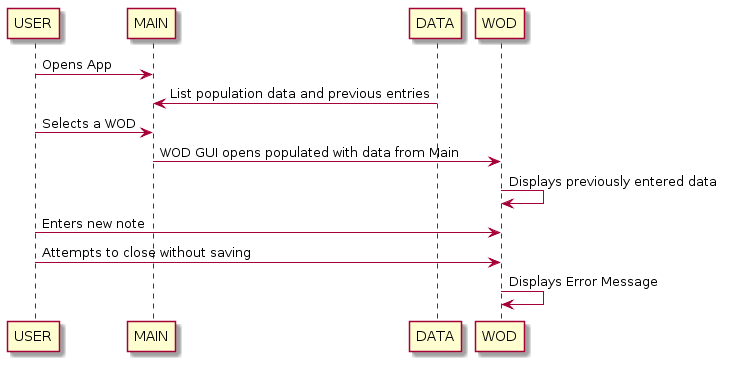


Figure 6: Scenario 5

Scenario 6: Error Handling-Improper **1RM** entry

Description: User inputs **1RM** improperly.

Precondition: **PR2** is up and running, **User** has clicked “enter new **1RM**” button.

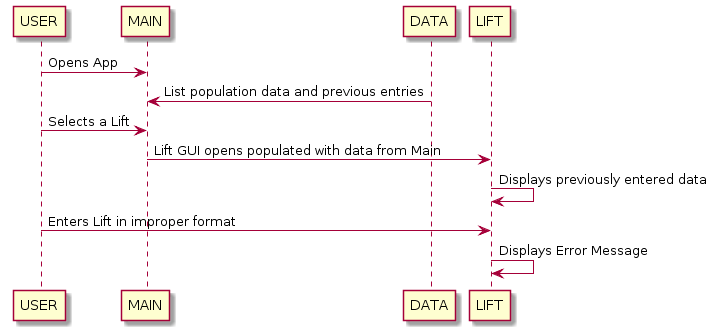
Post-condition: Entry is not accepted, **User** is prompted to reenter data correctly.

Figure 7: Scenario 6

Scenario 7: Error Handling-Improper **Time** Entry

Description: **User** inputs **Time** improperly.

Precondition: **PR2** is up and running, **User** has clicked “enter new **Time**” button.

Post-condition: Entry is not accepted, **User** is prompted to reenter data correctly.

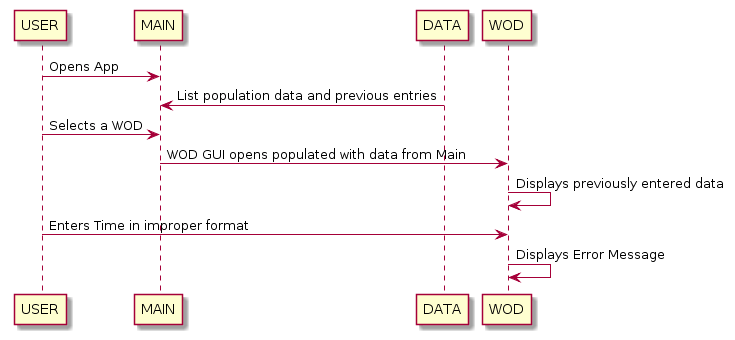


Figure 8: Scenario 7

Scenario 8: Load **Lift**

Description: User selects specific **Lift.**

Precondition: **PR2** is up and running.

Post-Condition: **1RM** and percentage decrements of specific **Lift** are displayed.

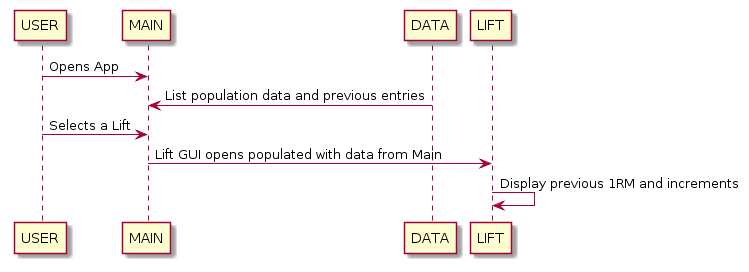


Figure 9: Scenario 8

Scenario 9: Load **Lift** and Change **1RM**

Description: User enters new **1RM** properly.

Precondition: User selects specific **Lift**, then clicks “enter new **1RM**” button.

Post-Condition: new **1RM** is displayed with decremented percentages.

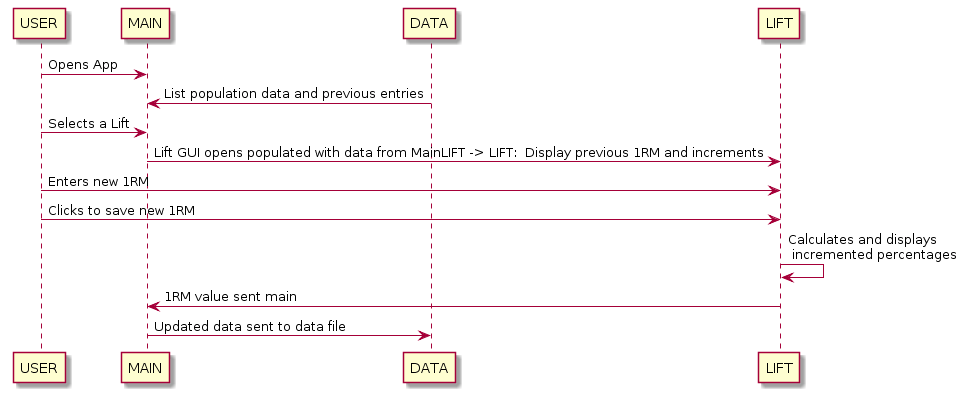


Figure 10: Scenario 9

Scenario 10: Load **WOD**

Description: User selects specific **WOD.**

Precondition: **PR2** is up and running.

Post-Condition: **Time** specific **WOD** is displayed.

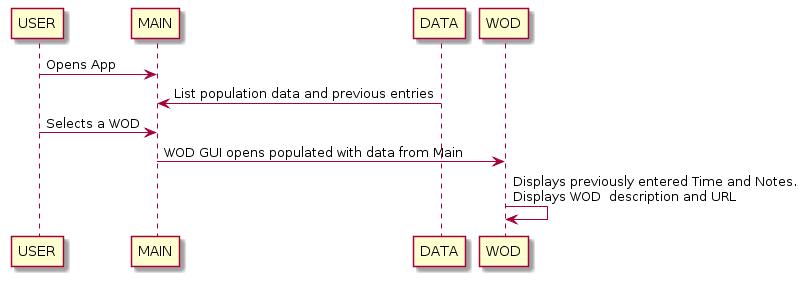


Figure 11: Scenario 10

Scenario 11: Load **WOD** and Change **Time**

Description: User enters new **Time** properly.

Precondition: User selects specific **Time**, then clicks “enter new **Time**” button.

Post-Condition: new **Time** is displayed.

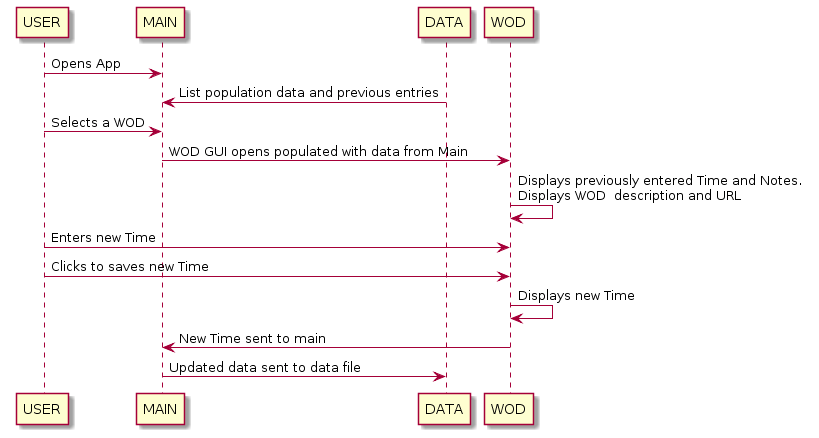


Figure 12: Scenario 11

Scenario 12: Load **WOD** and Change **Note**

Description: User enters new **Note**

Precondition: User selects specific **Time**, then clicks “enter new **Note**” button.

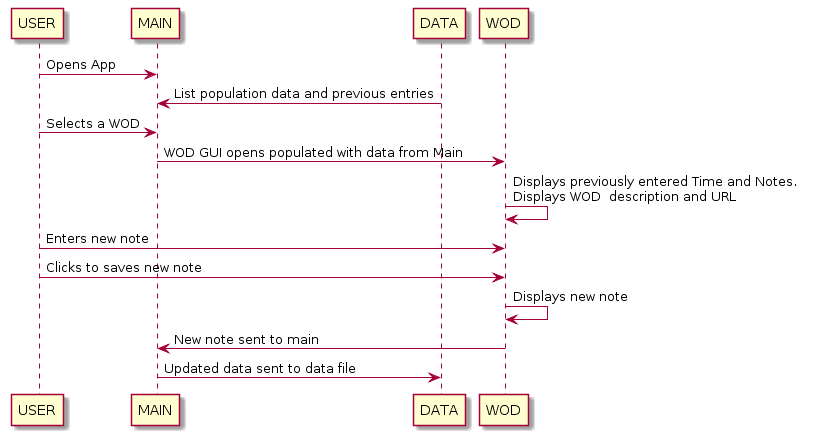
Post-Condition: new **Note** is displayed.

Figure 13: Scenario 12

Scenario 13: Full use of Application

Description: User enters new **1RM**, **WOD** and **Note**

Precondition: **PR2** is up and running.

Post-condition: **PR2** is closed and data persistence is maintained.

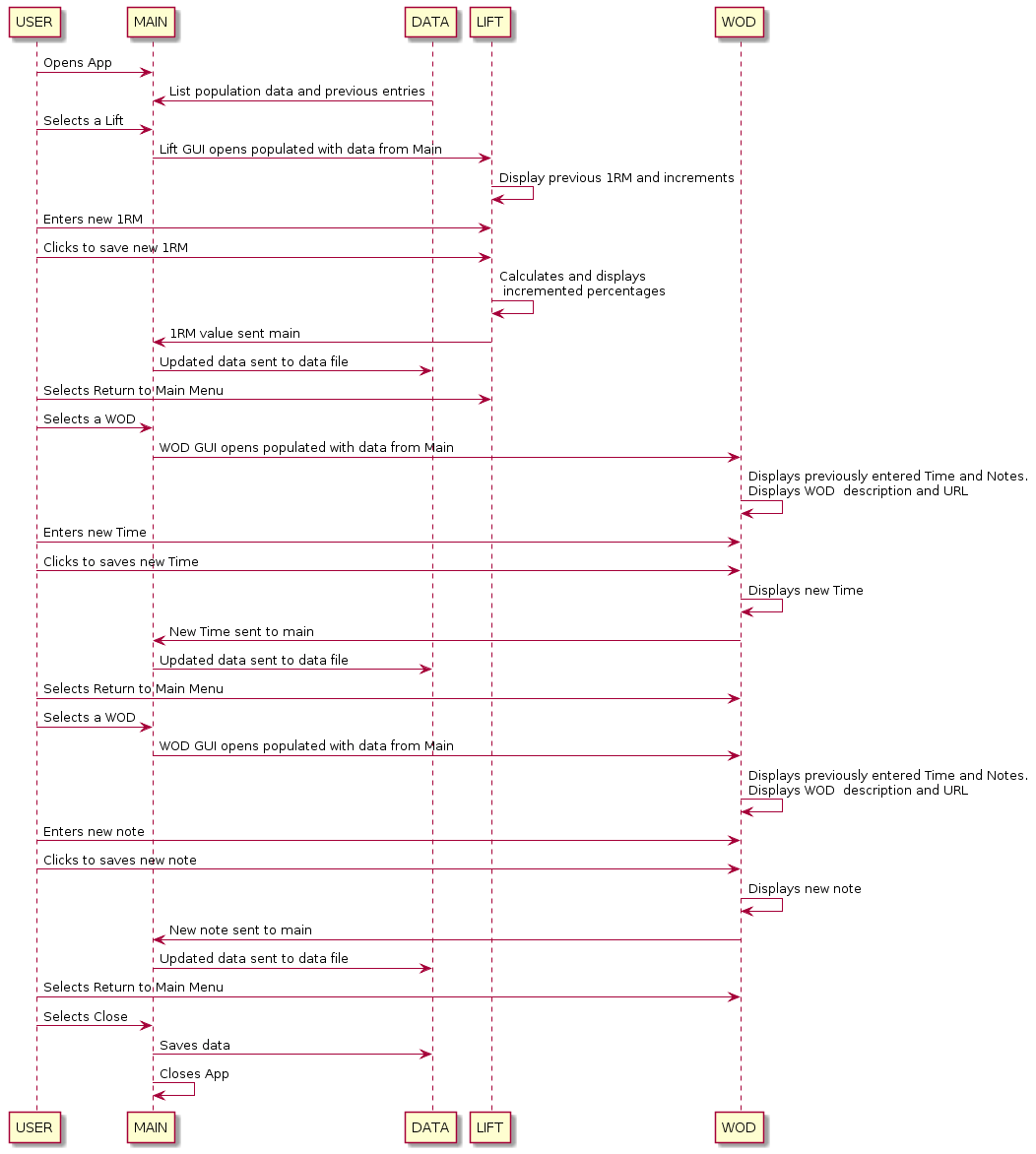


Figure 14: Scenario 13

C. Pseudocode:

1. Main Subsystem

Main class{

creates a new FromText.class object

public main(String input){

1. Reads contents of **Data File subsystem** into **FromText**.class Object, if error is encountered reads contents of backup.txt

2. Create a GUI displaying **Lift** dropbox and a **WOD** dropbox and a close button.

4. Populates **Lift** dropbox using lift names from the **lift** arrayList.

5. Populates **WOD** dropbox using **WOD** names from the **WOD** ArrayList.

6. Passes Lift dropbox **User** selection to **Lift Subsystem.**

7. Passes **WOD** dropbox **User** selection to **WOD Subsystem**.

}

}

1. Lift Subsystem

LiftSys Class{

1. accepts the **User** selection from the **Lift** dropbox in the **Main Subsystem**
2. Using the input, the LiftSys Class retrieves the **Lift** name and 1RM from the **Main**

**Subsystem**

1. Closes GUI and returns to Main Menu when “Return to Main Menu” button is pressed.
2. Verifies 1RM to be type float, prompts user to correct input if not.

String input;

String name;

String oneRepMax;

String increment;

public LiftSys(String input){

1. Generates a **GUI** that displays the **Lift** name, **Lift** oneRepMax, increments, “Return to Main Menu” button, input box for new 1RM and “Change 1RM” button.

}

public String increments(String oneRepMax){

1. Accepts the oneRepMax, converts it to a float, generates the percentage increments, and returns them in String format.

}

public String changeOneRM(String chgInput){

1. Accepts input from “Change 1RM” passes to increments(), changes oneRepMax and displays the updated oneRepMax and increment.

} }

1. WOD Subsystem

WodSys class{

1. Accepts the User selection from the **WOD** dropbox in the **Main Subsystem**
2. Using the input, the WodSys Class retrieves the **WOD** name, hour, min, sec, contents, notes and url from the **Main Subsystem.**
3. Closes GUI and returns to Main Menu when “Return to Main Menu” button is pressed.

String input;

public WodSys(String input){

1. Generates a **GUI** that displays the **WOD** name, hour, min, sec, contents, notes, url, “Return to Main Menu” button, “Enter New Time” button, and “Update Notes” button. The input boxes for the Notes and time will be editable.

}

public String newTime(String timeInput){

1. When “Enter New Time” button is pressed, user input is accepts, error message is thrown if incorrect, and new time is displayed in updated window.

}

public String newNotes(String noteInput){

1. When “Update Notes” button is pressed user input from notes field is accepted and the notes field is updated and then displayed.

}

1. Data File Subsystem

Datafile is a text file with the layout shown in the following example

// : Lift format

// : l :<name>:<1 rep max>

// : name 1 rep max

l : Back squat : 100

// : WOD format

// : w :<name>:<hour>:<minute>:<second><Content><Notes><URL>

// : name hour minute second

w : Abbate : 0 : 0 : 0 : Run 1 mile-155 pound Clean and jerk : Add Any Notes here : www.cf.com

1. List of classes utilized by the Subsystems.

5a. WOD Class is an object class to hold pertinent data for the **WOD Subsystem**

Class WOD{

String wodName;

String hour;

String min:

String sec;

String content;

String notes;

String url;

public WOD(String wodName, String hour, String min, String sec){

constructor}

appropriate Accessors and Mutators

public toString(){

displays WOD object}

}

5b. Lift class is an object class to hold data pertinent to the **Lift Subsystem**

Class Lift{

String liftName;

String 1RepMax;

public Lift(String liftName, String 1RepMax){

constructor}

appropriate Accessors and Mutators

public toString(){

displays Lift object}

}

5c. The FromText Class is a class utilized by the **Main Subsystem** for storing the data retrieved from the **Data File Subsystem**

Class FromText {

ArrayList<Lift> liftData = new ArrayList<Lift>(); //to hold the Lifts

ArrayList<WOD> wodData = new ArrayList<WOD>(); //to hold the WODs

boolean FromText(String newData) throws IllegalFormat{

Parse the input String to see if data is Lift or WOD

if(Lift){

Parse input string into Lift object and store in liftData ArrayList}

else if(WOD){

Parse input string into WOD object and store in wodData ArrayList}

else{

Throw illegalFormat}

}

Return true;

toTextFile(){

1. Writes contents of **Lift** arrayList and **WOD** arrayList into **Data File subsystem** to maintain persistence.

}

toBackupFile(){

1. Writes contents of **Lift** arrayList and **WOD** ArrayList into “backup.txt”}

Reads contents of **Data File Subsystem** into backup.txt file for future error handling purposes.

//Accessor and mutators of **WOD** objects in wodData ArrayList

**WOD** **Name** will be retrieved from Jcombobox in Main will not need to be accessed or mutated

getDescription(String wodName){

1. Accepts string input to search wodData ArrayList for **Description** of passed wod name
2. Returns said **Description.**}

Description will not be changed by user No mutator

getNotes(String wodName){

1. Accepts string input to search wodData ArrayList for **Notes** of passed **wod** name
2. Returns said **Notes.**}

setNotes(String wodName, String inputNotes){

1. Accepts string input to search wodData ArrayList for specific **wod** object.
2. Sets passed inputNotes as **Notes** data**.**}

getUrl(String wodName){

1. Accepts string input to search wodData ArrayList for **url** of passed **wod** name
2. Returns said **url.**}

URL will not be changed by user, No Mutator

getHour(String wodName){

1. Accepts string input to search wodData ArrayList for **Hour** of passed **wod** name
2. Returns said **Hour.**}

setHour(String wodName, String inputHour){

1. Accepts string input to search wodData ArrayList for specific **wod** object.
2. Sets passed inputHour as **Hour** data.

getMin(String wodName){

1. Accepts string input to search wodData ArrayList for **Minute** of passed **wod** name
2. Returns said **Minute.**}

setMin(String wodName, String inputMin){

1. Accepts string input to search wodData ArrayList for specific **wod** object.
2. Sets passed inputMin as **Minute** data**.**}

getSec(String wodName){

1. Accepts string input to search wodData ArrayList for **Second** of passed **wod** name
2. Returns said **Second.**}

setSec(String wodName, String inputSec){

1. Accepts string input to search wodData ArrayList for specific **wod** object.
2. Sets passed inputSec as **Second** data.}

//Accessors and mutators of **Lift** objects in liftData ArrayList

**Lift Name** will be retrieved from Jcombobox in Main will not need to be accessed or mutated

Get1RM(String liftName){

1. Accepts string input to search liftData ArrayList for **1RM** of passed **LIFT** name
2. Returns said **1RM.**}

Set1RM(String liftName, String input1RM){

1. Accepts string input to search liftData ArrayList for specific **Lift** object.
2. Sets passed input1RM as **1RM** data.}

}

D. Unresolved Risks: All anticipated risks have been identified in the preliminary analysis and mitigated in accordance with Risk Mitigation Table, as shown in the Project Analysis, Revision 1.14. Unanticipated risks will be addressed as they are identified.