/\*

Program 1 - Snowball Fight Team Score Checker

    It allows the user to enter a team’s ID and displays the number of hits for that team

Created by: John Akujobi

Date: 22nd of January 2023

Class: CSC 250 Fall 2023

\*/

/\* Detailed program requirements

A text file named snowball.txt contains a 3-digit Team ID, number of times the team hit a member of another time (hits), and the number of times a member of this team was hit (times hit) for each team participating in a campus-wide snowball fight.  The data is organized on up to 50 lines in the data file, as shown below.

182 37 12

837 14 25

374 29 28

…

Write a program that allows the user to enter a team’s ID and displays the number of hits for that team.  You must use the binary search algorithm studied in Unit 1 in this program.  Your program should do the following:

1. Read the data from the file into partially filled, parallel arrays.

2. Sort the arrays, in parallel, putting the IDs in increasing order and keeping the number of hits and times hit in parallel with them. (When you swap two elements in the ID array, also swap the corresponding elements in the hits and times hit arrays.)

3. Enter a loop that does the following until the user chooses to quit:

    1. Read a team ID from the user

    2. Use the Binary Search algorithm to find the number of hits and times hit for that team.

    3. Print the number of hits, and times hit for the team requested. If the hits is greater than the times hit, print a Congratulations message.  If the ID entered is not in the data, print a “Not Found” message.

Your program must be modular, with separate functions to: read the data file, sort the arrays in parallel, input a number from the user, search, and print the results.  Your output should be well-organized, neat, and easy to read.

\*/

/\* Draft FLowchart

- Print the welcome message to the screen

- Open the file

- Fill the file contents into the 3 arrays; ArrayTeamID, ArrayAttackHits and ArrayReceivedHits

- Prints the menu

    - Option 1: Check team Score

    - Option 2: Quit the Program

- Receive user menu input, verify the input and call the necessary functions

- Read the team ID, and check if it is in the right format

- Search the list of array and find the team or print an error message

- Print the team ID, Attack hits, Received Hits

- Print Congratulations message if the Attack hits are more than the Received hits

- Repeat the loop

- After the user chooses the quit option, close the file

- End the program

\*/

#include <iostream>

#include <fstream>

#include <string>

#include <iomanip>

using namespace std;

/\* PrintWelcomeMessage Design

    - Prototype and Function Call

        - `Void PrintWelcomeMessage()

        - `PrintWelcomeMessage\*\*\*\*\*\*()\*\*\*\*\*\*`

    - Description

        - Prints a welcome message to the screen

    - Pseudocode

        - Print statements

            - Welcome to The Snowball Fight Team Score Checker v1.0

            - This program is designed to help you check the scores of a team

            - And it tells you if they won or not.

            - Lets get started

        - Print Empty line twice

\*/

void PrintWelcomeMessage();

void PrintWelcomeMessage()

{

    cout <<endl;

    cout <<endl;

    cout << "Welcome to The Snowball Fight Team Score Checker v1.0" << endl;

    cout << "This program is designed to help you check the scores of a team" << endl;

    cout << "And it tells you if they won or not." << endl;

    cout << "Lets get started" << endl;

    cout <<endl;

    cout <<endl;

}

//! rEADFILE v1.0

// //This function reads the file and fills the arrays: ArrayTeamID, ArrayAttackHits and ArrayReceivedHits

// void ReadFile(int ArrayTeamID[], int ArrayAttackHits[], int ArrayReceivedHits[], int &ArraySize);

// void ReadFile(int ArrayTeamID[], int ArrayAttackHits[], int ArrayReceivedHits[], int &ArraySize)

// {

//     ifstream file("snowball.txt");

//     //Open the file

//     //file.open("snowball.txt");

//     //Check if the file is opened

//     if (!file)

//     {

//         cout << "Error opening file" << endl;

//         exit(1);

//     }

//     //Fill the file contents into the 3 arrays; ArrayTeamID, ArrayAttackHits and ArrayReceivedHits

//     int i = 0;

//     while (file >> ArrayTeamID[i] >> ArrayAttackHits[i] >> ArrayReceivedHits[i])

//     {

//         i++;

//     }

//     ArraySize = i;

//     //Close the file

//     file.close();

// }

// Function to read the file and fill the arrays

void ReadFile(int ArrayTeamID[], int ArrayAttackHits[], int ArrayReceivedHits[], int &ArraySize);

void ReadFile(int ArrayTeamID[], int ArrayAttackHits[], int ArrayReceivedHits[], int &ArraySize)

{

    ifstream file("snowball.txt");  // open file

    // check if file is open

    if (!file.is\_open()) {

        cout << "Error: Could not open file." << endl;

        exit(1);

    }

    // fill the file contents into the 3 arrays

    int i = 0;

    while (file >> ArrayTeamID[i] >> ArrayAttackHits[i] >> ArrayReceivedHits[i])

    {

        i++;

    }

    ArraySize = i;

    file.close();

    return;

}

//This function prints the menu

void PrintMenu();

void PrintMenu()

{

    cout << "Menu" << endl;

    cout << "1. Check Team Score" << endl;

    cout << "2. Quit the Program" << endl;

    cout << "Enter your choice: ";

}

//This function reads the team ID and checks if it is in the right format

int GetTeamID();

int GetTeamID()

{

    int TeamID;

    cout << "Enter the Team ID: ";

    cin >> TeamID;

    if (TeamID < 100 || TeamID > 999)

    {

        cout << "Invalid Team ID" << endl;

        cout << "Please enter a valid Team ID" << endl;

        GetTeamID();

    }

    return TeamID;

}

//This function searches the list of array and find the team or print an error message

int SearchTeam(int ArrayTeamID[], int ArrayAttackHits[], int ArrayReceivedHits[], int ArraySize, int TeamID);

int  SearchTeam(int ArrayTeamID[], int ArrayAttackHits[], int ArrayReceivedHits[], int ArraySize, int TeamID)

{

    int i = 0;

    while (i < ArraySize)

    {

        if (ArrayTeamID[i] == TeamID)

        {

            cout << "Team ID: " << ArrayTeamID[i] << endl;

            cout << "Attack Hits: " << ArrayAttackHits[i] << endl;

            cout << "Received Hits: " << ArrayReceivedHits[i] << endl;

            if (ArrayAttackHits[i] > ArrayReceivedHits[i])

            {

                cout << "Congratulations! You won the snowball fight" << endl;

            }

            break;

        }

        i++;

    }

    if (i == ArraySize)

    {

        cout << "Team ID not found" << endl;

    }

    int TeamKey = i;

    return TeamKey;;

}

//This function prints the team ID, Attack hits, Received Hits

void PrintTeamScore(int ArrayTeamID[], int ArrayAttackHits[], int ArrayReceivedHits[], int ArraySize, int TeamID);

void PrintTeamScore(int ArrayTeamID[], int ArrayAttackHits[], int ArrayReceivedHits[], int ArraySize, int TeamID)

{

    cout << "Team ID: " << ArrayTeamID[TeamID] << endl;

    cout << "Attack Hits: " << ArrayAttackHits[TeamID] << endl;

    cout << "Received Hits: " << ArrayReceivedHits[TeamID] << endl;

    //Check if the team won the snowball fight

    if (ArrayAttackHits[TeamID] > ArrayReceivedHits[TeamID])

    {

        cout << "Congratulations! You won the snowball fight" << endl;

    }

}

// //This function prints the goodbye message

// void PrintGoodbyeMessage();

// void PrintGoodbyeMessage()

// {

//     cout << "Thank you for using The Snowball Fight Team Score Checker v1.0" << endl;

//     cout << "Goodbye" << endl;

// }

//Sort the arrays parallel to each other

void SortArrays(int ArrayTeamID[], int ArrayAttackHits[], int ArrayReceivedHits[], int ArraySize);

void SortArrays(int ArrayTeamID[], int ArrayAttackHits[], int ArrayReceivedHits[], int ArraySize)

{

    int i, j, temp;

    for (i = 0; i < ArraySize; i++)

    {

        for (j = i + 1; j < ArraySize; j++)

        {

            if (ArrayTeamID[i] > ArrayTeamID[j])

            {

                temp = ArrayTeamID[i];

                ArrayTeamID[i] = ArrayTeamID[j];

                ArrayTeamID[j] = temp;

                temp = ArrayAttackHits[i];

                ArrayAttackHits[i] = ArrayAttackHits[j];

                ArrayAttackHits[j] = temp;

                temp = ArrayReceivedHits[i];

                ArrayReceivedHits[i] = ArrayReceivedHits[j];

                ArrayReceivedHits[j] = temp;

            }

        }

    }

}

//Main Function

// !\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

// !\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

int main()

{

    int Option;

    //Variables

    const int SIZE = 100;

    int ArrayTeamID[SIZE], ArrayAttackHits[SIZE], ArrayReceivedHits[SIZE];

    int ArraySize;

    int TeamKey; //Position of the team in the array and the key for the array position

    int TeamID; //Team ID of the team. It is a 3 digit number

    //Prints the welcome message

    PrintWelcomeMessage();

    //Opens and reads the file then fills the arrays

    ReadFile(ArrayTeamID, ArrayAttackHits, ArrayReceivedHits, ArraySize);

    //Loop that runs the program until the user quits

    do{

        //Prints the menu

        PrintMenu();

        //Receives the user input

        cout << "Enter your choice: ";

        cin >> Option;

        //Executes statements for Option 1: Check Team Score

        if (Option == 1)

        {

            //Gets the Team ID from the user

            TeamID = GetTeamID();

            //Searches the array for the Team ID

            TeamKey = SearchTeam(ArrayTeamID, ArrayAttackHits, ArrayReceivedHits, ArraySize, TeamID);

            //Prints the team score

            PrintTeamScore(ArrayTeamID, ArrayAttackHits, ArrayReceivedHits, ArraySize, TeamKey);

        }

        //Executes statements for Option 2: Quit the Program and  exits the program

        else if (Option == 2)

        {

            exit(0);

        }

        //Takes care of invalid input and asks the user to enter a valid input until they do

        else

        {

            while (Option <1 || Option >2)

            {

                //Prints the error message

                cout << "Invalid Input" << endl;

                cout << "Please type in a valid option (Must be 1 or 2)" << endl;

                //Receives the user input

                cin >> Option;

            }

        }

        cout << "Would you like to check the score of another team" << endl;

    } while (Option != 2);

    return 0;

}