ENGM2283: Data Structures & Algorithms
Group Project: Olympic Athlete Tracker

Wehyee Tarlow - B00865607
Semilore Kayode - B00863886
Sarah Fraser - B00863783
Tristan Fraser - B00904591
Elias Francis - B00897478

Submitted to:
Dr. Farzaneh Naghibi
Department of Engineering
Dalhousie University

Due Date: Wednesday, April 12th, 2023

Introduction:

The Olympic Athlete Tracker is a program designed to track and manage athletes who participate in the Olympics. The program allows users to add new athletes, retrieve athlete information, display all athletes, sort athletes, remove a specific athlete, count the number of athletes, and clear all records.

Implementation:

The Olympic Athlete Tracker is implemented in C++. The program uses two linked lists, one for sprinters and another for soccer players. Each linked list stores pointers to dynamically allocated objects of the sprinter and soccer player classes, respectively. The sprinter and soccer player classes are derived from the athlete class, which stores information common to all athletes.

The program uses a menu-driven interface to allow users to interact with the system. Users can select from various options by entering a corresponding letter. For example, users can add a new athlete by selecting the "a" option, retrieve athlete information by selecting the "r" option, and so on.

When a user selects the "a" option, they are prompted to enter the type of athlete they would like to add (sprinter or soccer player). They are then prompted to enter the relevant athlete information including an id that is checked for duplication before being verified, which is stored in a new object of the corresponding class. The object is then dynamically allocated and its pointer is stored in the appropriate linked list.

When a user selects the "r" option, they are prompted to enter the id of the athlete they would like to retrieve. The program searches both linked lists for an athlete with the given name and displays the athlete's information if found.

When a user selects the "b" option, the program displays all athletes in the system. The program iterates through both linked lists and displays the information of each athlete.

When a user selects the "s" option, they are prompted to enter the type of athlete they would like to sort (sprinter or soccer player). The program then prompts the user to select the sorting method (distance or record time for sprinters and goals or assists for soccer players). The program then sorts the relevant linked list in ascending order based on the selected sorting method.

When a user selects the "d" option, they are prompted to enter the name of the athlete they would like to remove. The program searches both linked lists for an athlete with the given name and removes the athlete if found.

When a user selects the "c" option, the program displays the total number of athletes in the system. The program counts the number of objects stored in both linked lists and returns the sum.

When a user selects the "o" option, the program clears all athlete records. The program iterates through both linked lists and deallocates the memory used by each object.

Cases that may break the code ;

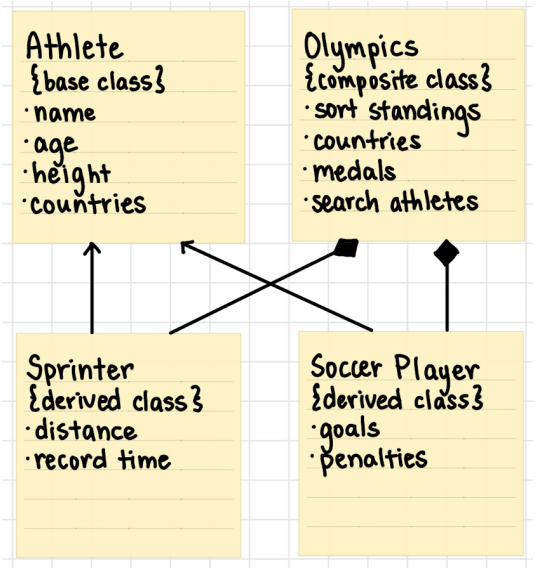
- When you insert a string instead of an int.

Memory Allocation:

The project, which is implemented using linked lists to store athlete information, has the potential for memory leaks due to the use of dynamically allocated objects without proper deallocation. However, these memory allocation issues are addressed through the use of the delete keyword to free up the memory used by objects that are no longer needed. The system utilizes two linked lists, one for sprinters and another for soccer players, to store pointers to dynamically allocated objects of the athlete, sprinter, and soccer player classes. The project report documents the design process, including UML diagrams and flow diagrams, as well as test cases completed to ensure the system functions as desired. The report also highlights potential memory leaks in the code and how they are addressed, as well as special data input cases that may lead to system failure. Overall, the project demonstrates effective memory management practices and provides a robust system for tracking Olympic athletes.

Changes Made Since Presentation:

- We added an ID so that if there are athletes with the same name we can differentiate between the two athletes. We also changed to use the ID to retrieve and remove as opposed to the name.
- We also tried to add a duplicate function but it was unsuccessful.
- We made repeated code in the main into called functions to make the application more concise
-



Examples of menu prompts:

```
------------------MENU------------------
Select from the following options:
a - add athletes
r - retrieve an athlete's information
b - display all athletes
s - sort athletes
d - remove a specific athlete
c - count the number of athletes
o - clear records
e - exit
----------------------------------------
```

Add

```
a

Enter the type of athlete you would like to add:
p - sprinter
k - soccer player
e - previous menu
p
Enter athlete's name: a
Enter athlete's age: 1
Enter athlete's country: 1
Enter an id (integars only!):1
Enter the distance in meters: 1
Enter the record time in seconds: 1
```

Retrieve:

```
r

Enter the name of the athlete you would like to retrieve: 2
Athlete Found
Name: c, Age: 2, Country: 2, Distance (m): 2, Record Time (s): 2
```

Count:

```
c

There are 3 total athletes in the Olympics.
There are 3 sprinters.
```

Sort (by distance for example)

```
s

Enter the type of athlete you would like to add:
p - sprinter
k - soccer player
e - previous menu
p

Enter how you would like to sort:
d - distance
t - record time
d

Standings based on distance:
Name: b, Age: 3, Country: 3, Distance (m): 3
Name: c, Age: 2, Country: 2, Distance (m): 2
Name: a, Age: 1, Country: 1, Distance (m): 1
```

Clear (clearing all)

```
o

Enter the records you would like to clear?
a - all records
p - sprinter records
k - soccer player records
e - previous menu
a

Are you sure you want to clear all records? (y/n): y
All records have been cleared.
```

Display:

- Empty:

```
b
There are no athletes in the Olympics
```

- With athletes:

```
b

Sprinters:
Name: Elias, Age: 20, Country: Canada, id: 20, Distance (m): 20, Record Time (s): 20

Soccer Players:
Name: Francis, Age: 42, Country: NewZealand, id: 9875, Goals: 365, Penalties: 1
```

Conclusion:

The Olympic Athlete Tracker is a simple yet useful program for managing athletes participating in the Olympics. The program allows users to add new athletes, retrieve athlete information, display all athletes, sort athletes, remove a specific athlete, count the number of athletes, and clear all records. The program is implemented using linked lists and dynamic memory allocation, making it efficient and scalable.