CS 5040 (Dr. Shaffer): Project 1: try #9 (10.0/100.0)

src/Main.java

View: Overall summary of results Go

```
1
2   /**
3    * {Project Description Here}
4    */
5
6   /**
7    * The class containing the main method.
8    *
9    * @author Aayush Bagrecha
10   * @author Yash Shrikant
11   * @version 1.0
12   */
13
14  // On my honor:
```

```
15  // - I have not used source code obtained from another current or
16  //   former student, or any other unauthorized source, either
17  //   modified or unmodified.
18  //
19  // - All source code and documentation used in my program is
20  //   either my original work, or was derived by me from the
21  //   source code published in the textbook for this course.
22  //
23  // - I have not discussed coding details about this project with
24  //   anyone other than my partner (in the case of a joint
25  //   submission), instructor, ACM/UPE tutors or the TAs assigned
26  //   to this course. I understand that I may discuss the concepts
27  //   of this program with other students, and that another student
28  //   may help me debug my program so long as neither of us writes
29  //   anything during the discussion or modifies any computer file
30  //   during the discussion. I have violated neither the spirit nor
```

```
31  //   letter of this restriction.
32
33  import java.io.File;
34  import java.io.FileWriter;
35  import java.io.PrintWriter;
36  import java.util.Scanner;
37
38  public class Main {
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
The Javadoc for this class or interface is missing. All visible (i.e. not private) classes and interfaces must be docume

```
39
40      /**
41       * @param args
42       */
43
44      public static void main(String[] args) {
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
The @param tag for the parameter 'args' is missing or incomplete. The @param tag should be followed by the nam of the list, the first name of the customer, etc.). There should be one @param tag for each parameter in the metho

```
45          // int MEMORY_POOL_SIZE = Integer.parseInt(args[0]);
46          // int INITIAL_CAPACITY = Integer.parseInt(args[1]);
47          // String filename = args[2];// Pass the function a full fi
48
49          // System.out.println(MEMORY_POOL_SIZE + INITIAL_CAPACITY +
50
51          String filename = "/Users/yashshrikant/Documents/Courses/Int
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
52          int MEMORY_POOL_SIZE = 64;
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
Variables should be named in camelCase where the name starts with a lowercase letter and the first letter of each

```
53          int INITIAL_CAPACITY = 4;
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
Variables should be named in camelCase where the name starts with a lowercase letter and the first letter of each

```
54
55          beginParsing(filename, MEMORY_POOL_SIZE, INITIAL_CAPACITY);
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

☐ **Error [Checkstyle]: 0 (limit exceeded)**
';' is not followed by whitespace (i.e. a space, tab, or new line). Add a space after the ';'.

```
56      }
57
58      public static void beginParsing(String filename, int MEMORY_POOL
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

☐ **Error [Checkstyle]: 0 (limit exceeded)**
The Javadoc for this method or constructor is missing. All visible (i.e. not private) methods and constructors must b

☐ **Error [Checkstyle]: 0 (limit exceeded)**
Parameters should be named in camelCase where the name starts with a lowercase letter and the first letter of eac

☐ **Error [Checkstyle]: 0 (limit exceeded)**
Parameters should be named in camelCase where the name starts with a lowercase letter and the first letter of eac

```
59          try {
60
61              String outputFile = "output.txt";
62              PrintWriter writer = new PrintWriter(new FileWriter(outp
63
64              HashTable ht = new HashTable(MEMORY_POOL_SIZE, INITIAL_C
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
65              Scanner lines = new Scanner(new File(filename));// Creat
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

☐ **Error [Checkstyle]: 0 (limit exceeded)**
';' is not followed by whitespace (i.e. a space, tab, or new line). Add a space after the ';'.

```
66              while (lines.hasNext()) {// While the scanner has inform
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

Error [Checkstyle]: 0 (limit exceeded)
'{' is not followed by whitespace (i.e. a space, tab, or new line). Most operators should have spaces both before and

```
67                  String cmd = lines.nextLine().replaceAll("\\s+", " "
68                  ;// Read the next term
```

Error [Checkstyle]: 0 (limit exceeded)
This statement is just a ';' and, therefore, does nothing. You may have added a ';' in an inappropriate location (like

```
69
70                  String verb = cmd.split("\\s")[0];
71
72                  switch (verb) {
73                      case "insert":
74                          int id = Integer.parseInt(cmd.split(" ")[1])
75                          String title = lines.nextLine();
76                          String dateField = lines.nextLine().replaceA
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
77                          String date = dateField.split(" ")[0];
```

```
78                          int length = Integer.parseInt(dateField.spli
79                          short x = Short.parseShort(dateField.split("
80                          short y = Short.parseShort(dateField.split("
81                          int cost = Integer.parseInt(dateField.split(
82                          String keywords = lines.nextLine().replaceAl
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
83                          String description = lines.nextLine().replac
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
84                          Record record = new Record(id, title, date,
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
85                          boolean inserted = ht.insert(record);
86                          if (inserted == true) {
87                              writer.println("Successfully inserted re
```

☐ **Error [Checkstyle]: 0 (limit exceeded)**
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
88                              writer.println("ID:" + id + ", Title: "
89                              writer
```

```
90                                  .println("Date: " + date + ", Le
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
91                                          + ", Y: " + y + ", Cost:
92                                  writer.println("Description: " + descrip
93                                  writer.println("Keywords: " + keywords);
94                                  // writer.println("Size: " + record.calc
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
95                              } else {
```

Error [Checkstyle]: 0 (limit exceeded)
This '}' should be alone on a line (i.e. no other code should be after it on the same line).

```
96                                  writer.println("Insert FAILED - There is
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
97                              }
98                              break;
99                          case "search":// Found an search command
100                             id = Integer.parseInt(cmd.split(" ")[1]);
101                             record = ht.search(id);
```

```
102                         if (record != null) {
103                             writer.println("Found record with ID " +
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
104                             writer.println("ID:" + record.ID + ", Ti
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
105                             writer
106                                 .println("Date: " + record.Date
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
107                                 + ", Y: " + record.Y + "
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
108                             writer.println("Description: " + record.
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
109                             writer.println("Keywords: " + record.Key
110                         }
111                         break;
112                     case "delete":// Found a delete command
```

```
113                         id = Integer.parseInt(cmd.split(" ")[1]);
114                         boolean deletedStatus = ht.delete(id);
115                         if (deletedStatus == true) {
116                             writer.println("Record with ID " + id +
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
117                         } else {
```

Error [Checkstyle]: 0 (limit exceeded)
This '}' should be alone on a line (i.e. no other code should be after it on the same line).

```
118                             writer.println("Delete FAILED -- There i
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
119                         }
120                         break;
121                     case "print":// Found a print command
122                         String printCondition = cmd.split(" ")[1].re
```

Error [Checkstyle]: 0 (limit exceeded)
This line is longer than 80 characters. Break it into multiple lines so that it is easier to read.

```
123
124                         if (printCondition.equals("blocks")) {
```

```
125                                 ht.printMemoryBlocks();
126                             } else {
```

Error [Checkstyle]: 0 (limit exceeded)
This '}' should be alone on a line (i.e. no other code should be after it on the same line).

```
127                                 ht.printHashTable();
128                             }
129                         default:// Found an unrecognized command
```

Error [Checkstyle]: 0 (limit exceeded)
The case above will fall through and run the code in this case too. This may be an indication that a break statemen comment on the last line of the previous case with the phrase 'falls through' to indicate that.

```
130                             break;
131                     }
132                 }
133
134                 writer.close();
135             } catch (Exception e) {
```

Error [Checkstyle]: 0 (limit exceeded)
This '}' should be alone on a line (i.e. no other code should be after it on the same line).

```
136                 e.printStackTrace();
137             }
```

```
138         }
139
140     }
141
142     /*
143      * - implement handle class in records
144      * - code refactoring and documentation
145      * - mutation testing
146      * - Junit testing
147      */
```