|  |
| --- |
| **8/9/2022** |

|  |
| --- |
| **Technical Report Template** |
| ICTPRG430 – Apply Introductory Object-Oriented Language Skills AT2 |
|  |
|  |
|  |
| **Shatha Almaktoum** |
|  |
|  |

Table of Contents

[Part 1 – Planning Documentation 2](#_Toc96108752)

[Task 1 – Email 2](#_Toc96108753)

[Task 2 – Report 2](#_Toc96108754)

[Proposed data structures 2](#_Toc96108755)

[Object Oriented Language details 2](#_Toc96108756)

[Object Oriented Techniques 2](#_Toc96108757)

[Internal and External Documentation plan 2](#_Toc96108758)

[Proposed Development stages and description 3](#_Toc96108759)

[Part 2 – Application Development 4](#_Toc96108760)

[Task 3 – Application Development 4](#_Toc96108761)

[Selection, Iteration and Sequence Constructs: 4](#_Toc96108762)

[Read/Write: 4](#_Toc96108763)

[Classes 5](#_Toc96108764)

[Data Types, Operators, Expressions 5](#_Toc96108765)

[Task 4 – Development Requirements 6](#_Toc96108766)

[Debugging: 6](#_Toc96108767)

[Organisational Standards 6](#_Toc96108768)

[Functionality testing with outcomes (#1 example provided - additional 5 tests required): 7](#_Toc96108769)

[Part 3 – Hand-Over 8](#_Toc96108770)

[Task 5 – Conclusion 8](#_Toc96108771)

# Part 1 – Planning Documentation

## Task 1 – Email

* An email to your client summarising your understanding of the application to be developed including:
  + A request for feedback on your understanding of the requirements
  + A checklist of required features of the application

**EMAIL TO CLIENT**

|  |  |
| --- | --- |
| **To:** | mark@uptownbiz.com |
| **From:** | [shatha@downtown.com](mailto:shatha@downtown.com) |
| **Subject:** | Requirements and Features for Gold Coast E-Sports Desktop Application |
| **Message:** | Hi Mark,  Thank you for choosing Downtown IT for your Gold Coast E-Sports application. I’m seeking your feedback on my understanding of the requirements, please confirm if the following checklist meets your business requirements:   1. Compatibility with Windows 10 2. Ability to add new entries in the following 3. New competition result i.e. date, location, game, team, points earned 4. New team i.e. team name, contact person, phone, email, player names 5. Ability to Update existing team data 6. Change contact person name, phone, email 7. Add or remove any player within team 8. Ability to save data to desk when application is closed 9. Ability to read previously entered data when application is opened 10. Ability to display the retrieved data in a list or table   Once confirmed, we will be able to begin the development, please let me know if you have any questions or if the list does not meet your requirements  Regards,  Shatha Almaktoum |

## Task 2 – Report

The developed report can make use of the headings below and is designed to demonstrate your knowledge of the language and tools to be used in the application development process.

### Proposed data structures

Competition class

Teams class

Player class

Array Lists

IOException

### Object Oriented Language details

1. **Encapsulation**: private data accessed by public methods

public class Player{

private String playerName;

private String team;

public Player (String playerName, String team)

{

this.playerName = playerName;

this.team = team;

}

1. **Polymorphism**: allows the child class to share information and behaviour of parent @Overide allows it to incorporate its own functionality.
2. Abstraction
3. Inheritance

### Object Oriented Techniques

*Inheritance*- eliminates redundant code, subclass inherits from superclass e.g. GC\_Esports\_GUI **extends** javax.swing.JFrame

*Polymorphism*- child class overrides (@Override) the parent class methods

*Encapsulation*- private data fields and public methods (getters and setters) to access

*Abstraction*- from the graphical user interface, features are abstracted and user is not required any knowledge of the background implementations

### Internal and External Documentation plan

Internal documentation will follow standard comments format, if internal note is very long it is wrapped in /\* text \*/ if it is completed in a single line the // will suffice

Version control will be implemented at each working stage to prevent any loss of code or logical/syntactical mistakes.

External documentation that will be implemented are following the Agile methodology which aims to document not more than functionally necessary during the development but to keep it simple due to the evolution of the code through each iteration, the external documentation will be produced predominantly towards the tail end of the project.

### Proposed Development stages and description

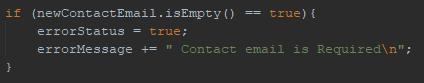
Agile development stages: Sprint planning process

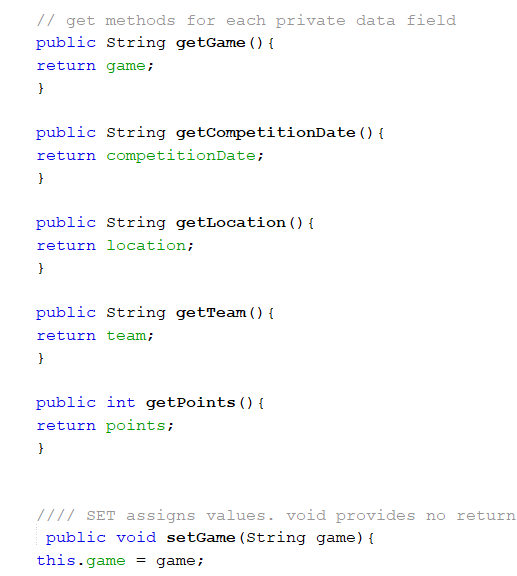
1. **Requirements gathering/ meeting stage**: At this stage understanding the business requirements and establishing the minimum viable product (MVP). GC E Sports MVP is utilising CSV files and the ability to complete basic actions such as updating existing teams, add teams, add competition results and viewing competition results
2. **Create user stories:** Can be broken down into two types of users, New or existing
3. **Add detail to the user stories:** 
   1. New user: Uses “Add New Team” tab to add their team name, contact details and all the players in the team. They can instantly progress to the functionality of ‘existing user’
   2. Existing user: Has various functionalities such as Add competition results, they can update their team members or contact details and they can view Results
4. **Break down user stories into specific tasks**: example tab 3- Add new competition Result. The team Combo Box needs to be paired with the CSV file. On Action performed, the method Display Team Details occurs when status is True, it will initialise on False. Array Lists for Team is instantiated. External CSV files are read in. comparison operators are utilised where if fields have any content i.e. <0 try/catch statements are implemented. The catch statements will use the methods File not found etc.
5. **Implementation**: Writing the object-oriented code with MVP in mind, aiming to provide the base functionality before any ‘gold plating’
6. **Testing**: use cases are tested with ‘should’ statements, When the New User clicks “Save New Team” The form ‘should’ return to empty and CSV ‘should’ contain the new record.

# Part 2 – Application Development

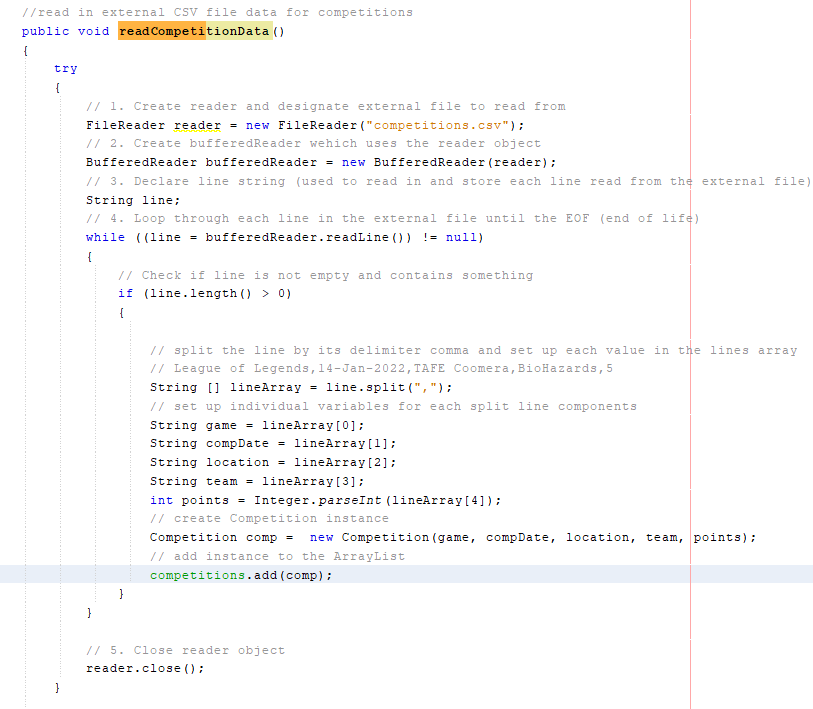
## Task 3 – Application Development

### Selection, Iteration and Sequence Constructs:

* Screenshot --- or --- COPY/PASTE of your Java code demonstrating at least one (1) **IF** or IF/ELSE construct  
    
    
  Screenshots --- or --- COPY/PASTE of your Java code demonstrating at least two (2) Iteration or loop constructs  
    
  Text

  Description automatically generated
* Screenshots --- or --- COPY/PASTE of your Java code demonstrating usage of user-created methods  
  

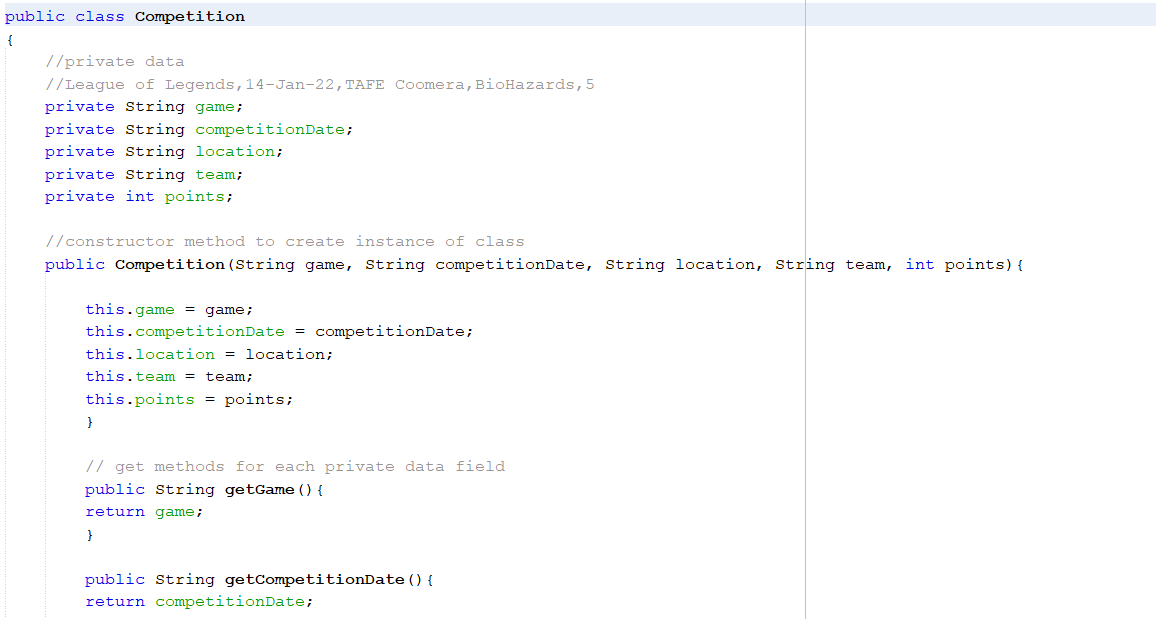
### Read/Write:

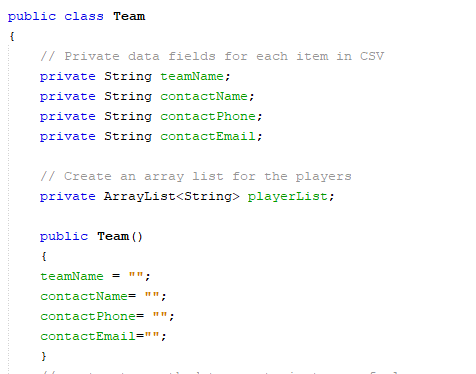
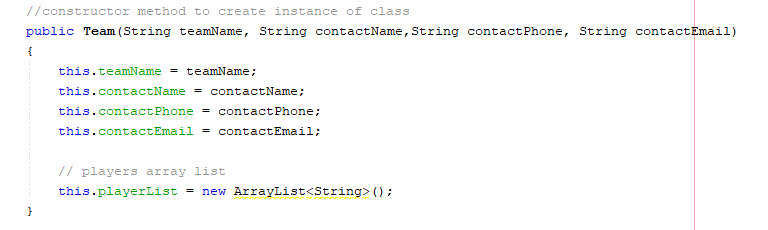
* Screenshot --- or --- COPY/PASTE of your Java code displaying what you have used for the program to read from an external CSV file
  + – Include a demonstration of an array used when reading from the file  
    
* Screenshot --- or --- COPY/PASTE of your Java code used to **write** to the external CSV file
  + – Include a demonstration of an array used when writing to the file

Text

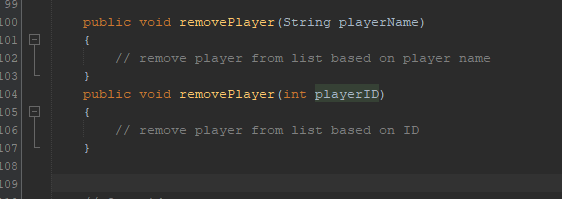
Description automatically generated

### Classes

* Screenshot --- or --- COPY/PASTE of your Java customised (user-defined) class with at least 4 data fields.  
  User-defined class holds 5 data fields below (game, competition date, location, team, points)   
  
* Screenshot --- or --- COPY/PASTE of another Java class making use of
  + At least 4 instance variables
  + Method overloading
  + User-defined objects combined into a data structure

Method Overloading:



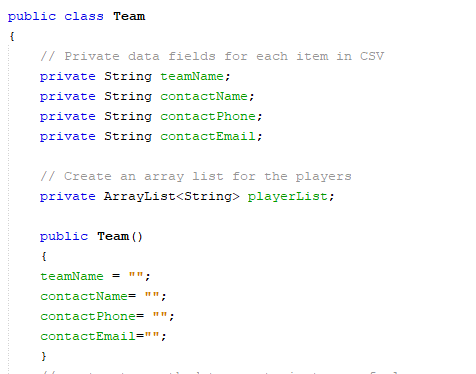
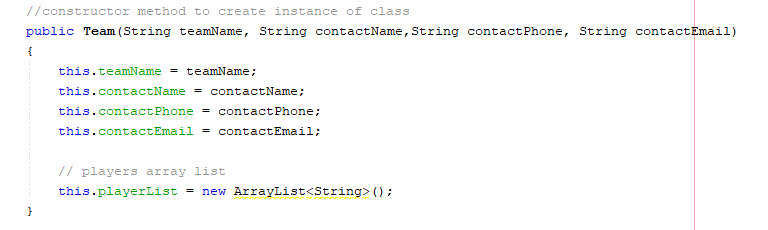
User defined object:

Text

Description automatically generated

* Screenshot --- or --- COPY/PASTE of the usage of two constructors in one class

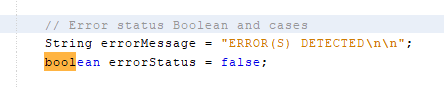
*Constructor method contains: Access specifier, Method name, Parameter list*

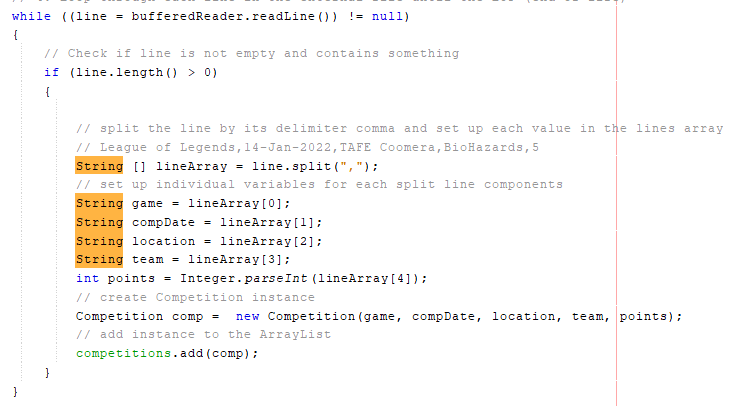
### Data Types, Operators, Expressions

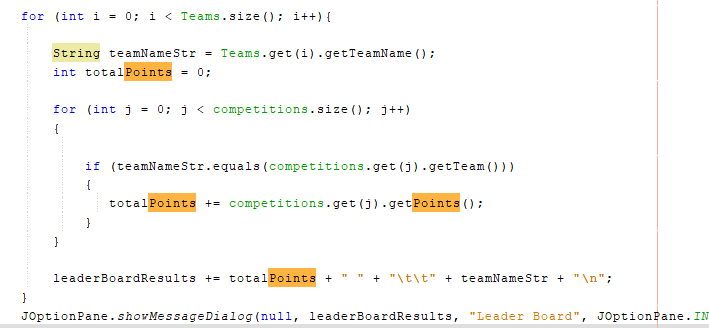
* Screenshot --- or --- COPY/PASTE of your Java code demonstrating usage of variables of at least 3 different data types

Boolean



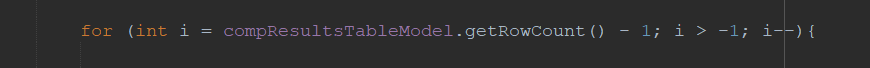
String

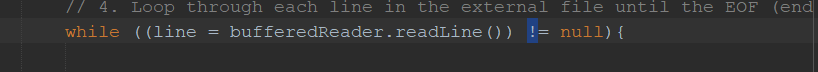
  
Integer (points)



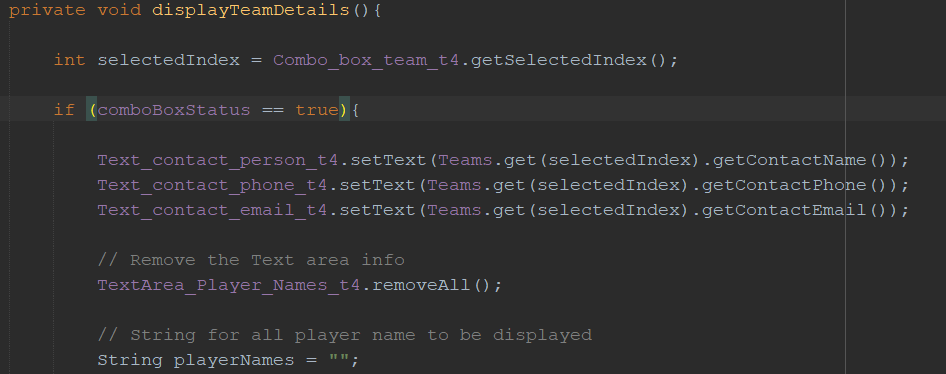
* Screenshots --- or --- COPY/PASTE of your Java code showing usage of at least 3 Operators from the following: (arithmetic, assignment, comparison, logical operators)  
  Comparison operator: Graphical user interface

  Description automatically generated

Arithmetic operator:   
  
Logical operator: ! (NOT EQUAL TO NULL)

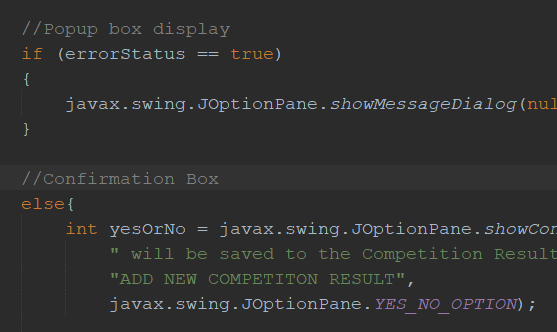


* Screenshots --- or --- COPY/PASTE of your Java code demonstrating usage of Expressions on at least 3 different occasions (expressions can be those used in conditional statements for if/else if or loops)



Text

Description automatically generated



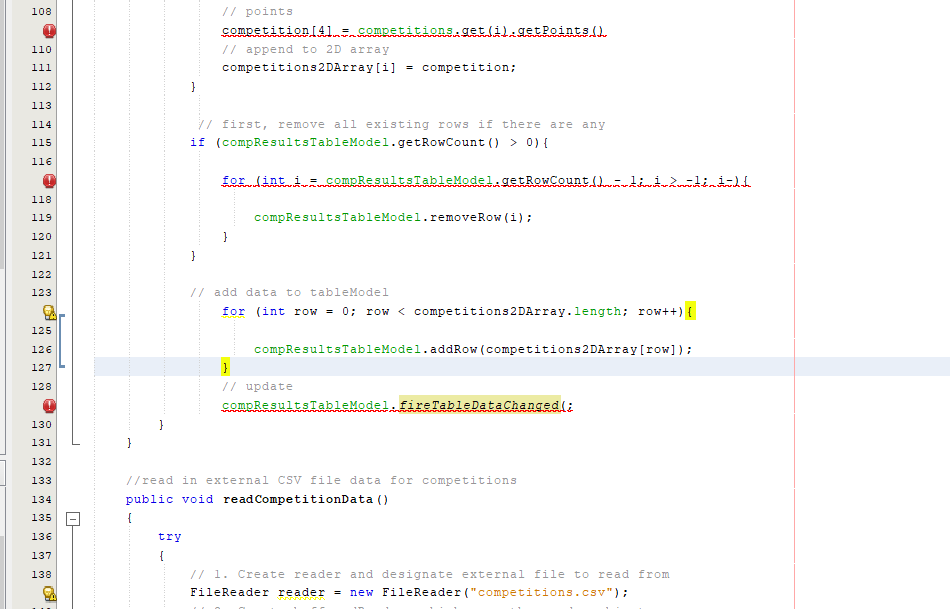
## Task 4 – Development Requirements

### Debugging:

* Screenshots --- or --- COPY/PASTE of (at most 5) syntax and logic errors, displaying usage of breakpoints and watches

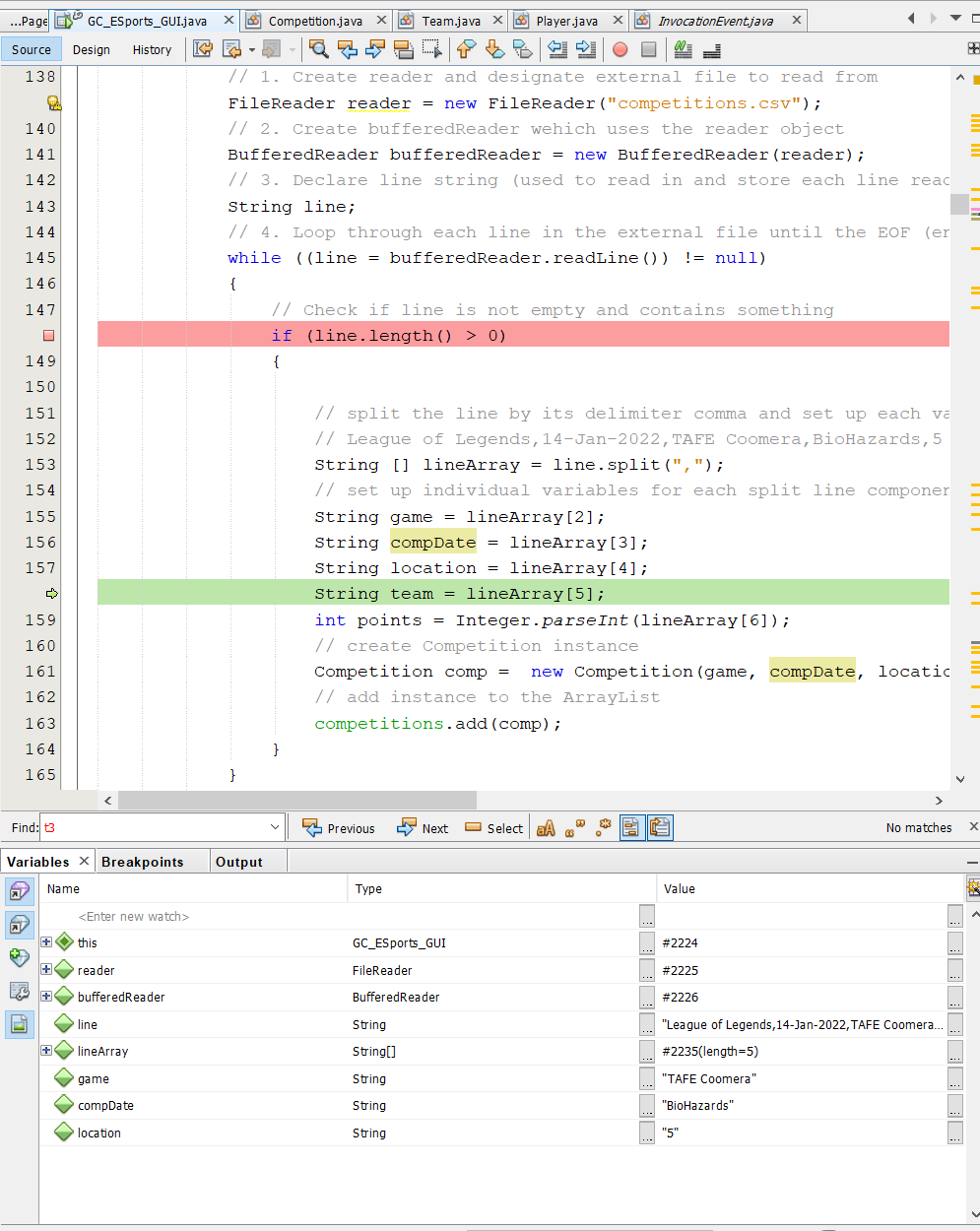
Syntax error x3:

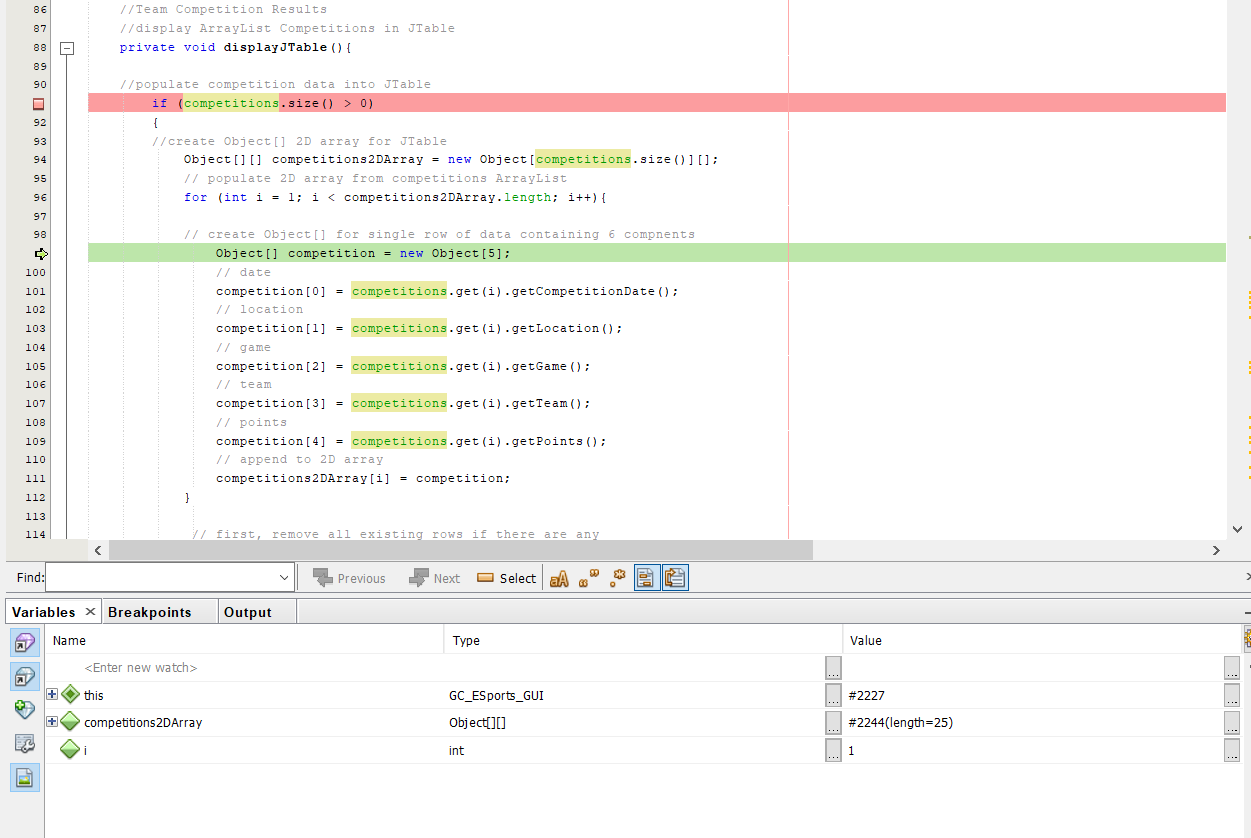
1. Missing semi colon: getPoints();
2. Missing – decrement after i-
3. Missing closing bracket after fireTableDataChanged();



Logic error x2:

1. Array list numbers are incorrect, in the variables The data is skewed e.g. Comp date is associated with the ‘Bio Hazards Team’ this allows one to see what the system is using in each variable

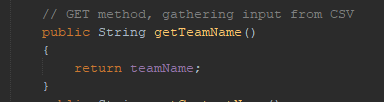


1. 

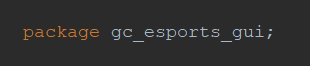
Logic error in the for loop, the int I should be starting value of 0 not 1. The Variables shows the int I value is 1

### Organisational Standards

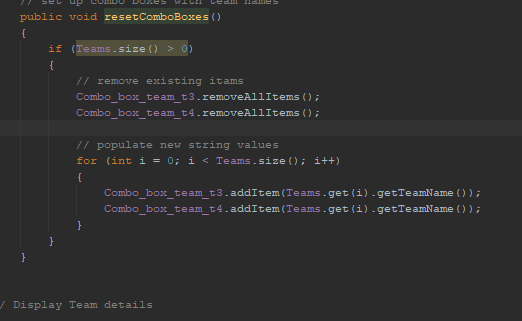
* Screenshot --- or --- COPY/PASTE of your Java code meeting naming conventions and a description of the applicable naming convention (e.g. variable and method declarations)

method is in camel case e.g. getTeamName()  


package is in snake case e.g. package gc\_esports\_gui;



* Screenshot --- or --- COPY/PASTE of your Java code meeting layout conventions and a description of the applicable layout convention (e.g. tab indentations, use of curly bracket alignment and spacing between operators and operands in expressions)

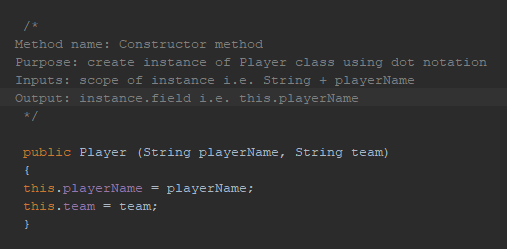
*Layout conventions are followed. There is space after the if and before the bracket, Curly brackets have their own line and opening bracket is aligned to the closing bracket. There is space around any operators i.e. > or = . Space is utilised between unrelated code blocks.*   


* Screenshot --- or --- COPY/PASTE of your Java code meeting documentation standards and a description of the demonstrated documentation standard (e.g. header and method comments, version control)

*Header comment incl version control*

Text

Description automatically generated

*Method comment*  


### Functionality testing with outcomes (#1 example provided - additional 5 tests required):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Function to test** | **Test Steps** | **Expected Result** | **Actual Result** | **Fix Required** | **Status**  **Pass/Fail** |
| 1. Add a new team (test with no inputs) | 1.1 Launch app 1.2 Click ADD NEW TEAM tab 1.3 Enter nothing for any of the text fields 1.4 Click SAVE NEW TEAM button | Pop-up displays 5 error messages – requiring inputs for: - team name - contact name - phone - email - player’s names | Pop-up displays 4 error messages – requiring inputs for: - team name - contact name - phone - email | Player’s names to be validated | FAIL (on first test)  Fix made and retested - PASS |
| 2.Add new competition result | * 1. Launch app   2. Add fields in: Date, location, game, Points   3. Select Team from Combo Box | Pop- up to confirm Save Yes or No option. | Pop up display  If YES selected, saves to CSV and wipes the form clear | None | pass |
| 3. Update an existing team | * 1. Launch app   2. Select team from Combo Box   3. Alter details | Pop up to confirm alteration, then when application is reopened, the change should be evident | Pop up and Yes selected, team member was altered. Tested removing a compulsory field and Error message works | None | Pass |
| 5. Add a new team (with full inputs) | 1.1 Launch app 1.2 Click ADD NEW TEAM tab 1.3 Enter details for all of the text fields 1.4 Click SAVE NEW TEAM button | Pop up “confirm add new team” Yes or No  Once Yes selected, form is wiped clear | Pop up confirmation evident | None | Pass |
| 6. Combo Box testing | * 1. Launch app   2. After adding new team, testing if it is instantly available in the combo box | Confirmation of add new team.  Team in Tab 3 and 4 should have the new team | Yes the new team was available | None | pass |

# Part 3 – Hand-Over

## Task 5 – Conclusion

* Summary of the development process including a checklist of developed features:

**Planning**

Development was to include the following features as planned, these have been met

* + Ability to add new entries in the following
  + New competition result i.e. date, location, game, team, points earned
  + New team i.e. team name, contact person, phone, email, player names
  + Ability to Update existing team data
  + Change contact person name, phone, email
  + Add or remove any player within team
  + Ability to save data to desk when application is closed
  + Ability to read previously entered data when application is opened
  + Ability to display the retrieved data in a list or table

**Development**

Creating the MVP with base functionalities. Version control was utilised where prior editions are saved and able to fall back upon.

**Testing**

The functionality testing was completed and application operates as it should, CSV files are written in the correct format and are all saving correctly.

* Email to the Client to request a meeting to discuss and review the developed application:

**EMAIL TO CLIENT**

|  |  |
| --- | --- |
| **To:** | mark@uptownbiz.com |
| **From:** | [shatha@downtown.com](mailto:shatha@downtown.com) |
| **Subject:** | Meeting to discuss and review Gold Coast E-Sports Desktop Application |
| **Message:** | Hi Mark, The application has been completed and checklist has been met and tested for bugs. Let’s arrange a meeting to go through the application for your feedback and ensure that it is meeting your business requirements  Regards,  Shatha |

* A summary of the feedback from the meeting, including planned solutions to any issues raised in the meeting.
* Tab one, arrange By features. Arrange by team, score or alphabetical (A-Z or Z-A)
* Date could be using a selector rather than everyone inputting with various styles 10/9 or 10/9 or 10th September
* Hover over explanations to explain the fields