Humza Salman

CS 2337.003

Professor Jason Smith

10 November 2019

**Project 2 Pseudocode**

* **Main**
  + **Main**
    - Retrieve input file name and output file name
    - Open keyfile.txt
    - Create hash table to store keys
    - Check to see if keyfile.txt opened
      * String category
      * While not end of keyfile.txt
        + Store category in string for each category encountered
        + Key’s will follow after the category, each in new line

store in hash table with the key being the ‘Keys’ and the value being the category

* + - Close input from keyfile.txt
    - Open input file to the input file name
    - Create new hash table for players
    - Check to see if opened
      * While not end of input file
        + read the line
        + check to see if last line in file empty

if true then break

* + - * + parse the Team, Name, and Key
        + string Plate Appearance

search the hash table for the key and return the statistic

* + - * + check to see if the name exists in the hash table for players

if it exists, then update the player’s statistics

if not, create a new player and insert into hash table

update the player’s statistics

* + - close input file
    - open output file
    - output player statistics for both teams
    - output league leaders for each category
    - close output file
* **Output player statistics – (Hash Table, file output) – void**
  + Create arrays to store away and home team
    - Go through hash table and store each player in either away or home team
  + Sort away team and home team arrays
  + Output Away Team
    - for each player in array
      * output player according to format
  + Output Home Team
    - For each player in home team array
      * Output player according to format
* **Output league leaders – (Hash Table, file output, category) – void**
  + Player First, second, and third, value
    - Traverse through hash table
      * If the category is strikeouts
        + Value is the player with most strikeouts
        + If value is less than first, then third is second, second is first, and first is value
        + Else if the value is less than second and the value is not equal to first, then the third is second and the second is value
        + Else if the value is less than third and the value is not second and not first, then the third is value
      * Else if the category is not strikeouts
        + Value is set as the first iterable player’s value for that category
        + If value is greater than first, then third is second, second is first, and first is value
        + Else if the value is greater than second and not equal to first, then third is second, and second is value
        + Else if the value is greater than third and is not second and not first, then third is value
  + Output the category
  + Keep track of first and second place ties
  + Output all players with first place – Output Player Names
  + If the first player ties are less than or equal to 3, then output players in second place – Output Player Names
  + If there are no ties for first and second players, then output all players with third place – Output Player Names
* **Output Player Names – (output, hash table, value, category) -- int**
  + Keep track of ties
  + Iterate through hash table
    - If player matches value for given category, print according to format
    - Increment ties
  + Return ties
* **Player**
  + **Members**
    - Hits, Outs, Walks, Hit By Pitch, Sacrifices, Errors – private
    - Name, Team -- private
  + **Functions – parameters – return type**
    - **Operator overload > -- (Player) – bool** 
      * return scope’s player’s lowercase name greater than player’s lowercase name
    - **Operator overload < -- (Player) –bool**
      * return scope’s player’s lowercase name less than player’s lowercase name
    - **Operator overload == -- (Player) -- bool**
      * return scope’s name equal to player’s name
    - **Operator overload << -- (output, Player) – bool**
      * output the player’s statistic according to the format in project documentation
    - **Calculate At-Bats – () -- int**
      * Return hits + outs + strikeouts + errors
    - **Calculate plate Apperaences – () – int**
      * Return atBats + walks + hitbypitch + sacrifices
    - **Calculate Batting Average – () – double**
      * Check to see if calculateAtBats = 0, if true then return 0;
      * Return (hits / calculateAtBats)
    - **Calculate OBPercentage – () – double**
      * Check to see if calculate plate apperances = 0, if true then return 0
      * Return (hits + walks + hitbypitch)/plate appearences
    - **Update Statistics -- (string stat) – void**
      * Check what the stat is, and increment the player’s stat according to the statistic
    - **Output statistics – () – void**
      * output the statistics of player to file according to format