|  |
| --- |
| Troy Thomas  U08223: Data Structures  Coursework  Report on Data Structure Coursework |

|  |  |
| --- | --- |
| Part 1 | Hash Table |

|  |  |
| --- | --- |
| A  (FR1) | *“You must create a Java class Supporter that implements the interface ISupporter. You will also need to implement a constructor or constructors for this class.*  *Paste the text of your class into your report.”* |

|  |  |
| --- | --- |
| **Supporter.Java** |  |
| /\*  \* To change this license header, choose License Headers in Project Properties.  \* To change this template file, choose Tools | Templates  \* and open the template in the editor.  \*/  package data**.**structures**.**coursework**.**part**.**pkg1**;**  /\*\*  \*  \* **@author** 17075763  \*/  public class Supporter **implements** ISupporter **{**    private String name**;**  private String ID**;**    public Supporter**(**String name**,** String ID**)** **{**    **this.**name **=** name**;**  **this.**ID **=** ID**;**  **}**    @Override  public String getName**()** **{**  **return** name**;**  **}**    @Override  public String getID**()** **{**  **return** ID**;**  **}**  **}** | |

|  |  |
| --- | --- |
| B  (NFR3) | *“You will need to perform some experiments to find a good hash function. Include an explanation of your hash function and your experiments in your report.”* |

In order to tackle the where to insert information into my Hash Table I used a method called **getHashValue**(). It looks like this:

|  |  |
| --- | --- |
| **getHashValue()** |  |
| private int getHashValue**(**String surname**)** **{**    int totalValue **=** 0**;** //each round of the following loop deposits the current characters vaule here  int hash**;** //this will be the final value, the hashed surname  char**[]** surnameArray **=** surname**.**toCharArray**();** //convert surname into an array that holds each character of surname  **for** **(**char character **:** surnameArray**)** **{** //loops through the whole array of characters  totalValue **=** totalValue **+** character**;** //adds the value of each character to a variable  **}**  hash **=** totalValue **%** hashCapacity**;** //hashing happens here  //The Quadratic Probing happens here  **while** **(this.**hashTable**[**hash**]** **!=** **null** **&&** **!(this.**hashTable**[**hash**].**getName**().**equals**(**surname**)))** **{** //while in an empty space in the has table and while the name at 'hash'th position in the hash table is not eqaul to surname  hash **=** hash **+** 1**;** //so that the posistion can still be sqaured if the hash is 0.  hash **=** hash **\*** hash **%** hashCapacity**;** //mutipling the hash value by itself for the quadratic part of Quadratic Probing, the modulus is so the resulting value is kept within the bounds of the hash table.  **}**  **return** hash**;**  **}** | |

I tried to explain every step of the hash function in the comments. Of note is that I used **Quadratic Probing** as the collision resolution method.

|  |  |
| --- | --- |
| B  (NFR7) | *“You must make your class log monitoring information, either to a text file or by calls of System.out.println.”* |

I used the System.out.println to log monitoring information. The log follows:

|  |  |
| --- | --- |
| **Output** |  |
| run:  Size of hashTable: 1  Load factor is:0.5 |(items: 1 |hashCapacity: 2)  Hash Table Print:  0: TROY 1  1: null    Get called! Supporter Name: TROY | Supporter ID: 1 | Hash Value: 0    1 TROY added to table / tree.    Size of hashTable: 2  Load factor is:1.0 |(items: 2 |hashCapacity: 2)  Hash Table capacity increased!  Old capacity: 2| New capacity: 4  Load factor is:0.5 |(items: 2 |hashCapacity: 4)  Hash Table Print:  0: THOMAS 2  1: null  2: TROY 1  3: null    Get called! Supporter Name: THOMAS | Supporter ID: 2 | Hash Value: 0    THOMAS 2 added to table / tree.    Size of hashTable: 3  Load factor is:0.75 |(items: 3 |hashCapacity: 4)  Hash Table capacity increased!  Old capacity: 4| New capacity: 8  Load factor is:0.375 |(items: 3 |hashCapacity: 8)  Hash Table Print:  0: null  1: null  2: null  3: BOB 3  4: THOMAS 2  5: null  6: TROY 1  7: null    Get called! Supporter Name: BOB | Supporter ID: 3 | Hash Value: 3    BOB 3 added to table / tree.    Size of hashTable: 4  Load factor is:0.5 |(items: 4 |hashCapacity: 8)  Hash Table Print:  0: SUPRISE 62  1: null  2: null  3: BOB 3  4: THOMAS 2  5: null  6: TROY 1  7: null    Get called! Supporter Name: SUPRISE | Supporter ID: 62 | Hash Value: 0    SUPRISE 62 added to table / tree.    Size of hashTable: 5  Load factor is:0.625 |(items: 5 |hashCapacity: 8)  Hash Table Print:  0: SUPRISE 62  1: JIM 4  2: null  3: BOB 3  4: THOMAS 2  5: null  6: TROY 1  7: null    Get called! Supporter Name: JIM | Supporter ID: 4 | Hash Value: 1    JIM 4 added to table / tree.    Size of hashTable: 6  Load factor is:0.75 |(items: 6 |hashCapacity: 8)  Hash Table capacity increased!  Old capacity: 8| New capacity: 16  Load factor is:0.375 |(items: 6 |hashCapacity: 16)  Hash Table Print:  0: JIM 4  1: null  2: PAUL 5  3: BOB 3  4: null  5: null  6: null  7: null  8: null  9: null  10: null  11: SUPRISE 62  12: THOMAS 2  13: null  14: TROY 1  15: null    Get called! Supporter Name: PAUL | Supporter ID: 5 | Hash Value: 2    PAUL 5 added to table / tree.    Size of hashTable: 7  Load factor is:0.4375 |(items: 7 |hashCapacity: 16)  Hash Table Print:  0: JIM 4  1: null  2: PAUL 5  3: BOB 3  4: null  5: null  6: null  7: null  8: null  9: ZZZZZ 6  10: null  11: SUPRISE 62  12: THOMAS 2  13: null  14: TROY 1  15: null    Get called! Supporter Name: ZZZZZ | Supporter ID: 6 | Hash Value: 9    ZZZZZ 6 added to table / tree.    Size of hashTable: 8  Load factor is:0.5 |(items: 8 |hashCapacity: 16)  Hash Table Print:  0: JIM 4  1: null  2: PAUL 5  3: BOB 3  4: ZZZZZZ 7  5: null  6: null  7: null  8: null  9: ZZZZZ 6  10: null  11: SUPRISE 62  12: THOMAS 2  13: null  14: TROY 1  15: null    Get called! Supporter Name: ZZZZZZ | Supporter ID: 7 | Hash Value: 4    ZZZZZZ 7 added to table / tree.    Size of hashTable: 9  Load factor is:0.5625 |(items: 9 |hashCapacity: 16)  Hash Table Print:  0: JIM 4  1: null  2: PAUL 5  3: BOB 3  4: ZZZZZZ 7  5: null  6: ZZZZZZZ 8  7: null  8: null  9: ZZZZZ 6  10: null  11: SUPRISE 62  12: THOMAS 2  13: null  14: TROY 1  15: null    Get called! Supporter Name: ZZZZZZZ | Supporter ID: 8 | Hash Value: 6    ZZZZZZZ 8 added to table / tree.      --------  containsName() test  Is TROY (root of tree) contained in the database: true  Is THOMAS (leaf of tree) contained in the database: true  Is HARRY (Not in the database) contained in the database: false    --------  isEmpty() test: (shouldn't be empty)  Is the hash table / binary search tree empty: false  --------  clear() test  clear() called!    Hash Table Print:  0: null  1: null  2: null  3: null  4: null  5: null  6: null  7: null  8: null  9: null  10: null  11: null  12: null  13: null  14: null  15: null      Is the hash table / binary search tree empty: true  BUILD SUCCESSFUL (total time: 0 seconds) | |

|  |
| --- |
| *“Paste the text of your class into your report.”* |

Not sure if this part means just the main or both the main and SupporterHT, so I’ve pasted both:

|  |  |
| --- | --- |
| **DataStucturesCourseworkPart1.java** |  |
| /\*  \* To change this license header, choose License Headers in Project Properties.  \* To change this template file, choose Tools | Templates  \* and open the template in the editor.  \*/  package data**.**structures**.**coursework**.**part**.**pkg1**;**  /\*\*  \*  \* **@author** 17075763  \*/  public class DataStructuresCourseworkPart1 **{**  /\*\*  \* **@param** args the command line arguments  \*/  public static void main**(**String**[]** args**)** **{**        String ID **=** "1"**;**  String name **=** "TROY"**;**    Supporter supporter **=** **new** Supporter**(**name**,** ID**);**    //FOR HASHTABLES UNCOMMENT "HT" AND COMMENT "BST"  //ISupporterDatabase db = new SupporterDatabaseBST(supporter);  ISupporterDatabase db **=** **new** SupporterDatabaseHT**(**2**);**    db**.**put**(**supporter**);**  ISupporter test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getID**()** **+** " " **+** test**.**getName**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "2"**;**  name **=** "THOMAS"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "3"**;**  name **=** "BOB"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    //test = db.remove(name);    ID **=** "62"**;**  name **=** "SUPRISE"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "4"**;**  name **=** "JIM"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "5"**;**  name **=** "PAUL"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "6"**;**  name **=** "ZZZZZ"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "7"**;**  name **=** "ZZZZZZ"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "8"**;**  name **=** "ZZZZZZZ"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**"--------"**);**  System**.**out**.**println**(**"containsName() test"**);**  System**.**out**.**println**(**"Is TROY (root of tree) contained in the database: " **+** db**.**containsName**(**"TROY"**));**  System**.**out**.**println**(**"Is THOMAS (leaf of tree) contained in the database: " **+** db**.**containsName**(**"THOMAS"**));**  System**.**out**.**println**(**"Is HARRY (Not in the database) contained in the database: " **+** db**.**containsName**(**"HARRY"**));**  System**.**out**.**println**(**" "**);**    System**.**out**.**println**(**"--------"**);**  System**.**out**.**println**(**"isEmpty() test: (shouldn't be empty)"**);**  System**.**out**.**println**(**"Is the hash table / binary search tree empty: " **+** db**.**isEmpty**());**    System**.**out**.**println**(**"--------"**);**  System**.**out**.**println**(**"clear() test"**);**  db**.**clear**();**  System**.**out**.**println**(**"clear() called!"**);**    System**.**out**.**println**(**" "**);**  db**.**printSupportersOrdered**();**    System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**"Is the hash table / binary search tree empty: " **+** db**.**isEmpty**());**    **}**    **}** | |

|  |  |
| --- | --- |
| **SupporterDatabaseHT.java** |  |
| /\*  \* To change this license header, choose License Headers in Project Properties.  \* To change this template file, choose Tools | Templates  \* and open the template in the editor.  \*/  package data**.**structures**.**coursework**.**part**.**pkg1**;**  /\*\*  \*  \* **@author** 17075763  \*/  public class SupporterDatabaseHT **implements** ISupporterDatabase **{**  private int itemsInTable**;**  private int hashCapacity**;**  Supporter**[]** hashTable**;**  private static final double LOAD\_FACTOR **=** 0.75**;**  public SupporterDatabaseHT**(**int hashCapacity**)** **{**  **this.**itemsInTable **=** 0**;**  **this.**hashCapacity **=** hashCapacity**;**  hashTable **=** **new** Supporter**[this.**hashCapacity**];**  **}**    private int getHashValue**(**String surname**)** **{**    int totalValue **=** 0**;** //each round of the following loop deposits the current characters vaule here  int hash**;** //this will be the final value, the hashed surname  char**[]** surnameArray **=** surname**.**toCharArray**();** //convert surname into an array that holds each character of surname  **for** **(**char character **:** surnameArray**)** **{** //loops through the whole array of characters  totalValue **=** totalValue **+** character**;** //adds the value of each character to a variable  **}**  hash **=** totalValue **%** hashCapacity**;** //hashing happens here  //The Quadratic Probing happens here  **while** **(this.**hashTable**[**hash**]** **!=** **null** **&&** **!(this.**hashTable**[**hash**].**getName**().**equals**(**surname**)))** **{** //while in an empty space in the has table and while the name at 'hash'th position in the hash table is not eqaul to surname  hash **=** hash **+** 1**;** //so that the posistion can still be sqaured if the hash is 0.  hash **=** hash **\*** hash **%** hashCapacity**;** //mutipling the hash value by itself for the quadratic part of Quadratic Probing, the modulus is so the resulting value is kept within the bounds of the hash table.  **}**  **return** hash**;**  **}**    public void loadFactor**()** **{**  double items **=** itemsInTable**;**  double hashCap **=** hashCapacity**;**  double load **=** items **/** hashCap**;**  System**.**out**.**println**(**"Load factor is:" **+** load **+** " |(items: " **+** itemsInTable **+** " |hashCapacity: " **+** hashCapacity **+** ")"**);**  **if** **(**load **>=** LOAD\_FACTOR**)** **{**  //printSupportersOrdered();  increaseCapacity**();**  loadFactor**();**  **}**  **}**    public void increaseCapacity**()** **{**  int oldCapacity **=** hashCapacity**;**  hashCapacity **=** hashCapacity **\*** 2**;**  Supporter**[]** oldHashTable**;**    String name**;**  String ID**;**  itemsInTable **=** 0**;**    oldHashTable **=** hashTable**;**  hashTable **=** **new** Supporter**[**hashCapacity**];**    **for** **(**int i **=** 0**;** i **<** oldCapacity**;** i**++)** **{**  **if(**oldHashTable**[**i**]** **!=** **null){**  itemsInTable **=** itemsInTable **+** 1**;**  name **=** oldHashTable**[**i**].**getName**();**  ID **=** oldHashTable**[**i**].**getID**();**  Supporter supporter **=** **new** Supporter**(**name**,** ID**);**    hashTable**[**getHashValue**(**name**)]** **=** supporter**;**  **}**  **}**    System**.**out**.**println**(**"Hash Table capacity increased!"**);**  System**.**out**.**println**(**"Old capacity: " **+** oldCapacity **+** "| New capacity: " **+** hashCapacity**);**  **}**    /\*\*  \* Empties the database.  \*  \* **@pre** true  \*  \* TRY: Get the size of the hash table, save it, then set the hash table  \* size to 0, then set the size back to the old size. ALT: Look for one of  \* those "." (e.g: .length) that does this.  \*/  @Override  public void clear**()** **{**  hashTable **=** **new** Supporter**[this.**hashCapacity**];**  itemsInTable **=** 0**;**  **}**  /\*\*  \* Determines whether a Supporter name exists as a key inside the database  \*  \* **@pre** true  \* **@param** name the Supporter name (key) to locate  \* **@return** true iff the name exists as a key in the database  \*  \* This is to check through the associative array to check for whether  \* THOMAS 72 is in the hash table or not.  \*/  @Override  public boolean containsName**(**String name**)** **{**  boolean contains**;**    **if** **(**hashTable**[**getHashValue**(**name**)]!=** **null)** **{**  contains **=** **true;**  **}** **else** **{**  contains **=** **false;**  **}**  **return** contains**;**  **}**  /\*\*  \* Returns a Supporter object mapped to the supplied name.  \*  \* **@pre** true  \* **@param** name The Supporter name (key) to locate  \* **@return** the Supporter object mapped to the key name if the name exists as  \* key in the database, otherwise null  \*/  @Override  public Supporter get**(**String name**)** **{**  System**.**out**.**println**(**"Get called! Supporter Name: " **+** name **+** " | Supporter ID: " **+** hashTable**[**getHashValue**(**name**)].**getID**()** **+** " | Hash Value: " **+** getHashValue**(**name**));**  **return** hashTable**[**getHashValue**(**name**)];**  **}**  /\*\*  \* Returns the number of supporters in the database  \*  \* **@pre** true  \* **@return** number of supporters in the database. 0 if empty  \*/  @Override  public int size**()** **{**  **return** itemsInTable**;**  **}**  /\*\*  \* Determines if the database is empty or not.  \*  \* **@pre** true  \* **@return** true if the database is empty  \*/  @Override  public boolean isEmpty**()** **{**  **return** itemsInTable **==** 0**;**  **}**  /\*\*  \* Inserts a supporter object into the database, with the key of the  \* supplied supporter's name. Note: If the name already exists as a key,  \* then then the original entry is overwritten. This method should return  \* the previous associated value if one exists, otherwise null  \*  \* **@pre** true  \* **@param** supporter the supporter object being inserted into the database  \* **@return** "The previous associated value" (whatever that means) or null  \* EDIT: Maybe means the value that was previously in that position in the  \* hash table? Return the old key? Return the whole old supporter??  \*/  @Override  public Supporter put**(**Supporter supporter**)** **{**  //increment items in table if adding to a null space, not if replacing already existing item  //chk load val then change size of table (or not) accordingly  int hash **=** getHashValue**(**supporter**.**getName**());**    Supporter oldValue **=** hashTable**[**hash**];**  hashTable**[**hash**]** **=** supporter**;**    **if(**hashTable**[**hash**]** **!=** **null){**  itemsInTable **=** itemsInTable **+** 1**;**  **}**  System**.**out**.**println**(**"Size of hashTable: " **+** size**());**  loadFactor**();**  printSupportersOrdered**();**    **return** oldValue**;**  **}**  /\*\*  \* Removes and returns a supporter from the database, with the key the  \* supplied name.  \*  \* **@param** name The name (key) to remove.  \* **@pre** true  \* **@return** the removed supporter object mapped to the name, or null if the  \* name does not exist.  \*/  @Override  public Supporter remove**(**String name**)** **{**  int hash **=** getHashValue**(**name**);**  Supporter removedObject **=** **null;**  Supporter blankObject **=** **new** Supporter**(**" "**,**" "**);**    **if** **(**hashTable**[**hash**]!=** **null)** **{**  removedObject **=** hashTable**[**hash**];**  hashTable**[**hash**]** **=** blankObject**;**  **}**    System**.**out**.**println**(**"Removed: " **+** removedObject**.**getName**()** **+** " " **+** removedObject**.**getID**())** **;**  **return** removedObject**;**  **}**  /\*\*  \* Prints the names and IDs of all the supporters in the database in  \* alphabetic order.  \*  \* **@pre** true  \*/  @Override  public void printSupportersOrdered**()** **{** //this isnt ordered  System**.**out**.**println**(**"Hash Table Print:"**);**  **for** **(**int i **=** 0**;** i **<** hashCapacity**;** i**++)** **{**  **if** **(**hashTable**[**i**]** **!=** **null)** **{**  System**.**out**.**println**(**i **+** ": " **+** hashTable**[**i**].**getName**()** **+** " " **+** hashTable**[**i**].**getID**());**  **}** **else** **{**  System**.**out**.**println**(**i **+** ": " **+** hashTable**[**i**]);**  **}**  **}**  System**.**out**.**println**(**" "**);**  **}**      **}** | |

|  |  |
| --- | --- |
| C | *“Include your test plan, test data used, expected results and actual results in your report.”* |

|  |  |  |
| --- | --- | --- |
| Testing | | |
| Methods | | |
| Test Data | Expected Results | Actual Output  Green: Pass  Red: Fail |
| **getHashValue**() | | |
| surname: Troy  hashCapacity: 2 | 0 | 0: Troy 1 |
| surname: Thomas  hashCapacity: 16 | 12 | 12: Thomas 2 |
| surname: Bob  hashCapacity:8 | 3 | 3: Bob 3 |
| **loadFactor**() | | |
| itemsInTable: 1  hashCapacity: 2 | 0.5 | Load factor is:0.5 |(items: 1 |hashCapacity: 2) |
| itemsInTable: 6  hashCapacity:8 | 0.75 | Load factor is:0.75 |(items: 6 |hashCapacity: 8) |
| itemsInTable: 4  hashCapacity:8 | 0.5 | Load factor is:0.5 |(items: 4 |hashCapacity: 8) |

|  |  |  |
| --- | --- | --- |
| **increaseCapacity**() | | |
| LoadFactor: 1.0  Capacity: 2  Table Before:  Hash Table Print:  0: Troy 1  1: Thomas 2 | LoadFactor: 0.5  Capacity: 4 | Load factor is:0.5 |(items: 2 |hashCapacity: 4)  Hash Table Print:  0: Thomas 2  1: null  2: Troy 1  3: null |
| **clear**() | | |
| Hash Table Print:  0: Jim 4  1:  2: Paul 5  3: null  4: null  5: null  6: null  7: null  8: null  9: null  10: null  11: Suprise 62  12: Thomas 2  13: null  14: Troy 1  15: null | All spaces in the array to be null | Hash Table Print:  0: null  1: null  2: null  3: null  4: null  5: null  6: null  7: null  8: null  9: null  10: null  11: null  12: null  13: null  14: null  15: null |
| **containsName**() | | |
| Looking for word:  Troy  Hash Table:  Hash Table Print:  0: Jim 4  1:  2: Paul 5  3: null  4: null  5: null  6: null  7: null  8: null  9: null  10: null  11: Suprise 62  12: Thomas 2  13: null  14: Troy 1  15: null | True | Is Troy contained in the database: true |

|  |  |  |
| --- | --- | --- |
| Looking for word:  Haberdasher  Hash Table:  Hash Table Print:  0: Jim 4  1:  2: Paul 5  3: null  4: null  5: null  6: null  7: null  8: null  9: null  10: null  11: Suprise 62  12: Thomas 2  13: null  14: Troy 1  15: null | False | Is Haberdasher contained in the database: false |
| **get**() | | |
| Called on:  Paul | 2 | Get called! Supporter Name: Paul| Hash Value: 2 |
| **size**() | | |
| Hashtable:  Hash Table Print:  0: Thomas 2  1: null  2: Troy 1  3: Bob 3 | 3 | Size of hashTable: 3 |
| **isEmpty**() | | |
| Hash Table:  0: Jim 4  1:  2: Paul 5  3: null  4: null  5: null  6: null  7: null  8: null  9: null  10: null  11: Suprise 62  12: Thomas 2  13: null  14: Troy 1  15: null | False | Is the hash table empty: false |

|  |  |  |
| --- | --- | --- |
| Hash Table:  0: null  1: null  2: null  3: null  4: null  5: null  6: null  7: null  8: null  9: null  10: null  11: null  12: null  13: null  14: null  15: null | True | Is the hash table empty: true |
| **put**() | | |
| Name being Inserted: Jim  ID: 4  Hash: 1  Hash Table:  0: null  1: null  2: null  3:  4: Thomas 2  5: null  6: Troy 1  7: null | Jim Is in the Table | Hash Table Print:  0: Suprise 62  1: Jim 4  2: Paul 5  3:  4: Thomas 2  5: null  6: Troy 1  7: null |
| **remove**() | | |
| Name being removed: Bob  Hash Table:  0: null  1: null  2: null  3: Bob 3  4: Thomas 2  5: null  6: Troy 1  7: null | Bob to be gone from the Hash Table | Hash Table Print:  0: null  1: null  2: null  3:  4: Thomas 2  5: null  6: Troy 1  7: null |
| **printSupportersOrdered**() | | |
| Hash Table:  0: null  1: null  2: null  3: Bob 3  4: Thomas 2  5: null  6: Troy 1  7: null | The names to be ordered | Hash Table Print:  0: null  1: null  2: null  3: Bob 3  4: Thomas 2  5: null  6: Troy 1  7: null |

|  |  |
| --- | --- |
| D | *“* *You must state honestly which of the requirements of part 1 you have successfully fulfilled, citing evidence. Also comment on the time efficiency and space efficiency of your implementation of the hash table.”* |

|  |  |
| --- | --- |
| FR1 | Done. See: above or the source files |
| FR2 | Done. See the source files |
| FR3 | Done. See: above or the source files |
| FR4 | Partially Done. See: above or source files. I couldn’t figure out how to include the blank spaces (made by my **remove**() method) in the if statement that decides what to remove from the array, so the blank spaces are not removed upon creating a new array. |
| NFR1 | As far as I can tell, it does. |
| NFR2 | Done. See the source files |
| NFR3 | I created a hash function, no experiments preformed with multiple types (only ones to show that my function works). |
| NFR4 | Built in hashCode method was not used. |
| NFR5 | Partially Done. See: above or source files. I didn’t try to avoid non-termination. |
| NFR6 | Done. See: above or source files. |
| NFR7 | Partially Done. Everything listed should be in this document, excluding “the sequence of buckets (array locations) visited” bullet point, which I just wasn’t too sure of. |

|  |  |
| --- | --- |
| Part 2 | Binary Search Tree |

|  |  |
| --- | --- |
| E  (FR6) | *“* *You must make your class log monitoring information, either to a text file or by calls of System.out.println…*  *…Paste the text of your class into your report.”* |

|  |  |
| --- | --- |
| **SupporterDatabaseBST.java** |  |
| /\*  \* To change this license header, choose License Headers in Project Properties.  \* To change this template file, choose Tools | Templates  \* and open the template in the editor.  \*/  package data**.**structures**.**coursework**.**part**.**pkg1**;**  /\*\*  \*  \* **@author** Troy  \*/  class Node **{**  Supporter data**;**  Node left**;**  Node right**;**  public Node**(**Supporter data**)** **{**  **this.**data **=** data**;**  **this.**left **=** **null;**  **this.**right **=** **null;**  **}**  **}**  public class SupporterDatabaseBST **implements** ISupporterDatabase **{**    Node root**;**  static int putCounter **=** 0**;**  public SupporterDatabaseBST**(**Supporter value**)** **{**  **this.**root **=** **null;**  **}** // constructor        public int getValue**(**String name**)** **{**  int totalValue **=** 0**;**  char**[]** surnameArray **=** name**.**toCharArray**();**  **for** **(**char character **:** surnameArray**)** **{**  totalValue **=** totalValue **+** character**;**  **}**  **return** totalValue**;**  **}**    public int getDepth**()** **{**    **return** getDepth**(**root**);**  **}**    private int getDepth**(**Node node**)** **{**  int depthLeft **=** 0**;**  int depthRight **=** 0**;**  int depth**;**    **if** **(**node**.**left **!=** **null)** **{**  depthLeft **=** size**(**node**.**left**);**  **}**  **if** **(**node**.**right **!=** **null)** **{**  depthRight **=** size**(**node**.**right**);**  **}**    **if** **(**depthLeft **>** depthRight**)** **{**  depth **=** depthLeft**;**  **}** **else** **{**  depth **=** depthRight**;**  **}**  **return** depth**;**  **}**  /\*\*  \* Empties the database.  \* **@pre** true  \*/  @Override  public void clear**(){**  root **=** **null;**  **}**        @Override  public boolean containsName**(**String name**)** **{**  **return** containsName**(**name**,** root**);**  **}**  private boolean containsName**(**String name**,** Node node**)** **{**  int value **=** getValue**(**name**);**  boolean found**;**    **if** **(**value **==** getValue**(**node**.**data**.**getName**()))** **{**  found **=** **true;**  **}** **else** **if** **(**value **<** getValue**(**node**.**data**.**getName**()))** **{**  **if** **(**node**.**left **==** **null)** **{**  found **=** **false;**  **}** **else** **{**  found **=** containsName**(**name**,** node**.**left**);**  **}**  **}** **else** **{**  **if** **(**node**.**right **==** **null)** **{**  found **=** **false;**  **}** **else** **{**  found **=** containsName**(**name**,** node**.**right**);**  **}**  **}**    **return** found**;**  **}**  @Override  public Supporter get**(**String name**)** **{**  Supporter got**;**    got **=** get**(**name**,** root**);**    **return** got**;**  **}**    private Supporter get**(**String name**,** Node node**)** **{**  Supporter got**;**    **if** **(**getValue**(**name**)** **==** getValue**(**node**.**data**.**getName**()))** **{**  got **=** node**.**data**;**  **}** **else** **if** **(**getValue**(**name**)** **<** getValue**(**node**.**data**.**getName**()))** **{**  **if** **(**node**.**left **==** **null)** **{**  got **=** **null;**  **}** **else** **{**  got **=** get**(**name **,** node**.**left**);**  **}**    **}** **else** **{**  **if** **(**node**.**right **==** **null)** **{**  got **=** **null;**  **}** **else** **{**  got **=** get**(**name **,** node**.**right**);**  **}**  **}**  **return** got**;**  **}**    @Override  public Supporter put**(**Supporter supporter**)** **{**  putCounter **=** 0**;**    Node node **=** **new** Node**(**supporter**);**    **if** **(**root **==** **null)** **{**  root **=** node**;**  **}** **else** **{**  supporter **=** put**(**supporter**,** root**);**  **}**    System**.**out**.**println**(**"Size: " **+** size**()** **+** "|Depth: " **+** getDepth**()** **+** "|Nodes Visted: " **+** putCounter**);**  printSupportersOrdered**();**  **return** supporter**;**  **}**    private Supporter put**(**Supporter supporter**,** Node node**)** **{**      **if** **(**getValue**(**supporter**.**getName**())** **<** getValue**(**node**.**data**.**getName**()))** **{**  **if** **(**node**.**left **!=** **null)** **{**  putCounter **=** putCounter **+** 1**;**  supporter **=** put**(**supporter**,** node**.**left**);**  **}** **else** **{**  node**.**left **=** **new** Node**(**supporter**);**  **}**  **}** **else** **if** **(**getValue**(**supporter**.**getName**())** **>** getValue**(**node**.**data**.**getName**()))** **{**  **if** **(**node**.**right **!=** **null)** **{**  putCounter **=** putCounter **+** 1**;**  supporter **=** put**(**supporter**,** node**.**right**);**  **}** **else** **{**  node**.**right **=** **new** Node**(**supporter**);**  **}**  **}**    **return** supporter**;**  **}**  @Override  public Supporter remove**(**String name**)** **{**  **throw** **new** UnsupportedOperationException**(**"Not supported yet."**);** //To change body of generated methods, choose Tools | Templates.  **}**  // ...  /\*\*  \* Returns the number of supporters in the database  \*  \* **@pre** true  \* **@return** number of supporters in the database. 0 if empty  \*/  @Override  public int size**()** **{**    **return** size**(**root**);**  **}**    private int size**(**Node node**)** **{**  **if** **(**node **==** **null)** **{**  **return** 0**;**  **}** **else** **{**  **return** 1 **+** size**(**node**.**left**)** **+** size**(**node**.**right**);**  **}**  **}**    /\*\*  \* Determines if the database is empty or not.  \*  \* **@pre** true  \* **@return** true iff the database is empty  \*/  @Override  public boolean isEmpty**()** **{**  **return** root **==** **null;**  **}**  /\*\*  \* Prints the names and IDs of all the supporters in the database in  \* alphabetic order.  \*  \* **@pre** true  \*/  @Override  public void printSupportersOrdered**()** **{**  System**.**out**.**println**(**"Binary Search Tree Ordered Print"**);**  **if** **(**root **==** **null)** **{**  System**.**out**.**println**(**"The Tree is empty."**);**  **}** **else** **{**  printSupportersOrdered**(**root**);**  **}**  **}**    private void printSupportersOrdered**(**Node node**)** **{**    **if** **(**node**.**left **!=** **null)** **{**  printSupportersOrdered**(**node**.**left**);**  **}**  System**.**out**.**println**(**node**.**data**.**getName**()** **+** " " **+** node**.**data**.**getID**());**  **if** **(**node**.**right **!=** **null)** **{**  printSupportersOrdered**(**node**.**right**);**  **}**  **}**  **}** | |

And the main used to run the SupporterDatabase.java:

|  |  |
| --- | --- |
| **DataStucturesCourseworkPart1.java** |  |
| /\*  \* To change this license header, choose License Headers in Project Properties.  \* To change this template file, choose Tools | Templates  \* and open the template in the editor.  \*/  package data**.**structures**.**coursework**.**part**.**pkg1**;**  /\*\*  \*  \* **@author** 17075763  \*/  public class DataStructuresCourseworkPart1 **{**  /\*\*  \* **@param** args the command line arguments  \*/  public static void main**(**String**[]** args**)** **{**        String ID **=** "1"**;**  String name **=** "TROY"**;**    Supporter supporter **=** **new** Supporter**(**name**,** ID**);**    //FOR HASHTABLES UNCOMMENT "HT" AND COMMENT "BST"  ISupporterDatabase db **=** **new** SupporterDatabaseBST**(**supporter**);**  //ISupporterDatabase db = new SupporterDatabaseHT(2);    db**.**put**(**supporter**);**  ISupporter test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getID**()** **+** " " **+** test**.**getName**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "2"**;**  name **=** "THOMAS"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "3"**;**  name **=** "BOB"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    //test = db.remove(name);    ID **=** "62"**;**  name **=** "SUPRISE"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "4"**;**  name **=** "JIM"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "5"**;**  name **=** "PAUL"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "6"**;**  name **=** "ZZZZZ"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "7"**;**  name **=** "ZZZZZZ"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    ID **=** "8"**;**  name **=** "ZZZZZZZ"**;**  supporter **=** **new** Supporter**(**name**,** ID**);**  db**.**put**(**supporter**);**  test **=** db**.**get**(**name**);**  System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**test**.**getName**()** **+** " " **+** test**.**getID**()** **+** " added to table / tree."**);**  System**.**out**.**println**(**" "**);**    System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**"--------"**);**  System**.**out**.**println**(**"containsName() test"**);**  System**.**out**.**println**(**"Is Troy (root of tree) contained in the database: " **+** db**.**containsName**(**"Troy"**));**  System**.**out**.**println**(**"Is Troy (leaf of tree) contained in the database: " **+** db**.**containsName**(**"Thomas"**));**  System**.**out**.**println**(**"Is Haberdasher (Not in the database) contained in the database: " **+** db**.**containsName**(**"Haberdasher"**));**  System**.**out**.**println**(**" "**);**    System**.**out**.**println**(**"--------"**);**  System**.**out**.**println**(**"isEmpty() test: (shouldn't be empty)"**);**  System**.**out**.**println**(**"Is the hash table / binary search tree empty: " **+** db**.**isEmpty**());**    System**.**out**.**println**(**"--------"**);**  System**.**out**.**println**(**"clear() test"**);**  db**.**clear**();**  System**.**out**.**println**(**"clear() called!"**);**    System**.**out**.**println**(**" "**);**  db**.**printSupportersOrdered**();**    System**.**out**.**println**(**" "**);**  System**.**out**.**println**(**"Is the hash table / binary search tree empty: " **+** db**.**isEmpty**());**    **}**    **}** | |

The Output:

|  |  |
| --- | --- |
| **Output** |  |
| run:  Size: 1|Depth: 0|Nodes Visted: 0  Binary Search Tree Ordered Print  TROY 1    1 TROY added to table / tree.    Size: 2|Depth: 1|Nodes Visted: 0  Binary Search Tree Ordered Print  TROY 1  THOMAS 2    THOMAS 2 added to table / tree.    Size: 3|Depth: 1|Nodes Visted: 0  Binary Search Tree Ordered Print  BOB 3  TROY 1  THOMAS 2    BOB 3 added to table / tree.    Size: 4|Depth: 2|Nodes Visted: 1  Binary Search Tree Ordered Print  BOB 3  TROY 1  THOMAS 2  SUPRISE 62    SUPRISE 62 added to table / tree.    Size: 5|Depth: 2|Nodes Visted: 1  Binary Search Tree Ordered Print  BOB 3  JIM 4  TROY 1  THOMAS 2  SUPRISE 62    JIM 4 added to table / tree.    Size: 6|Depth: 3|Nodes Visted: 2  Binary Search Tree Ordered Print  BOB 3  JIM 4  PAUL 5  TROY 1  THOMAS 2  SUPRISE 62    PAUL 5 added to table / tree.    Size: 7|Depth: 3|Nodes Visted: 1  Binary Search Tree Ordered Print  BOB 3  JIM 4  PAUL 5  TROY 1  ZZZZZ 6  THOMAS 2  SUPRISE 62    ZZZZZ 6 added to table / tree.    Size: 8|Depth: 4|Nodes Visted: 2  Binary Search Tree Ordered Print  BOB 3  JIM 4  PAUL 5  TROY 1  ZZZZZ 6  THOMAS 2  ZZZZZZ 7  SUPRISE 62    ZZZZZZ 7 added to table / tree.    Size: 9|Depth: 5|Nodes Visted: 2  Binary Search Tree Ordered Print  BOB 3  JIM 4  PAUL 5  TROY 1  ZZZZZ 6  THOMAS 2  ZZZZZZ 7  SUPRISE 62  ZZZZZZZ 8    ZZZZZZZ 8 added to table / tree.      --------  containsName() test  Is Troy (root of tree) contained in the database: false  Is Troy (leaf of tree) contained in the database: false  Is Haberdasher (Not in the database) contained in the database: false    --------  isEmpty() test: (shouldn't be empty)  Is the hash table / binary search tree empty: false  --------  clear() test  clear() called!    Binary Search Tree Ordered Print  The Tree is empty.    Is the hash table / binary search tree empty: true  BUILD SUCCESSFUL (total time: 0 seconds) | |

|  |  |
| --- | --- |
| F | *“By using your main program from |Part 1 or (JUnit) test your implementation of SupporterDatabaseBST. To do this, create an object of the class.*  *Be sure to check (among many other cases).”* |

|  |  |  |
| --- | --- | --- |
| Testing | | |
| Methods | | |
| Test Data | Expected Results | Actual Output  Green: Pass  Red: Fail |
| **getValue**() | | |
| Getting the value of word:  Troy | 334 | \*Result from the Variable viewer because it’s hard to print this code and doesn’t make much sense to print it either. |
| **getDepth**() | | |
| Getting Depth of Tree:  TROY 1  THOMAS 2 | 1 | Size: 2|Depth: 1|Nodes Visted: 0  Binary Search Tree Ordered Print  TROY 1  THOMAS 2 |
| **clear**() | | |
| Clearing Tree:  Binary Search Tree Ordered Print  BOB 3  JIM 4  PAUL 5  TROY 1  ZZZZZ 6  THOMAS 2  ZZZZZZ 7  SUPRISE 62  ZZZZZZZ 8 | Tree to be clear | Binary Search Tree Ordered Print  The Tree is empty. |
| Is the hash table / binary search tree empty: true |
| **containsName**() | | |
| Tree to Search:  BOB 3  JIM 4  PAUL 5  TROY 1  ZZZZZ 6  THOMAS 2  ZZZZZZ 7  SUPRISE 62  ZZZZZZZ 8 | TROY found | Is Troy (root of tree) contained in the database: true |

|  |  |  |
| --- | --- | --- |
| Tree to Search:  BOB 3  JIM 4  PAUL 5  TROY 1  ZZZZZ 6  THOMAS 2  ZZZZZZ 7  SUPRISE 62  ZZZZZZZ 8 | THOMAS found | Is Thomas (leaf of tree) contained in the database: true |
| Tree to Search:  BOB 3  JIM 4  PAUL 5  TROY 1  ZZZZZ 6  THOMAS 2  ZZZZZZ 7  SUPRISE 62  ZZZZZZZ 8 | HARRY not found | Is HARRY (Not in the database) contained in the database: false |
| **get**() | | |
| Supporter to get:  Supporter(TROY, 1) | Displays name | 1 TROY added to table / tree. |
| **put**() | | |
| Tree Before **put**() is called:  BOB 3  TROY 1  THOMAS 2  SUPRISE 62 | For JIM to be added to the tree | Size: 5|Depth: 2|Nodes Visted: 1  Binary Search Tree Ordered Print  BOB 3  JIM 4  TROY 1  THOMAS 2  SUPRISE 62 |
| **remove**() | | |
| - | - | It doesn’t have a body |
| **size**() | | |
| Getting size of tree:  BOB 3  JIM 4  PAUL 5  TROY 1  ZZZZZ 6  THOMAS 2  SUPRISE 62 | 7 | Size: 7|Depth: 3|Nodes Visted: 1  Binary Search Tree Ordered Print  BOB 3  JIM 4  PAUL 5  TROY 1  ZZZZZ 6  THOMAS 2  SUPRISE 62 |

|  |  |  |
| --- | --- | --- |
| **isEmpty**() | | |
| Tree **isEmpty**() is being called on:  BOB 3  JIM 4  PAUL 5  TROY 1  ZZZZZ 6  THOMAS 2  ZZZZZZ 7  SUPRISE 62  ZZZZZZZ 8 | false | isEmpty() test: (shouldn't be empty)  Is the hash table / binary search tree empty: false |
| Tree **isEmpty**() is being called on: | true | Is the hash table / binary search tree empty: true |
| **printSupporterOrdered**() | | |
| Information used to print table (in the order it added to the database):  TROY 1  THOMAS 2  BOB 3  SURPRISE 62  JIM 4  PAUL 5  ZZZZZ 6  ZZZZZZ 7  ZZZZZZZ 8  ZZZZZZZZ 9 | For the information to be in alphabetical order | Size: 9|Depth: 5|Nodes Visted: 2  Binary Search Tree Ordered Print  BOB 3  JIM 4  PAUL 5  TROY 1  ZZZZZ 6  THOMAS 2  ZZZZZZ 7  SUPRISE 62  ZZZZZZZ 8 |

|  |  |
| --- | --- |
| G | *“* State honestly which of the requirements of Part 2 you have successfully fulfilled, citing evidence.*.”* |

|  |  |
| --- | --- |
| FR5 | Done. See: logs pasted in above. No logs for the **remove**() function because I didn’t implement it. |
| FR6 | Done. See: source files pasted in above. No logs for the **remove**() function because I didn’t implement it. |

There might be small discrepancies between bits of code in this document, such as now the code produces and accepts names in “CAPITIALS” instead of “Only Capitalised First Letters In Names”, this is because the codes been updated along with this document, all the important changes to the code are reflected in this report.