Micro Version of Facebook

Aniqua Azad

Design and Implementation

For the hash table, I made it an array of FBLinkedList. So the hash table is created as

FBLinkedList[] hashTable = new FBLinkedList[LENGTH_OF_TABLE];

For my particular LENGTH_OF_TABLE, I chose the number of slots to be 10, because I noticed that it distributed the keys evenly among the slots. I originally picked 13, an odd and prime number, but I noticed that more slots were null and one slot had a lot of keys.

Design for Division Method

Hash Function: For the hash function, I got the String name for each Person and obtained the ASCII code for each letter. I added up all the ASCII values and saved it to a variable called totalAscii. I then did totalAscii mod LENGTH_OF_TABLE, which is the number of slots in the hash table, to get the index of the Person key.

ex: Person p = new Person("Ava")

totalAscii = 65 + 118 + 97 = 280

totalAscii % LENGTH_OF_TABLE = 280 % 10 = 0

p will be stored in hashTable[0]

Design for Multiplication Method

Hash Function: For the hash function, I got the String name for each Person and obtained the ASCII code for each letter. I added up all the ASCII values and saved it to a variable called totalAscii. I then did (int)(Math.floor(LENGTH_OF_TABLE*((totalAscii * A) % 1))) to get the index of the key. I made A a final static variable and set it to 0.6180339887.

ex: Person p = new Person("Ava")

totalAscii = 65 + 118 + 97 = 280

totalAscii * A = 280 * 0.6180339887 = 173.049516836

(totalAscii * A) % 1 = 0.049516836

(int)Math.floor(LENGTH_OF_TABLE*((totalAscii * A) % 1 = 0

p will be stored in hashTable[0]

List of classes/subroutines/function calls

- **Person**: This class holds the name and the friends list of the Person. It also has methods which get the name and friend list. It also has methods which adds friends, deletes friends, and searches for the name of a friend.
    - getName(): Returns the name of the Person
    - getFriendsList(): Returns the friend list of the Person
    - addToList(): Adds a friend to the Person's friend list
    - deleteFromList(): Deletes a friend to the Person's friend list
    - SearchList(): Searches for a name in the Person's friend list
- **FBNode**: This is the node class used by FBLinkedList. It has methods which get and set the next nodes, and get data of the Person.
    - getNext(): Gets the next node
    - setNext(): Sets the next node
    - getData(): Gets the data of the Person for the node
- **FBLinkedList**: This class is the LinkedList class created specifically for this project. It has methods which insert a Person into a LinkedList, deletes a Person from a LinkedList, and searches for a Person with a given name. It also has a toString() method which gets the name from each node and adds it to a String and then is returned.
    - insert(): This method adds a Friend at the head to the LinkedList.
    - delete(): This method deletes Person p from the linked list. It goes through various checks to see if the linked list has any edge cases and does deletion based on the cases.
    - search(): This method searches for the String p(name) in each node. Once it finds it, it returns the node's data which contains name
    - toString(): This method gets the name from each node and returns the String
- **HashWithDiv**: This class is a micro-version of the popular web based media application, Facebook. It implements a HashTable to store people. This particular class uses the division method in hashing to store each person in the hash table; if there are any collisions, they are fixed with chaining. Users can choose to create an "account", add a friend, unfriend, check a person's friend list, and see if two people are friends.

- divHashFxn(): This method is the division method used to determine which slot in the hash table the Person will be added. It adds up the ASCII value of each letter of the person's name and mods the total values by the total number of slots.
- add50People(): This method creates and adds the 50 people to the hashTable
- addTempFriends(): This method adds friends to a small number of people from the list of 50 people
- chainedHashInsert(): This method gets the Person then calls the hash function to get the index. Then, the person is added to the hash table at the calculated index
- chainedHashDelete(): This method gets the Person then calls the hash function to get the index. Then, the person is deleted from the hash table at the calculated index
- chainedHashSearch(): This method uses the name given by the user and creates a temp Person. Then, it finds the index of the hash table where that name would be located. Then, it searches through the hash table at that index to find the name. Once the name is found, the corresponding Person is returned.
- checkIfFriends(): This method checks if two people are friends. If they are friends, the method returns "Yes"; if not, the method returns "No"
- option1(): This method allows the user to create a Person by entering a name. They are prompted to enter a name, which is then fixed to make the first char capitalized. Then, it adds the name to the names list. Finally, it adds the Person to the hash table by calling chainedHashInsert.
- option2(): This method allows users to make two people friends. They enter the 1st person's name, and the method uses chainedHashSearch to find the corresponding Person of that name. Then the user does the same for the 2nd person. Then, the method makes sure that the two people aren't already friends. If they're not, then each person is added to the others friend list. If they are friends, the user is notified.
- option3(): This method allows users to unfriend two people. They enter the 1st person's name, and the method uses chainedHashSearch to find the corresponding

Person of that name. Then the user does the same for the 2nd person. Then, the method makes sure that the two people are already friends. If they are, then each person is deleted from the others friend list. If they're not friends, the user is notified.

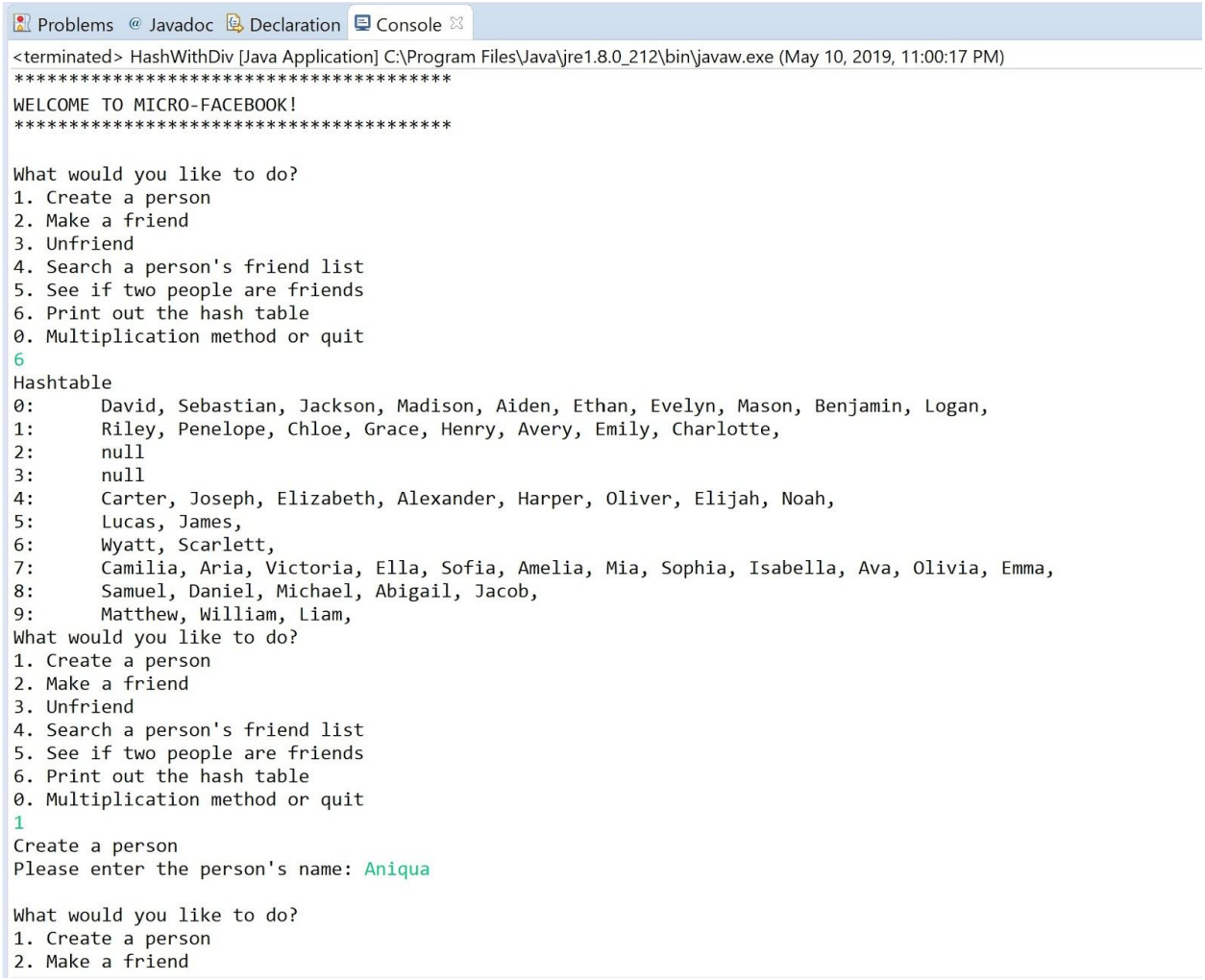
- option4(): This method allows the user to search any Person's friend list in Facebook. The user enters the name of the person. and the method uses chainedHashSearch to find the corresponding Person of that name. Fianlly, the Person's friend list is printed to the console.
- option5(): This method checks to see if two people are friends. They enter the 1st person's name, and the method uses chainedHashSearch to find the corresponding Person of that name. Then the user does the same for the 2nd person. Then, the method checks to see if the two people are friends. If they are, then "Yes" is printed. If not, then "No" is printed.
- option6(): This method prints out the hash table
- nameExists(): This method checks to see if a name exists on Facebook before any friending, unfriending, searching takes place.
- options(): This method lists the options for the user to choose from. If they enter 0, then they have the choice of terminating the program or seeing the same program but using the multiplication method.

- **HashWithMult**: This class is a micro-version of the popular web based media application, Facebook. It implements a HashTable to store people. This particular class uses the multiplication method in hashing to store each person in the hash table; if there are any collisions, they are fixed with chaining. Users can choose to create an "account", add a friend, unfriend, check a person's friend list, and see if two people are friends.
  - multHashFxn(): This method is the multiplcation method used to determine which slot in the hash table the Person will be added. It adds up the ASCII value of each letter of the person's name. Then it uses the equation m((key x A)%1) to get the index of the key.

- start(): This method acts as HashWithMult's main() method. It prints the header and runs the program.

HashWithMult has the same methods as HashWithDiv except the ones listed above

Self-Testing screenshots


Problems
Javadoc
Declaration
Console
<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)
****
WELCOME TO MICRO-FACEBOOK!
****
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
6
Hashtable
0: David, Sebastian, Jackson, Madison, Aiden, Ethan, Evelyn, Mason, Benjamin, Logan,
1: Riley, Penelope, Chloe, Grace, Henry, Avery, Emily, Charlotte,
2: null
3: null
4: Carter, Joseph, Elizabeth, Alexander, Harper, Oliver, Elijah, Noah,
5: Lucas, James,
6: Wyatt, Scarlett,
7: Camilia, Aria, Victoria, Ella, Sofia, Amelia, Mia, Sophia, Isabella, Ava, Olivia, Emma,
8: Samuel, Daniel, Michael, Abigail, Jacob,
9: Matthew, William, Liam,
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
1
Create a person
Please enter the person's name: Aniqua
What would you like to do?
1. Create a person
2. Make a friend

Problems Javadoc Declaration Console

<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)

```
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
1
Create a person
Please enter the person's name: Mike

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
2
Make a friend
Enter the name of person 1:     Aniqua
Enter the name of person 2:     Mike
Successfully Friended!

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
1
Create a person
Please enter the person's name: Rob

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
```

Problems @ Javadoc Declaration Console

<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)

```
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
1
Create a person
Please enter the person's name: Kim

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
1
Create a person
Please enter the person's name: Bart

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
1
Create a person
Please enter the person's name: Homer

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
```

Problems @ Javadoc Declaration Console

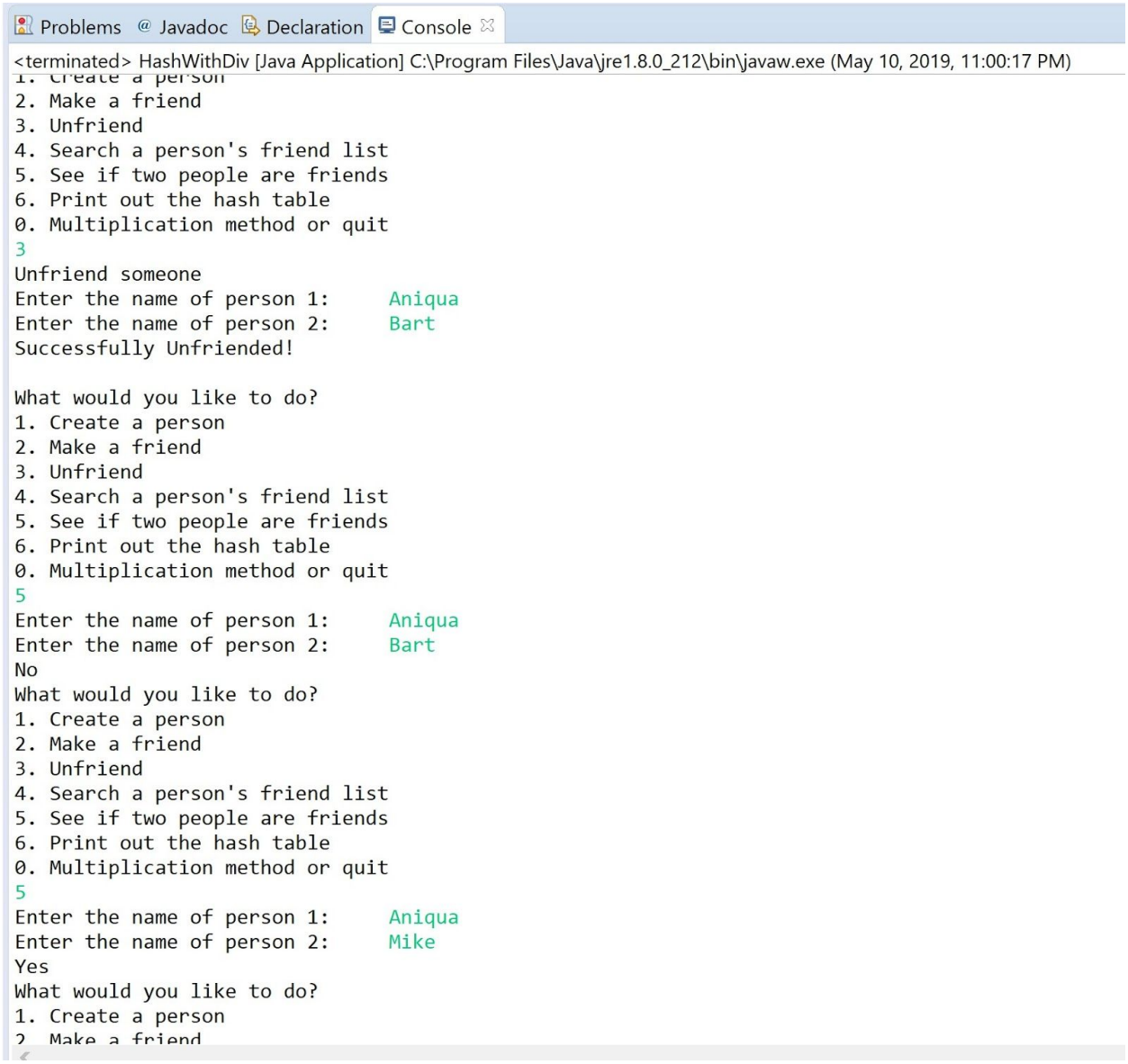
<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)

```
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
2
Make a friend
Enter the name of person 1:     Aniqua
Enter the name of person 2:     Bart
Successfully Friended!

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
2
Make a friend
Enter the name of person 1:     Aniqua
Enter the name of person 2:     Avery
Successfully Friended!

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
2
Make a friend
Enter the name of person 1:     Aniqua
Enter the name of person 2:     Harper
Successfully Friended!

What would you like to do?
1. Create a person
```

Problems Javadoc Declaration Console

<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)

```
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
2
Make a friend
Enter the name of person 1:     Aniqua
Enter the name of person 2:     Emily
Successfully Friended!

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
4
Search a person's friend list
Enter the name: Aniqua
Aniqua's Friend List:   Emily, Harper, Avery, Bart, Mike,
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
4
Search a person's friend list
Enter the name: Emily
Emily's Friend List:    Aniqua, Sophia, Logan, Isabella, James, Ava, William, Olivia, Noah, Emma, Liam,
What would you like to do?
1. Create a person
```


Problems
Javadoc
Declaration
Console
<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
3
Unfriend someone
Enter the name of person 1: Aniqua
Enter the name of person 2: Bart
Successfully Unfriended!
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
5
Enter the name of person 1: Aniqua
Enter the name of person 2: Bart
No
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
5
Enter the name of person 1: Aniqua
Enter the name of person 2: Mike
Yes
What would you like to do?
1. Create a person
2. Make a friend

Problems Javadoc Declaration Console

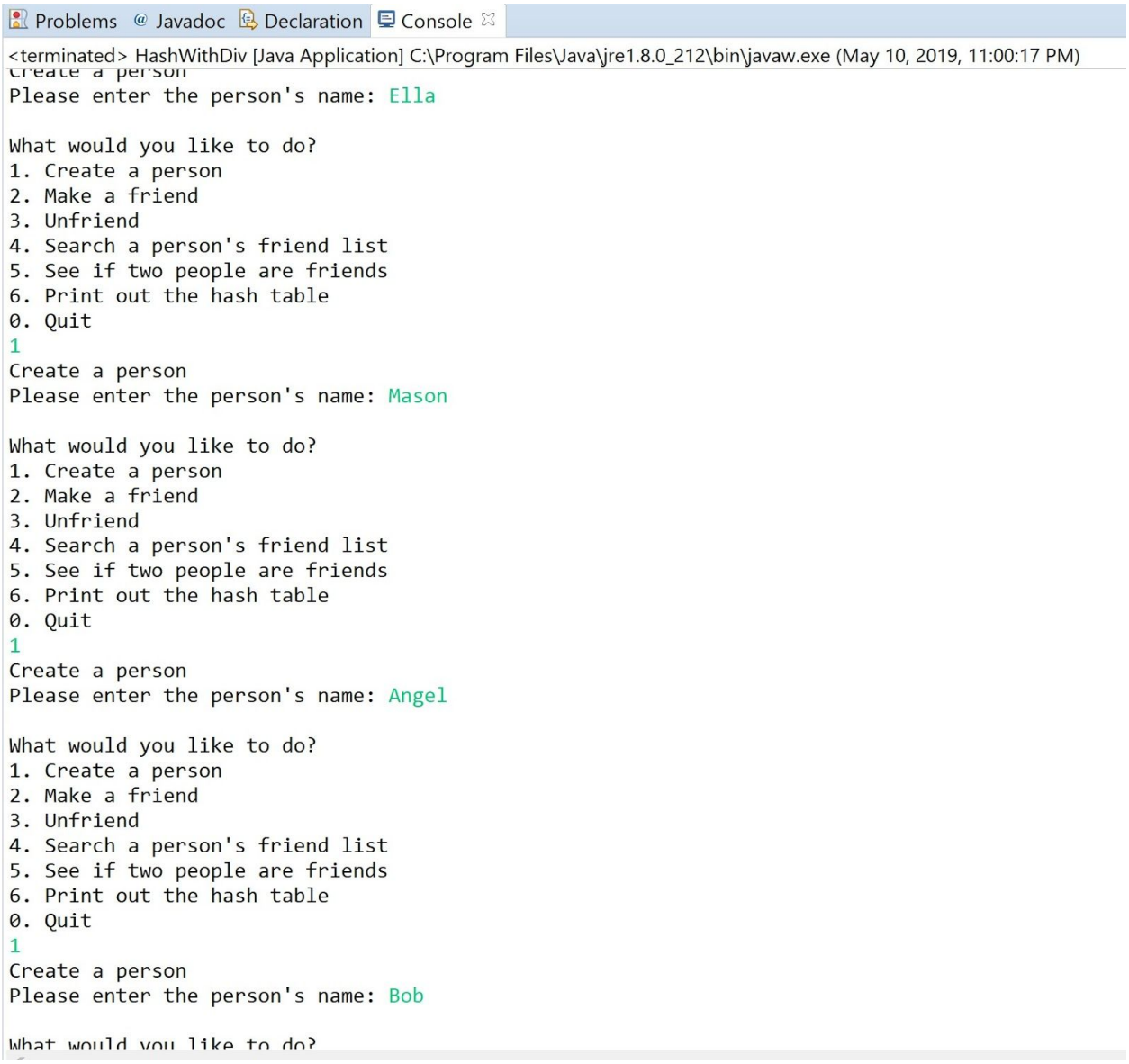
<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)

```
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
4
Search a person's friend list
Enter the name: Aniqua
Aniqua's Friend List:   Emily, Harper, Avery, Mike,
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
6
Hashtable
0:      David, Sebastian, Jackson, Madison, Aiden, Ethan, Evelyn, Mason, Benjamin, Logan,
1:      Mike, Riley, Penelope, Chloe, Grace, Henry, Avery, Emily, Charlotte,
2:      null
3:      null
4:      Homer, Carter, Joseph, Elizabeth, Alexander, Harper, Oliver, Elijah, Noah,
5:      Lucas, James,
6:      Bart, Wyatt, Scarlett,
7:      Aniqua, Camilia, Aria, Victoria, Ella, Sofia, Amelia, Mia, Sophia, Isabella, Ava, Olivia, Emma,
8:      Rob, Samuel, Daniel, Michael, Abigail, Jacob,
9:      Kim, Matthew, William, Liam,
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Multiplication method or quit
0
```

Problems Javadoc Declaration Console

<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)

```
6. Print out the hash table
0. Multiplication method or quit
0
Would you like to try the multiplication method?
Press 8 for yes, 9 for no
8
****************************************
WELCOME TO THE MULT METHOD FB
****************************************
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
6
Hashtable
0:      Lucas, James,
1:      null
2:      Joseph, Elizabeth, Elijah, Noah,
3:      William, Liam,
4:      Penelope, Carter, Chloe, Grace, Alexander, Harper, Oliver, Charlotte,
5:      Matthew, Jacob,
6:      Wyatt, Scarlett,
7:      Riley, Samuel, Henry, Daniel, Avery, Emily, Michael, Abigail,
8:      David,
9:      Camilia, Sebastian, Aria, Jackson, Victoria, Madison, Aiden, Ella, Sofia, Ethan, Evelyn, Amelia, Mason, Mia, Benjamin, Sophia, Logan, Isabella, Ava, Olivia, Emma,
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
1
Create a person
Please enter the person's name: Ella
```


Problems
Javadoc
Declaration
Console
<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)
Create a person
Please enter the person's name: Ella
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
1
Create a person
Please enter the person's name: Mason
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
1
Create a person
Please enter the person's name: Angel
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
1
Create a person
Please enter the person's name: Bob
What would you like to do?

Problems Javadoc Declaration Console

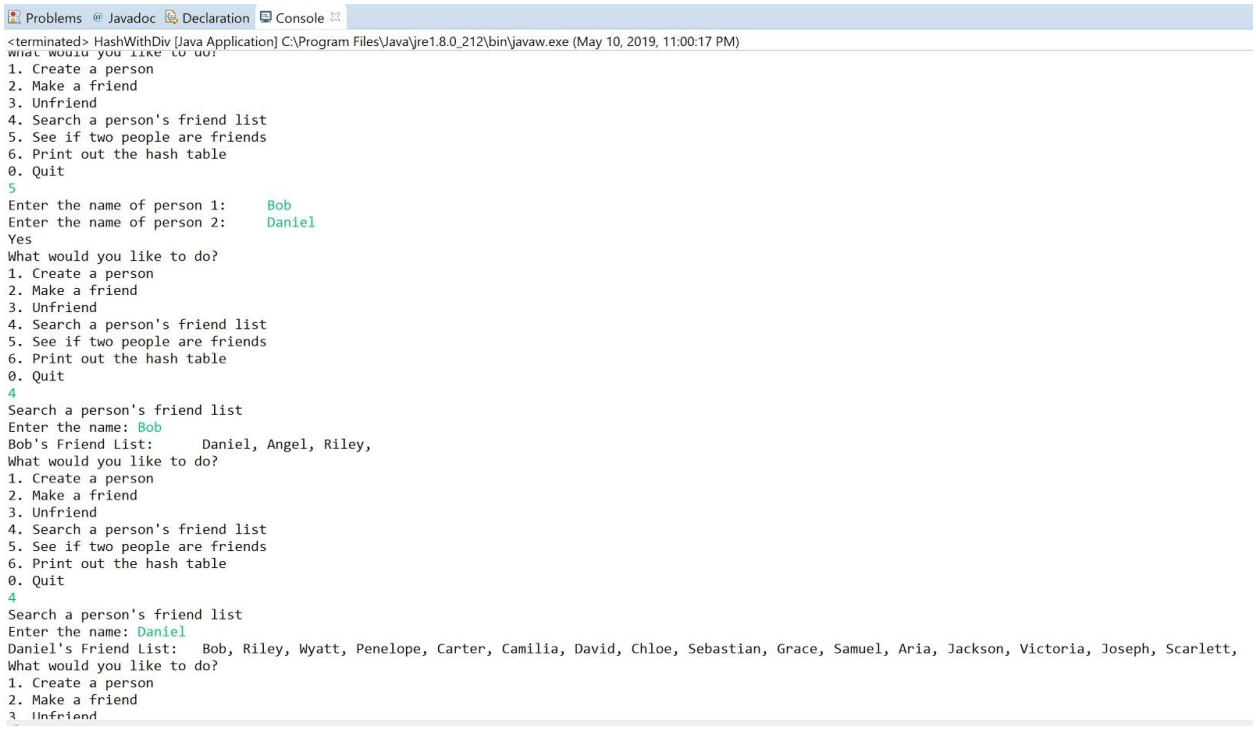
<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)

```
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
1
Create a person
Please enter the person's name: May

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
1
Create a person
Please enter the person's name: Tiff

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
6
Hashtable
0:      Tiff, Lucas, James,
1:      null
2:      Joseph, Elizabeth, Elijah, Noah,
3:      William, Liam,
4:      Penelope, Carter, Chloe, Grace, Alexander, Harper, Oliver, Charlotte,
```

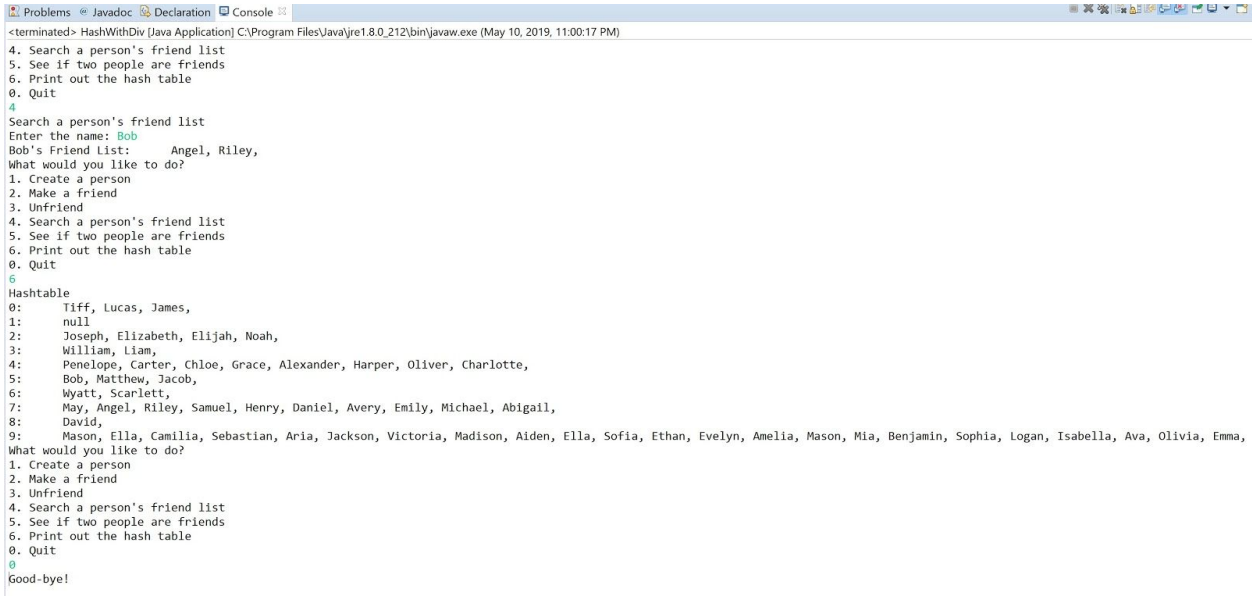
Problems Javadoc Declaration Console

<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)

```
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
2
Make a friend
Enter the name of person 1:     Wyatt
Enter the name of person 2:     Angel
Successfully Friended!

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
2
Make a friend
Enter the name of person 1:     Bob
Enter the name of person 2:     Angel
Successfully Friended!

What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
2
Make a friend
Enter the name of person 1:     Bob
Enter the name of person 2:     Daniel
Successfully Friended!

What would you like to do?
```



```
Problems  Javadoc  Declaration  Console
<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
5
Enter the name of person 1:     Bob
Enter the name of person 2:     Daniel
Yes
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
4
Search a person's friend list
Enter the name: Bob
Bob's Friend List:      Daniel, Angel, Riley,
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
4
Search a person's friend list
Enter the name: Daniel
Daniel's Friend List:   Bob, Riley, Wyatt, Penelope, Carter, Camilia, David, Chloe, Sebastian, Grace, Samuel, Aria, Jackson, Victoria, Joseph, Scarlett,
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
```



```
Problems  Javadoc  Declaration  Console
<terminated> HashWithDiv [Java Application] C:\Program Files\Java\jre1.8.0_212\bin\javaw.exe (May 10, 2019, 11:00:17 PM)
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
4
Search a person's friend list
Enter the name: Bob
Bob's Friend List:      Angel, Riley,
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
6
Hashtable
0:      Tiff, Lucas, James,
1:      null
2:      Joseph, Elizabeth, Elijah, Noah,
3:      William, Liam,
4:      Penelope, Carter, Chloe, Grace, Alexander, Harper, Oliver, Charlotte,
5:      Bob, Matthew, Jacob,
6:      Wyatt, Scarlett,
7:      May, Angel, Riley, Samuel, Henry, Daniel, Avery, Emily, Michael, Abigail,
8:      David,
9:      Mason, Ella, Camilia, Sebastian, Aria, Jackson, Victoria, Madison, Aiden, Ella, Sofia, Ethan, Evelyn, Amelia, Mason, Mia, Benjamin, Sophia, Logan, Isabella, Ava, Olivia, Emma,
What would you like to do?
1. Create a person
2. Make a friend
3. Unfriend
4. Search a person's friend list
5. See if two people are friends
6. Print out the hash table
0. Quit
0
Good-bye!
```

Procedure

a. Extract files by unzipping folder

b. Double-click folder titled **PA3-Section 7-Azad**

c. To access *.java files:

   i. Unzip the folder titled “src.zip”

ii. double-click folder titled “src”(you will have to do this twice), then double-click the folder titled “pa3”

iii. Copy/paste the code into an IDE

d. To access formal report, double-click PDF titled “AzadPA3_Formal Report.pdf”

e. In order to run code, click on the AzadPA3.jar

f. If that doesn’t work, open command prompt/terminal and change directory to where AzadPA3.jar is stored

i. type: java -jar AzadPA3.jar

ii. When it runs, you should be able to see the output on the command prompt.

Problem(s) Encountered

- While programming, I was confused as to what the difference between inserting to the hash table and the friend list was. I originally thought I would have to use chainedHashInsert to insert friends into a Person’s friend linked list. I found it confusing beacsue I thought that was only reserved for the hash table. I asked the professor for clarification. I was able to differentiate the difference and I created methods in my Person class which used the LinkedList class to insert friends into the friends LinkedList. I also learned that chainedHashInsert was only for inserting to the hash table.
- I also ran into problems when using chainedHashSearch. I didn’t know what it was supposed to return, in addition to the search method in my LinkedList class. I originally made the chainedHashSearch method return a boolean and then I would create a new Person with the name that the user typed. This created problems as when I tried adding two people as friends, it would say success, but when I checked if those two people were friends it said they weren’t. To fix it, I set breakpoints in various lines of my code and I realized that because I was creating more Persons, it wasn’t using the ones that were already created. So, I made my chainedHashSearch and search methods return Person, which finally made my code work.
- Additionally, I was confused as to how I would add the 50 people to the hash table. I originally thought that I would have to manually add all 50 people, which really worried

me. When the professor said that they should already by added when the program runs, I thought about creating a HashTable class and having a global static hash table; but then, I didn't know what hash function to use. I emailed the professor and he said that I should have twop hash functions and two hash tables. So, I made a method which adds all 50 people and a methods which adds friends to some of the 50 people.

Lesson(s) Learned

- I learned that chainedHashInsert is only used for adding Persons to the hash table. I thought it would also be used for adding friends to the linked list, which made the assignment more confusing for me. However, after talking to the professor, it was clarified and I was finally able to successfully start coding.
- It wasn't really a lesson learned, but more like a lesson refreshed. I worked with creating my own LinkedLists a while back and last semester in CS46B, but I had forgotten how to use the different methods and nodes. This project allowed me to refresh my memory in how to make and use my own LinkedList and nodes; however, I did have to watch a few videos and go through past lectures in order to get my code to work.
- I really enjoyed coding this assignment even though I ran into some roadblocks before. However, once everything was clarified, it did not take too long to get my code up and running successfully. I realized that I was overthinking many parts of the requirements which added to my confusion. The run time of my code seems pretty fast, which probably means that I implemented the algorithms correctly. However, when I have the time, I would like to go back and make my code more efficient and manageable.