```
 1 ************************************************
 2 * PROGRAMMED BY : Blake Allard & Khaled Ajaj
 3 * CLASS         : CS1A
 4 * SECTION       : MW - 8am
 5 * LAB #15       : Arrays & Files - Name Search
 6 ************************************************
 7
 8 Who do you want to search for (enter done to exit)? Joe
 9 There are 4 instances of the name Joe.
10
11 Who do you want to search for (enter done to exit)? Sally
12 There are 2 instances of the name Sally.
13
14 Who do you want to search for (enter done to exit)? Adam
15 There are 3 instances of the name Adam.
16
17 Who do you want to search for (enter done to exit)? Sue
18 There is one instance of the name Sue.
19
20 Who do you want to search for (enter done to exit)? John
21 John's name does not exist in this list.
22
23 Who do you want to search for (enter done to exit)? done
24
25 Thank you for using my program.
26
27 NAME          INSTANCES
28 ----          ---------
29 Joe           4
30 Sally         2
31 Adam          3
32 Sue           1
33 John          -
34
```

```
 1 ************************************************
 2 * PROGRAMMED BY : Blake Allard & Khaled Ajaj
 3 * CLASS         : CS1A
 4 * SECTION       : MW - 8am
 5 * LAB #15       : Arrays & Files - Name Search
 6 ************************************************
 7
 8
 9 NAME          INSTANCES
10 ----          ---------
11 Joe           4
12 Sally         2
13 Adam          3
14 Sue           1
15 John          -
16
```

```
 1 /*****************************************************************************
 2  * AUTHOR     : Blake Allard , Khaled Ajaj
 3  * STUDENT ID : 358888       , 1125796
 4  * LAB #15    : Arrays & Files
 5  * CLASS      : CS1A
 6  * SECTION    : MW 8am
 7  * DUE DATE   : 11/27/24
 8  *****************************************************************************/
 9
10 #ifndef HEADER_H_
11 #define HEADER_H_
12
13 #include <iostream>   /* cout , cin    */
14 #include <iomanip>    /* setw          */
15 #include <string>     /* string        */
16 #include <fstream>    /* fout , fin    */
17 #include <sstream>    /* ostreamstring */
18 using namespace std;
19
20
21 /*****************************************************************************
22  * OutputClassHeader
23  *   This function receives an assignment name, type and number then outputs
24  *   the appropriate class header.
25  *   asType is defaulted to Lab as there are more labs than Assignments.
26  *---------------------------------------------------------------------------
27  *   ==> returns nothing - This will output the class heading.
28  *****************************************************************************/
29
30 string OutputClassHeader(string asName,        // IN - assignment Name
31                          int    asNum,         // IN - assignment number
32                          char   asType);       // IN - assignment type
33                                                // 'L' = Lab
34                                                // 'A' = Assignment
35
36 /*****************************************************************************
37  * ReadInputFile
38  *   This function passes in an array and a constant array size in order to be
39  *   used for reading in data from an input file.
40  *---------------------------------------------------------------------------
41  *   ==> returns nothing - the function returns nothing, passes the array &
42  *                         array size by reference to read the input file into
43  *                         the array
44  *****************************************************************************/
45
46 void ReadInputFile(string    namesArray[],        // IN - array of names
47                    int       AR_SIZE,             // IN - size of the array
48                    ifstream &fin);                // IN - read file input
49
50
51 /*****************************************************************************
```

```
 52 * GetAndSearchName
 53 *   This function takes in the name input, and checks to see if it is found
 54 *      within the array of names. If it is a part of the name array, it then
 55 *      checks to see which name it matches. Once the matched name is found,
 56 *      the number of instances for that name is output.
 57 *
 58 *      If the name is not found within the name array, it will return a message
 59 *      saying the name is not a part of the name array. It will then track the
 60 *      name for output into the final table.
 61 *-------------------------------------------------------------------------
 62 *   ==> returns nothing - the function returns nothing, alters the value for
 63 *      nameNotFound variable, and outputs the count of the specified name.
 64 *************************************************************************/
 65 void GetAndSearchName(string  namesArray[],    // IN     - array of names
 66                 const int     AR_SIZE,        // IN     - size of the array
 67                       int     index,          // ASSIGN - index of namesArray
 68                       bool    found,          // ASSIGN - if name is found
 69                       string  name,           // IN     - name input by user
 70                       int     joeCount,       // OUT    - instances of "Joe"
 71                       int     sallyCount,     // OUT    - instances of "Sally"
 72                       int     adamCount,      // OUT    - instances of "Adam"
 73                       int     sueCount,       // OUT    - instances of "Sue"
 74                       string &nameNotFound);  // OUT    - name not found
 75
 76 /************************************************************************
 77 * OutputNamesAndCounts
 78 *   This function outputs the table of names and instances of each name in the
 79 *      array of names. It then outputs the name not found that the user input
 80 *      into the program.
 81 *-----------------------------------------------------------------------
 82 *   ==> returns nothing - the function returns nothing, outputs a table of
 83 *      names and arrays.
 84 ************************************************************************/
 85 string OutputNamesAndCounts(string namesArray[],  // IN     - array of names
 86                             int    AR_SIZE,       // IN     - size of the array
 87                             int    joeCount,      // OUT    - count of "Joe"
 88                             int    sallyCount,    // OUT    - count of "Sally"
 89                             int    adamCount,     // OUT    - count of "Adam"
 90                             int    sueCount,      // OUT    - count of "Sue"
 91                             string nameNotFound); // OUT    - name not found
 92
 93 /************************************************************************
 94 * CountItemsInArray
 95 *   This function counts the instances of each name in the names array. It
 96 *      loops 10 times to account for the number of items in the input file.
 97 *-----------------------------------------------------------------------
 98 *   ==> returns nothing - the function returns nothing, increments each name's
 99 *      counter on each instance.
100 ************************************************************************/
101 void CountItemsInArray(string namesArray[],   // IN     - array of names
102                        int    AR_SIZE,        // IN     - size of the array
```

```
103                         int     &joeCount,      // OUT    - count of "Joe"
104                         int     &sallyCount,    // OUT    - count of "Sally"
105                         int     &adamCount,     // OUT    - count of "Adam"
106                         int     &sueCount);     // OUT    - count of "Sue"
107
108
109
110 #endif /* HEADER_H_ */
111
```

```cpp
/*****************************************************************************
 * AUTHOR     : Blake Allard , Khaled Ajaj
 * STUDENT ID : 358888       , 1125796
 * LAB #15    : Arrays & Files
 * CLASS      : CS1A
 * SECTION    : MW 8am
 * DUE DATE   : 11/27/24
 *****************************************************************************/

#include "header.h"

/*****************************************************************************
 * PROGRAM DESCRIPTION
 * ---------------------------------------------------------------------------
 * This program will read in an input file containing a list of 10 names and
 * store them in an array, then the program will prompt the user for their name
 * and will output the total number of name instances to the console and to an
 * output file.  The program will continue to prompt the user to input names
 * until "done" is entered.
 * ---------------------------------------------------------------------------
 * INPUT: the user will input their name and the number of rounds they wish to
 *        play.
 *
 * OUTPUT: the program will output the total number of instances an entered name
 *         appears in the program to the console and to an output file.
 *
 *
 * Example Input/Output:
 *
 * Who do you want to search for (enter done to exit)? Joe
 * There are 4 instances of the name Joe.
 *
 * Who do you want to search for (enter done to exit)? Sally
 * There are 2 instances of the name Sally.
 *
 * Who do you want to search for (enter done to exit)? Adam
 * There are 3 instances of the name Adam.
 *
 * Who do you want to search for (enter done to exit)? Sue
 * There is one instance of the name Sue.
 *
 * Who do you want to search for (enter done to exit)? John
 * John's name does not exist in this list.
 *
 * Who do you want to search for (enter done to exit)? done
 *
 * Thank you for using my program.
 *
 *****************************************************************************/

int main()
```

```
52 {
53
54    /*************************************************************************
55     * CONSTANTS
56     * -----------------------------------------------------------------------
57     * PROCESSING - USED TO DEFINE ARRAY SIZE
58     * -----------------------------------------------------------------------
59     * AR_SIZE - NUMBER OF ARRAY ELEMENTS
60     *************************************************************************/
61
62     //PROCESSING - used for array size
63     const int AR_SIZE = 10;
64
65     /*************************************************************************
66      * VARIABLES
67      *************************************************************************/
68
69     ifstream fin;
70     ofstream fout;                          // PROC   & OUT  - file output
71     string   classHeader;                   //          OUT  - class header
72     string   asName;                        //          OUT  - asn name
73     int      asNum;                         //          OUT  - asn number
74     char     asType;                        //          OUT  - asn type
75     string   name;                          // IN     & OUT  - name
76     string   namesArray[AR_SIZE];           // CALC          - array of names
77     int      joeCount;                      // CALC   & OUT  - # joe counts
78     int      sallyCount;                    // CALC   & OUT  - # sally counts
79     int      sueCount;                      // CALC   & OUT  - # sue counts
80     int      adamCount;                     // CALC   & OUT  - # adam counts
81     int      index;                         // ASSIGN & OUT  - array index
82     bool     found;                         // ASSIGN & CALC - name found
83     string   instanceOutput;                // ASSIGN & OUT  - name table
84     string   nameNotFound;                  // ASSIGN & OUT  - not found name
85
86     /*************************************************************************
87      * INITIALIZATIONS
88      *************************************************************************/
89
90     asName     = "Arrays & Files"
91                  " - Name Search";          //        OUT - assignment name
92     asNum      =  15;                      //        OUT - assignment number
93     asType     = 'L';                      //        OUT - assignment type
94     joeCount   = 0;                        // CALC & OUT - # of joe counts
95     sallyCount = 0;                        // CALC & OUT - # of sally counts
96     sueCount   = 0;                        // CALC & OUT - # of sue counts
97     adamCount  = 0;                        // CALC & OUT - # of adam counts
98     index      = 0;                        // CALC       - element name
99
100
101    fin.open("NamesInputFile.txt");
102    fout.open("NamesOutputFile.txt");
```

```
103
104
105     //PROCESSING - function call class header into string variable
106     classHeader = OutputClassHeader(asName, asNum, asType);
107
108     //OUTPUT - output the converted class heading string to console and file
109     cout << classHeader;
110     fout << classHeader;
111
112     //PROCESSING - the program reads in the names from "NamesInputFile.txt" and
113     //             stores the names in the array namesArray[AR_SIZE]
114     ReadInputFile(namesArray, AR_SIZE, fin);
115
116
117     /*****************************************************************************
118      * INPUT - prompt the user for a name to search the number of instances
119      * ---------------------------------------------------------------------------
120      * EXAMPLE:
121      *          Who do you want to search for (enter done to exit)? Joe
122      *          There are 4 instances of the name Joe.
123      *****************************************************************************/
124
125     cout << "Who do you want to search for (enter done to exit)? ";
126     getline(cin, name);
127
128     //FUNCTION CALL - updates specific name counter if applicable
129     CountItemsInArray(namesArray,
130                       AR_SIZE,
131                       joeCount,
132                       sallyCount,
133                       adamCount,
134                       sueCount);
135
136     /*****************************************************************************
137      * PROCESSING - the name is compared to specific names in the input file, if
138      *              the name does appear in the input file, the amount of times
139      *              it appears is output, if the name does not appear in the
140      *              input file, a specific prompt is output telling the name is
141      *              not found in the list.
142      *****************************************************************************/
143
144     while(name != "done" && name != "Done")
145         {
146
147             //INITIALIZATION - reinitializing variables each loop iteration
148             index = 0;
149             found = false;
150
151             //FUNCTION CALL - comparing name input to valid input list names
152             GetAndSearchName(namesArray,
153                              AR_SIZE,
```

```
                              index,
                              found,
                              name,
                              joeCount,
                              sallyCount,
                              adamCount,
                              sueCount,
                              nameNotFound);

            // INPUT - reprompt the user to enter a name
            cout << endl << endl;
            cout << "Who do you want to search for (enter done to exit)? ";
            getline(cin, name);

        }

    /*****************************************************************************
     * OUTPUT - output the names entered to console and to file out
     * ---------------------------------------------------------------------------
     * EXAMPLE:
     *         NAME          INSTANCES
     *         ----          ---------
     *         Joe           4
     *         Sally         2
     *         Adam          3
     *         Sue           1
     *         John          -
     *
     *****************************************************************************/

        //OUTPUT - output goodbye prompt
        cout << endl;
        cout << "Thank you for using my program. ";
        cout << endl;

        //PROCESSING - function call to output name table
        instanceOutput = OutputNamesAndCounts(namesArray,
                                              AR_SIZE,
                                              joeCount,
                                              sallyCount,
                                              adamCount,
                                              sueCount,
                                              nameNotFound);

        //OUTPUT - output names table to console and file
        cout << instanceOutput;
        fout << instanceOutput;

        //FORMATTING - closing file
        fin.close();
        fout.close();
```

```
205
206 return 0;
207
208 }
209
```

```cpp
/*******************************************************************************
 * AUTHOR     : Blake Allard , Khaled Ajaj
 * STUDENT ID : 358888       , 1125796
 * LAB #15    : Arrays & Files
 * CLASS      : CS1A
 * SECTION    : MW 8am
 * DUE DATE   : 11/27/24
 *******************************************************************************/

#include "header.h"

/*******************************************************************************
* FUNCTION CountItemsInArray
*------------------------------------------------------------------
* This function counts the instances of each name in the names array. It
* loops 10 times to account for the number of items in the input file.
*
*------------------------------------------------------------------
* PRE-CONDITIONS
* The following parameters need to have a defined value
* prior to calling the function:
* namesArray: Array of names
* AR_SIZE : Size of array of names
* joeCount: instances of "Joe"
* sallyCount: instances of "Sally"
* adamCount: instances of "Adam"
* sueCount: instances of "Sue"
*
* POST-CONDITIONS
* This function will increment the instances of each name in the input file.
*******************************************************************************/

void CountItemsInArray(string namesArray[],   // IN     - array of names
                       int    AR_SIZE,        // IN     - size of the array
                       int    &joeCount,      // OUT    - count of "Joe"
                       int    &sallyCount,    // OUT    - count of "Sally"
                       int    &adamCount,     // OUT    - count of "Adam"
                       int    &sueCount)      // OUT    - count of "Sue"
{
    int    index;      // IN -
    string name;

    for (index = 0; index < AR_SIZE; index++)
    {
        name = namesArray[index];

        if(name == "Joe")
        {
            joeCount++;
        }
        else if(name == "Sally")
```

```
52          {
53              sallyCount++;
54          }
55          else if(name == "Adam")
56          {
57              adamCount++;
58          }
59          else if(name == "Sue")
60          {
61              sueCount++;
62          }
63      }
64  }
65
```

```cpp
/*******************************************************************************
 * AUTHOR     : Blake Allard , Khaled Ajaj
 * STUDENT ID : 358888       , 1125796
 * LAB #15    : Arrays & Files
 * CLASS      : CS1A
 * SECTION    : MW 8am
 * DUE DATE   : 11/27/24
 *******************************************************************************/

#include "header.h"

/*******************************************************************************
* FUNCTION GetAndSearchName
*------------------------------------------------------------------
* This function takes in the name input, and checks to see if it is found
* within the array of names. If it is a part of the name array, it then
* checks to see which name it matches. Once the matched name is found,
* the number of instances for that name is output.
*
*------------------------------------------------------------------
* PRE-CONDITIONS
* The following parameters need to have a defined value
* prior to calling the function:
* namesArray: Array of names
* AR_SIZE : Size of array of names
* index: the index of the array of names
* found: determines if name was found in array or not
* name: the name input
* joeCount: instances of "Joe"
* sallyCount: instances of "Sally"
* adamCount: instances of "Adam"
* sueCount: instances of "Sue"
* nameNotFound: name not found in array
*
* POST-CONDITIONS
* This function will output the name input and its instance.
*******************************************************************************/

void GetAndSearchName(string  namesArray[],    // IN     - array of names
                const int     AR_SIZE,         // IN     - size of the array
                      int     index,           // ASSIGN - index of namesArray
                      bool    found,           // ASSIGN - if name is found
                      string  name,            // IN     - name input by user
                      int     joeCount,        // OUT    - instances of "Joe"
                      int     sallyCount,      // OUT    - instances of "Sally"
                      int     adamCount,       // OUT    - instances of "Adam"
                      int     sueCount,        // OUT    - instances of "Sue"
                      string &nameNotFound)    // OUT    - name not found
{

    // PROCESSING - loops through entire array of names, checks for name found
```

```
 52     while(!found && index < AR_SIZE)
 53     {
 54         if(namesArray[index] == name)
 55         {
 56             found = true;
 57             if(name == "Joe")
 58             {
 59                 if (joeCount != 1)
 60                 {
 61                     cout << "There are " << joeCount
 62                          << " instances of the name Joe.";
 63                 }
 64                 else
 65                 {
 66                     cout << "There is one"
 67                          << " instance of the name Joe.";
 68                 }
 69             }
 70             else if(name == "Sally")
 71             {
 72                 if (sallyCount != 1)
 73                 {
 74                     cout << "There are " << sallyCount
 75                          << " instances of the name Sally.";
 76                 }
 77                 else
 78                 {
 79                     cout << "There is one"
 80                          << " instance of the name Sally.";
 81                 }
 82             }
 83             else if(name == "Adam")
 84             {
 85                 if (adamCount != 1)
 86                 {
 87                     cout << "There are "<< adamCount
 88                          << " instances of the name Adam.";
 89                 }
 90                 else
 91                 {
 92                     cout << "There is one"
 93                          << " instance of the name Adam.";
 94                 }
 95             }
 96             else if(name == "Sue")
 97             {
 98                 if (sueCount != 1)
 99                 {
100                     cout << "There are " << sueCount
101                          << " instances of the name Sue.";
102                 }
```

```
103                 else
104                 {
105                     cout << "There is one"
106                          << " instance of the name Sue.";
107                 }
108             }
109             else
110             {
111                 cout << "There are no instances of the name \n" << name;
112             }
113         }
114
115         index++;
116     }
117
118     // PROCESSING - stores name not found, outputs lack of instances
119     if (index == AR_SIZE)
120     {
121         cout << name << "'s name does not exist in this list. ";
122
123         nameNotFound = name;
124
125     }
126
127 }
128
```

```cpp
/*******************************************************************************
 * AUTHOR     : Blake Allard , Khaled Ajaj
 * STUDENT ID : 358888       , 1125796
 * LAB #15    : Arrays & Files
 * CLASS      : CS1A
 * SECTION    : MW 8am
 * DUE DATE   : 11/27/24
 *******************************************************************************/

#include "header.h"

/*****************************************************************
* FUNCTION OutputClassHeader
*----------------------------------------------------------------
* This function receives an assignment name, type
* and number then outputs the appropriate class header -
* returns nothing.
*
*----------------------------------------------------------------
* PRE-CONDITIONS
* The following parameters need to have a defined value
* prior to calling the function:
* asName: Assignment Name
* asNum : Assignment Number
* asType: Assignment Type ==> THIS SHOULD CONTAIN:
* 'L' for Labs
* 'A' for Assignments
*
* POST-CONDITIONS
* This function will output the class heading.
* <Post-conditions are the changed outputs either
* passed by value or by reference OR anything affected
* by the function>
*****************************************************************/

string OutputClassHeader(string asName,        // IN - assignment Name
                         int    asNum,         // IN - assignment number
                         char   asType)        // IN - assignment type
                                               // 'L' = Lab
                                               // 'A' = Assignment
{

    /*************************************************************************
     * CONSTANTS
     * -----------------------------------------------------------------------
     * FORMATTING - used for setw
     * -----------------------------------------------------------------------
     * TITLE_COL : the first column that displays headings for the data
     *************************************************************************/

     const short TITLE_COL = 13;
```

```
52
53      /*************************************************************************
54      * VARIABLES
55      *************************************************************************/
56
57      ostringstream outOss;
58      string asStr;       // PROC & OUT  - type of assignment (LAB, ASSIGN, etc.)
59      short  asNumCol;    // CALC & FORM - column width for the assignment number
60                          //               specific to the type of assignment
61
62      /*************************************************************************
63      * PROCESSING: 1. Assigns the asStr( assignment string) based on the
64      * AS_TYPE (assignment type).
65      * 2. Assigns the asNumCol(assignment column width for the
66      * assignment number). The setw will format appropriately
67      * based on if this is a lab 'L' or assignment 'A'.
68      *************************************************************************/
69
70      asStr = "ASSIGNMENT";
71
72      if (toupper(asType) == 'L')
73      {
74          asStr = "LAB";
75      }
76
77      asStr += " #";
78
79      asNumCol = TITLE_COL - asStr.length();
80
81      /*************************************************************************
82      * OUTPUT - the class heading table
83      *
84      * ********************************************************************
85      * * PROGRAMMED BY : Blake Allard & Khaled Ajaj
86      * * CLASS         : CS1A
87      * * SECTION       : MW - 8am
88      * * LAB #14       : Rock, Paper, Scissors
89      * ********************************************************************
90      *
91      *************************************************************************/
92
93      outOss << left;
94      outOss << "*************************************************\n";
95      outOss << "* PROGRAMMED BY : Blake Allard & Khaled Ajaj\n";
96      outOss << "* "   << setw(TITLE_COL) << "CLASS"   << " : CS1A\n";
97      outOss << "* "   << setw(TITLE_COL) << "SECTION" << " : MW - 8am\n" ;
98      outOss << "* "   << asStr << setw(asNumCol) << asNum << " : ";
99      outOss << asName << endl;
100     outOss << "*************************************************\n\n";
101     outOss << right;
102
```

```
103      return outOss.str();
104
105 }
106
```

```cpp
/*****************************************************************************
 * AUTHOR     : Blake Allard , Khaled Ajaj
 * STUDENT ID : 358888       , 1125796
 * LAB #15    : Arrays & Files
 * CLASS      : CS1A
 * SECTION    : MW 8am
 * DUE DATE   : 11/27/24
 *****************************************************************************/

#include "header.h"

/*****************************************************************************
* FUNCTION OutputNamesAndCounts
*----------------------------------------------------------------
* This function outputs the table of names and instances of each name in the
* array of names. It then outputs the name not found that the user input
* into the program.
*
*----------------------------------------------------------------
* PRE-CONDITIONS
* The following parameters need to have a defined value
* prior to calling the function:
* namesArray: Array of names
* AR_SIZE : Size of array of names
* joeCount: instances of "Joe"
* sallyCount: instances of "Sally"
* adamCount: instances of "Adam"
* sueCount: instances of "Sue"
* nameNotFound: name not found in array
*
* POST-CONDITIONS
* This function will output a table of names and their instances.
*****************************************************************************/

string OutputNamesAndCounts(string namesArray[],  // IN     - array of names
                            int    AR_SIZE,       // IN     - size of the array
                            int    joeCount,      // OUT    - count of "Joe"
                            int    sallyCount,    // OUT    - count of "Sally"
                            int    adamCount,     // OUT    - count of "Adam"
                            int    sueCount,      // OUT    - count of "Sue"
                            string nameNotFound)  // OUT    - name not found
{


    // FORMATTING - used for setw
    const int PROMPT_COL = 19;

    // OUTPUT - outputs results to console and output file
    ostringstream outOss;

    // OUTPUT - the table headers and separation lines
```

```
52      outOss << endl;
53      outOss << left;
54      outOss << "NAME" << right << setw(PROMPT_COL) << "INSTANCES" << endl
55             << "----"          << setw(PROMPT_COL) << "---------" << endl;
56
57      // OUTPUT - the table of names and their instances
58      outOss << "Joe"   << setw(PROMPT_COL - 7) << joeCount   << endl
59             << "Sally" << setw(PROMPT_COL - 9) << sallyCount << endl
60             << "Adam"  << setw(PROMPT_COL - 8) << adamCount  << endl
61             << "Sue"   << setw(PROMPT_COL - 7) << sueCount   << endl
62             << nameNotFound << setw(PROMPT_COL - 8) << "-" << endl;
63
64
65      return outOss.str();
66  }
67
```

```cpp
/*******************************************************************************
 * AUTHOR     : Blake Allard , Khaled Ajaj
 * STUDENT ID : 358888       , 1125796
 * LAB #15    : Arrays & Files
 * CLASS      : CS1A
 * SECTION    : MW 8am
 * DUE DATE   : 11/27/24
 ******************************************************************************/

#include "header.h"

/*******************************************************************************
* FUNCTION ReadInputFile
*------------------------------------------------------------------
* This function passes in an array and a constant array size in order to be
* used for reading in data from an input file.
*
*------------------------------------------------------------------
* PRE-CONDITIONS
* The following parameters need to have a defined value
* prior to calling the function:
* namesArray: Array of names
* AR_SIZE : Size of array of names
* &fin: file input
*
* POST-CONDITIONS
* This function will output a table of names and their instances.
*******************************************************************************/

void ReadInputFile(string    namesArray[],     // IN - array of names
                   int       AR_SIZE,          // IN - size of the array
                   ifstream &fin)              // IN - read file input
{
    int index = 0;
    string name;

    //PROCESSING - write name input to file
    while (fin && index < AR_SIZE)
    {
        getline(fin, name);
        namesArray[index] = name;

        index++;
    }

}

```

```
 1 Joe
 2 Sally
 3 Joe
 4 Sue
 5 Sally
 6 Adam
 7 Joe
 8 Adam
 9 Adam
10 Joe
```