Tejveer Singh

Ts355

Section 004

**1. Exercises (25 points)**

1. (15 points) Show the syntax (form) , explain its meaning, and give an example of each of
   * 1. while loop,

SYNTAX:

//while (expression){

// statement(s)

//}

MEANING:

While the condition is met, do the following until the condition is not met anymore.

EXAMPLE:

//int x = 3;

//while (x > 0) {

// x = x-1;

//}

* + 1. for loop,

SYNTAX:

//for (initialization; termination; increment) {

// statement(s)

//}

MEANING:

Do the following code statement(s) from the point it is initialized to the termination point following the increment rule. Once it reaches or goes over the termination point end the loop.

EXAMPLE:

//for (int i=1; i<10; i++){

// System.out.println("Count: " + i);

//}

* + 1. if statement,

SYNTAX:

//If (expression) {

// statement(s)

//}

MEANING:

If the condition or expression is true, it performs the code statements. If it is false, it does not perform the statement.

* + 1. array (give an example declaration and its use), and

SYNTAX:

//dataType[] arrayName;

MEANING/USE:

An array is an object that is used to store multiple variables of the same data type mainly for organization for more complex coding practices.

EXAMPLE:

//double[] aList;

* + 1. boolean type.

SYNTAX:

//boolean variableName = true;

MEANING:

Boolean expressions can only hold two possible values: either true or false. This expression can be used to determine other actions in code when paired with loops or if statements.

EXAMPLE:

//String list[] = [“A”, “bee”];

//String word;

//boolean notInList = true:

//if (notInList) {

// list+= word;

//}

1. (5 points) What does a break statement do when executed inside a loop?

A break statement automatically cuts out of the loop.

1. (5 points) String is a class type, not a primitive type, but it is treated specially in Java. Give an example of this special treatment.

A string is treated specially because it is used so much in programming. It has special operands that work with it such as concatenation (+) and is immutable.

//name = “Andrew”

//system.out.println(“Hi “ + name + “!”);

**2. Directory[[1]](#footnote-1) Lookup Application (75 points)**

**What is the assignment?**

Write a directory (database) lookup application which operates by

1. reading a list of strings stored in file DataS.txt, to create a database of strings:

* The list of strings is preceded by a number that specifies the size of the list.
* This file name Data3.txt is supplied as a command-line argument to the application.

1. accepting user input supplied at the keyboard which are strings that it checks to see whether or not they are in its database.

For each string, x, input by the user, the program prints “x is in the list” or “x is not in the list” depending upon whether or not x is in the list.

The program terminates when the user enters Ctrl-Z for Windows (Ctrl-D for Linux) (Enter EOT for jGRASP) on the keyboard. Before terminating it prints the number of searchers performed.

**Input File DataS.txt**

14

Christian

Vincent

Joseph

Unman

Andrew

James

Ali

Narain

Chengjun

Marvin

Frank

Jason

Reza

David

The list has 14 values as indicated by the first entry in the file.

**Sample user interaction (submit your version of the user interaction)**

Database Server is Ready for String Lookups!

String to Lookup: Ralph

>> Ralph is not in the database

String to Lookup: James

>> James is in the database

String to Lookup: George

>> George is not in the database

Number to Lookup: ^Z

Number of searches = 3  
Goodbye!

**Hints for writing the program**

1. Use the ReverseList.java program we studied in class as example for reading input from a file whose name is supplied as a command-line argument:
   1. check to make sure that a command-line argument is supplied
   2. set up a Scanner object to read from a file (file named DataS.txt)
   3. read the data in the file and store it in an array (named, for example, list)
2. Set up a Scanner object to read input from the keyboard (from System.in)
3. As long as there is input to be read, read input, see if the value read is in the array list or not and print an appropriate message.
4. You will need 3 loops   
   1. one for reading data from the file,
   2. one for reading user inputs, and
   3. one for checking whether or not the value supplied by the user is in the list.

The third loop will be nested inside the second loop. The first and third loops could be for loops like the one in ReverseList.java and the second one would be a while loop as in the example copy.java that we also did in class.

1. To see whether or not a value is in the list of strings stored in the array, you will have to use an if statement to compare values using the String method *equalsIgnoreCase(String s)* which returns *true* is both strings have the same characters irrespective of case otherwise it returns *false*.
2. If you find that the value is in the list, then you need to exit from the nested for loop (third loop) using the break statement.
3. Upon exiting the third loop (either the loop completes by not finding the value we are searching for or it terminates because we found the value and we used a break statement to force an exit from the loop) we need to know whether or not the value was found. For this, before starting execution of the loop, we can use a boolean variable, say found, set to false indicating that we have not found the value as yet. Variable found is set to true in the loop if the value is found in the list (prior to executing the break statement).
4. The value of found controls the message printed that specifies whether or not we found the searched value.
5. You need to count the number of searches that are performed. Start off with the number of searches being zero and increment this number by one each time you look up a number. Print this number when exiting the program.

**Grading**

The program will be graded as follow for a total of 75 points:

* Appropriate comments: 10 points
* Appropriate indentation: 10 points
* Reads the data from a file: 5 points
* Sets up an array of the correct size and stores the data in the array: 10 points
* Does the searches correctly: 25 points
* Prints the number of searches correctly: 5 points
* Good user interaction (asks user for input): 5 points
* Nicely formatted output: 5 points

PROGRAM

1 //Tejveer Singh  
 2 //ts355  
 3 //Section 004  
 4   
 5 import java.util.\*;  
 6 import java.io.\*; //file reader class  
 7   
 8 public class DirectoryLookUp  
 9 {  
10 public static void main( String[] args )  
11 {  
12 Scanner stdin = new Scanner( System.in );  
13 int listSize = 0;  
14 int count = 0; // tracks number of searches  
15 String[] list = new String[listSize];  
16 try{  
17 FileReader inputFile = new FileReader("DataS.txt");  
18 BufferedReader bufferReader = new BufferedReader(inputFile);  
19   
20 String strNum = bufferReader.readLine();  
21 listSize = Integer.parseInt(strNum.trim()); //saves number of values in database a int  
22 list = new String[listSize];  
23 for(int i=0; i<listSize; i++){ //fills the data base with values   
24 list[i] = (bufferReader.readLine()).trim();   
25 }  
26 bufferReader.close();  
27 }  
28 catch(Exception e){ //prints out error message if thier is a problem with reading from DataS.txt  
29 System.out.println("Error while reading file line by line:" + e.getMessage());   
30 e.printStackTrace();   
31 }  
32   
33 //check the the database  
34   
35 for (int i=1; i <= list.length; i++){  
36 System.out.println("#"+i+" "+list[i-1]);  
37 }  
38   
39 System.out.println("Database Server is Ready for String Lookups!");  
40 System.out.print("String to Lookup:");  
41 String input = stdin.next();  
42 do{  
43 boolean find = false;  
44 for(int i=0; i<listSize; i++){ //searches if imput is in the database  
45 if(input.equalsIgnoreCase(list[i])){ // test if user iumput is in data base  
46 find = true;  
47 break;  
48 }  
49 else  
50 find = false;  
51 }  
52 if(find)  
53 System.out.println(input + " is in is in the database");  
54 else  
55 System.out.println(input + " is not in is in the database");   
56 count++;  
57 System.out.print("String to Lookup:");  
58 input = stdin.next();  
59 }while(stdin.hasNext());  
60 System.out.println("number of searches = " + count);  
61   
62 }  
63 }

1. The word *directory* is used here similar to its use in "telephone" directory or a "store" directory" as opposed to its use in "file" directory which refers to the organization of files in a computer system such as Linux or Windows. [↑](#footnote-ref-1)