軟體測試 HW01 季後賽

1.設計方法概述

(1)建構類別Team，類別中包含隊伍名，所屬聯盟，所在區域，勝場數，敗場數，及勝率計算和比較兩種方法

```
class Team implements Comparable<Team> {
   String name;
   String league; // 聯盟: AL 或 NL
   String division; // 分區: East, Central, West
   int wins;
   int losses;
   int seed;
   //初始化球隊資訊，包含防錯機制
   public Team(String name, String league, String division, int wins,
int losses) {
       if (name == null || league == null || division == null || wins
< 0 || losses < 0) {
           throw new IllegalArgumentException("無效輸入");
       }
       this.name = name;
       this.league = league;
       this.division = division;
       this.wins = wins;
       this.losses = losses;
   }
   // 計算勝率
   public double getWinRate() {
       return (double) wins / (wins + losses);
   }
   // 比較方法，用於排序球隊，勝率高者排前面
   @Override
   public int compareTo(Team team) {
       return Double.compare(team.getWinRate(), this.getWinRate());
   }
}
```

(2)以聯盟為單位建立兩個List<Team>

```
List<Team> alTeams = new ArrayList<>();
List<Team> nlTeams = new ArrayList<>();
```

(3)用for迴圈讀取球隊各項數據數據，並加入異常偵測

偵測目標:

偵測確保輸入格式正確

勝場數 + 敗場數的 總場數為**162**

聯盟名稱正確(AL / NL)

```
for (int i = 0; i < 30; i++) {
   try {
       String[] input = scanner.nextLine().split(" ");
       if (input.length != 5) {
           throw new IllegalArgumentException("輸入異常. Expected: TeamName
League Division Wins Losses");
       }
       String teamName = input[0];
       String league = input[1];
       String division = input[2];
       int wins = Integer.parseInt(input[3]);
       int losses = Integer.parseInt(input[4]);
       // 檢查總場次是否一致
       totalGames += wins + losses;
       Team team = new Team(teamName, league, division, wins, losses);
       if (league.equals("AL")) {
           alTeams.add(team);
       } else if (league.equals("NL")) {
           nlTeams.add(team);
       } else {
           throw new IllegalArgumentException("聯盟名稱輸入異常. 應該是 AL or
NL.");
       }
       if(totalGames != 162){
           throw new IllegalArgumentException("場次輸入異常. 總場次應該是
162.");
       }
   } catch (Exception e) {
       System.out.println("Error processing input: " + e.getMessage());
       i--; // 重新輸入該隊資訊
   }
}
```

(4)透過勝率比對取出各區域冠軍

```
// 取得分區冠軍
public static Map<String, Team> getChampions(List<Team> teams) {
   Map<String, Team> champions = new HashMap<>();
   for (Team team : teams) {
       champions.putIfAbsent(team.division, team);
       if (team.getWinRate() >
champions.get(team.division).getWinRate()) {
           champions.put(team.division, team);
       }
   }
   return champions;
}
```

(5)將各區域冠軍照勝率排序

```
// 依勝率排序分區冠軍
Collections.sort(alChampionTeams);
Collections.sort(nlChampionTeams);
```

(6)將所有非區域冠軍的球隊進行勝率排序已獲得該聯盟的外卡隊伍

```
// 取得外卡球隊
List<Team> alWildCards = getWildCard(alTeams, alChampionTeams);
List<Team> nlWildCards = getWildCard(nlTeams, nlChampionTeams);
```

```
// 取得外卡球隊(非分區勝率前三高)
public static List<Team> getWildCard(List<Team> teams, List<Team>
champions) {
   List<Team> wildCards = new ArrayList<>();
   for (Team team : teams) {
       if (!champions.contains(team)) {
           wildCards.add(team);
       }
   }
   Collections.sort(wildCards);
   return wildCards.size() > 3 ? wildCards.subList(0, 3) : wildCards;
}
```

(7)輸出對戰圖表

2.執行畫面

(1)正常執行(完整測資放在src資料夾的測資.txt)

```
(AMERICAN LEAGUE)
|WILDCARD | ALDS  | ALCS    | WORLD SERIES |
TOR 6 ---
MIN 3 --- ? -----
      HOU 2 ----- ? -------
TEX 5 ---
TB 4 --- ? -----
      BAL 1 ----- ? ------- ? -------
                                      ?
ARI 6 --- ? ----- ? ------- ? -------
MIL 3 ---
      LAD 2 ---
MIA 5 --- ? ----- ? -------
PHI 4 ---
      ATL 1 -----
|WILDCARD | NLDS  | NLCS    | WORLD SERIES |

(NATIONAL LEAGUE)

Process finished with exit code 0
```

(2)輸入異常

```
BLA AL 162 0
Error processing input: 輸入異常. Expected: TeamName League Division Wins Losses
BLA BL EAST 162 0
Error processing input: 聯盟名稱輸入異常. 應該是 AL or NL.
BLA AL EAST 160 0
Error processing input: 場次輸入異常. 總場次應該是162.
```

3.參考資料與使用工具及比例 (包含 AI)

(1)ChatGPT:協助生成測資，查詢Comparable及List相關用法，生成註解