COP 3503 Summer 2014
Programmer's Guide and Report for project 2

David Campbell
Tues June 17 EDT 2014

Purpose of Program
Project 2 deals with reading and writing text files and different types of multiset operations

Options
0. exit
1. input file <filename>: open and read a list from a file to the current multiset
2. union file <filename>: open and union a multiset from a file with the current multiset
3. subtract file <filename>: open and subtract multiset from a file from the current multiset
4. difference file <filename>: open and find the difference between a multiset from a file and the current multiset
5. intersect file <filename>: open and find the intersection between a multiset from a file and the current multiset
6. reset current multiset to the empty multiset
7. output file <filename>: open and write the current multiset to a file
8. print current multiset to the console
9. find <item name>: test if <item name> is in the current multiset
10. insert <item name> <count>: add <item name> to the current multiset with number <count> if <item name> is not in the current multiset, or increase <item name>'s number by <count> if it is
11. delete <item name>: remove <item name> from the current multiset if it is in it
12. reduce <item name> <count>: reduce the number of <item name> by <count>
13. verbose output
14. normal output
15. silent output
16. help
17. max <filename>: open and find the maximum between a multiset from a file and the current multiset.

Organization of Code
Within the main function are the options above which are selected using a case switch. Each option calls a method(s) that performs the required function. There are 3 list variables that are used. A single int variable keeps track of option 12, 13, and 14.

Functions, Methods, Procedures
`void printTheMenu()` prints a basic menu.
`void help()` prints a more descriptive menu
`void readListFromFile(int);`The function reads a file into a vector and depending on the choice, performs an operation on that vector. If choice = 0, the function reads a file into a vector. If choice = 1, the function subtracts the new vector from the current vector. If choice = 2, the function gets the difference of the new vector from the current vector. If choice = 3, the union

function is used. This function uses 3 lists to get the intersection of two multisets. The new multiset is copied into vector 2 and each element of the original vector is compared to the second vector using a nested for loop. If an element exists in both vectors, it is copied to a 3rd vector. Finally, that 3rd vector is copied into the original vector and the other two vectors are cleared. The output mode variable just determines whether the output is silent, normal, or verbose.

`void testFile(string);` tests the file and outputs errors if any are found
`bool readListFromFile(string);` The function reads a file into a vector and outputs true if successful.
`void unionSet();`This function performs the union operation on the multiset
`void saveToFile();`This function simply saves the current vector to a file based on the file name the user provides
`void find();`This function tests if an element exists by iterating through the current vector using a for loop and testing whether or not the user-supplied search string exists in that vector.
`void insert();` This function just adds a new element to the current vector using the push_back function.
`void clear();`This function empties the multiset from memory.
`oid print();`This function prints the multiset to the console.
`void deleteFromList();` This function removes an element from the current vector using the erase vector function.
`void reduce();`This function reduces the count of an item in a multiset.
`void max();`This function is takes the max count value of the same element exists in two multisets.
`int main(int argc, char ** argv)` This function mainly houses the large case switch which is used to call the different required methods based on the option the user enters.

Efficiency
The code is pretty compact. The code that is used to add the contents of a file to a list is used a lot, but it is a little different each time.

Known Bugs
I found and fixed a few bugs. I'm sure there are others but I didn't find any.

Testing
Testing was performed by running through the same commands shown in the example output and comparing the outputs.

Available Commands:

Available Commands:
0. Exit
1. input file <filename>: open and read a list from a file to the current list
2. union file <filename>: open and union a set from a file with the current set
3. subtract file <filename>: open and subtract set from a file from the current set
4. difference file <filename>: open and find the difference between a set from a file and the current set
5. intersect file <filename>: open and find the intersection between a set from a file and the current set

6. reset current set to the empty set
7. output file <filename>: open and write the current set to a file
8. print current set to the console
9. find <item name>: test if <item name> is in the current set
10. insert <item name>: add <item name> to the current set if it is not already
in it
11. delete <item name>: remove <item name> from the current set if it is in it
12. reduce <item name> <count>: reduce the number of <item name> by <count>
13. verbose output
14. normal output
15. silent output
16. help
17. max <filename>: open and find the maximum between a multiset from a file and
the current multiset.
> 1 file 2 union
New multiset loaded.
> New multiset loaded.
Union completed.
> 8
Current multiset:
five 2
four 3
three 2
> 1 file 3 subtract 8
New multiset loaded.
> Subtraction completed.
> Current multiset:
one 2
> 1 file 4 diff 8
New multiset loaded.
> Difference completed.
> Current multiset:
four 4
one 2
three 2
> 1 file 5 diff 8
New multiset loaded.
> Intersection completed.
> Current multiset:
one 1
three 1
two 2
> 6
Reset completed
> 8
Current multiset:
> 1 file 9 one
New multiset loaded.
> Item one found with count 3.
> 9 four
Item four not found.
> 10 five 5
five inserted with count 5.
> 10 one 5
one inserted with count 8.
> 11 five
Item five deleted.
> 13

0. Exit
1. input file
2. union file

3. subtract file
4. difference file
5. intersect file
6. reset current set
7. output file
8. print current set
9. find
10. insert
11. delete
12. reduce
13. verbose output
14. normal output
15. silent output
16. help
> 1 file 2 union 8
New multiset loaded.

0. Exit
1. input file
2. union file
3. subtract file
4. difference file
5. intersect file
6. reset current set
7. output file
8. print current set
9. find
10. insert
11. delete
12. reduce
13. verbose output
14. normal output
15. silent output
16. help
> New multiset loaded.
Union completed.

0. Exit
1. input file
2. union file
3. subtract file
4. difference file
5. intersect file
6. reset current set
7. output file
8. print current set
9. find
10. insert
11. delete
12. reduce
13. verbose output
14. normal output
15. silent output
16. help
> Current multiset:
five 2
four 3
three 2

0. Exit
1. input file
2. union file

3. subtract file
4. difference file
5. intersect file
6. reset current set
7. output file
8. print current set
9. find
10. insert
11. delete
12. reduce
13. verbose output
14. normal output
15. silent output
16. help
> 14
> 16

Available Commands:
 0. Exit
 1. input file <filename>: open and read a list from a file to the current list
2. union file <filename>: open and union a set from a file with the current set
3. subtract file <filename>: open and subtract set from a file from the current
set
4. difference file <filename>: open and find the difference between a set from a
 file and the current set
5. intersect file <filename>: open and find the intersection between a set from
a file and the current set
6. reset current set to the empty set
7. output file <filename>: open and write the current set to a file
8. print current set to the console
9. find <item name>: test if <item name> is in the current set
10. insert <item name>: add <item name> to the current set if it is not already
in it
11. delete <item name>: remove <item name> from the current set if it is in it
12. reduce <item name> <count>: reduce the number of <item name> by <count>
13. verbose output
14. normal output
15. silent output
16. help
17. max <filename>: open and find the maximum between a multiset from a file and
 the current multiset.
> 1 file 17 diff 8
New multiset loaded.
> New multiset loaded.
Max completed.
> Current multiset:
four 4
one 1
three 3
two 2
> 15
1 file 2 union
8
Current multiset:
five 2
four 3
three 2