State Board of Cricket Council –V3.0 *

State Board of Cricket Council

State Board of Cricket Council (SBCC) is one of the leading cricket selection academies in the state. They are in need of an automated system that should manipulate the player details provided and also find the players who have secured star rating between a specific range from the database.

You being their software consultant have been approached to develop a pilot java application which can be used by the admin for the above mentioned requirement.

Click below to download Requirement Document(s)

Requirement Document - 1/7

Requirement Document - 2/7

Requirement Document - 3/7

State Board of Cricket Council – Running Case study

Requirement 3: Validate player details

The State Board of Cricket Council (SBCC) wants the system to perform some validation checks so that only for valid data the Player profile should to be created. Hence you being their software consultant have been approached to integrate this validation functionality into the existing system.

The parsePlayerDetails method has to invoke the validatePlayerId method in the SBCCUtility class. If the validatePlayerId method returns true then, this method has to parse the player details, create the Player and return the same. If the validatePlayerId method returns false then, this method has to return null.

Component Specification: SBCCUtility Class

Component Name	Type(Class)	Attributes	Methods	Responsibilities
Validate player details and create player	SBCCUtility		public Player parsePlayerDetails(String playerDetails)	This method takes the String which holds all the details of a Player as an argument. This method has to invoke the validatePlayerId method in the SBCCUtility class by passing the playerId as a parameter, If

				the playerId is valid it should parse the data stor ed in the String , Invoke the calculateTotalRuns method in the player class and create the Player and return the same else return null.

Validation Rules:

The playerId has to be validated based on all the following constraints in the validatePlayerId method of the SBCCUtility class

1) The playerId should contain exactly 9 characters.
2) The first four characters in the playerId should be alphabets in uppercase.
3) The next four characters in the playerId should be digits.
4) The Last character in the playerId should be an alphabet in uppercase.

Examples of Valid playerId

AXCD6422D, JHDF9876J

Examples for Invalid playerId

AXC1234F – The playerId does not contain exactly 9 characters
AXC32234F – The Fourth character in the playerId is not an alphabet.
ACDSH234F – The fifth character in the playerId is not a digit.
ASDF87655 – The last character in the playerId is not an alphabet.
ASDF1234d – The last character in the playerId is not an alphabet in uppercase.

Component Specification: SBCCUtility Class

Component Name	**Type(Class)**	**Attributes**	**Methods**	**Responsibilities**
Validating the playerId	SBCCUtility		public boolean validatePlayerId(String playerId)	This method should validate the playerId, if valid return **true** else return **false**.

Component Specification: Player (Model Class)

Component Name	**Type (Class)**	**Attributes**	**Methods**	**Responsibilities**
	Player	String playerId String playerName int matchesPlayed int runScored String playingZone	Include all necessary Getters and Setters for all the attributes Provide a no argument and a five argument constructor in the given order playerId, playerName, matchesPlayed, runScored and playingZone.	
Calculate the total runs scored by the Player	Player		public int calculateTotalRuns(String[] securedRuns)	This method takes the String array as an argument which contains the runs scored by the player in each match. It has to calculate the total runs scored by the player by summing the runs scored by the player in each match and return the sum.

In the UserInterface class, **in the main method provided, fill the code to produce the output as shown in the Sample input and Output.**

When the user selects option **1 i.e., Validate player details**, it should get the player details from the user and display the player details if the player id is valid else display "**Please provide a valid record**".

When the user selects option **2 i.e.,** Exit, display the message "**Thank you for using SBCC application**" and end the program.

OVERALL DESIGN CONSTRAINTS:

- The Player class should be inside the package com.sbcc.model
- The SBCCUtility class should be inside the package com.sbcc.utility
- The UserInterface class should be inside the package com.sbcc.main
- Adhere to the design specifications mentioned in the case study.
- **The classes and methods should be declared as public and all the attributes should be declared as private.**
- **Do not change or delete the class/method/attributes, names or return types which are provided to you as a part of the base code skeleton.**
- Please make sure that your code does not have any compilation errors while submitting

Sample Input and Output 1 [Values given in bold represents the input]:

1. Validate player details

2. Exit

Enter your choice

1

Enter the player details

HXCB1234D:Dhoni:5:20:130:55:102:100:North:Batsman:3:1

Player id: HXCB1234D

Player name: Dhoni

Matches played: 5

Total runs scored: 407

Playing zone: North

1. Validate player details

2. Exit

Enter your choice

1

```
Enter the player details

HXC234D:Dhoni:5:20:130:55:102:100:North:Batsman:3:1

Please provide a valid record

// Note: Display the message << Please provide a valid record>> if the parsePlayerDetails method returns null

1. Validate player details

2. Exit

Enter your choice

2

Thank you for using SBCC application
```