Sabancı University

Faculty of Engineering and Natural Sciences
CS204 Advanced Programming
Spring 2016

Homework 3 – Counting Words using Doubly Linked Lists

Due: 9/3/2016, Wednesday, 21:00

PLEASE NOTE:

Your program should be a robust one such that you have to consider all relevant programmer mistakes and extreme cases; you are expected to take actions accordingly!

You can NOT collaborate with your friends and discuss solutions. You have to write down the code on your own. Plagiarism will not be tolerated!

Introduction

In this homework, you are asked to implement a 2-dimensional storage to store words of a text file and their occurrences. The storage structure must use *Doubly Linked Lists*. In order to store the words with the same initial letter, you will have one doubly linked list so that you have to employ one double linked list for each letter of the alphabet. Moreover, your program should perform several operations. The details will be explained in this homework specification.

The Program Flow

At the very beginning, the program prompts a for a text file name. The program asks the name of the input file until a correct file name is given. This file will contain words separated with one or more space characters. The words might contain punctuation marks. A sample input file is given below.

```
He mechanically set about making the
preparations for departure. Around the world
in eighty days!  Was his master a fool?  No.
Was this a joke, then?  They were going to
Dover; good!  To Calais; good again! After
all, Passepartout, who had been away from
France five years, would not be sorry to set
foot on his native soil again.  Perhaps they
would go as far as Paris, and it would do his
eyes good to see Paris once more.
```

Figure 1. Sample input file

Your program should scan this file word by word, and it should also make two modifications on the word being processed. The first modification is that the word should be stripped of leading and trailing punctuation marks (you can use `StripPunc` function of strutils for this purpose). The second modification is to convert all the letters in the word to lowercase in order to process the words in case-insensitive manner (you can use `ToLower` function of strutils for this purpose). Then, the initial letter of the resulting string (word) should be checked. If this initial character is a letter in the English alphabet, the word should be accepted to be stored and counted. Otherwise, meaning that the word contains a non-letter initial character, it should be discarded. You are required to print a message for every word that has been discarded in the process. Please check the sample runs for clarification. For each valid word, it is checked whether it exists in the data structure or not. If exists its count must be incremented. If not, you have to create a new node for the word. After all words of the file are processed, we will have one node for each unique word in the data structure that also contains its number of occurrence. The details of how to store the words are laid out in **The Data Structure** section.

After the file has been processed, the program prompts a menu. The menu provides 7 different options and each can be chosen in any order by the user. When an option is entered, the corresponding operation is performed and then the menu is displayed again. This continues until the user exits. The menu options and the corresponding operations are explained below.

1. **Display the count of a specific word**

In this option, the program reads a word in order to display its count (total number of occurrence) by searching the corresponding node in the data structure. Before the search, the input word must be processed as described in the previous section (`StripPunc` and `ToLower`). Then, the first character of the word will be checked and if it is not a letter of English alphabet an appropriate error message will be displayed. If the word starts with a letter, then it will be searched in the data structure. In the case that the word is found, it will be displayed along with its count (how many times it occurred in the text). A message should be printed when the word entered does not appear in the data structure.

2. **Remove a specific word**

In this option, the program asks for a word to remove. The word will be processed as described before. Then, the first character of the word will be checked and if it is not a letter of English alphabet an appropriate error message will be displayed. If the word starts with a letter, then it will be searched in the data structure. In the case that the word is found, it will be removed from the data structure. A message should be printed when the word entered does not appear.

3. **Display words with the same initial letter (in alphabetical order)**

This option gets a letter as input from the user. If the user enters more than one character, your program should display an error message. If the user enters a non-letter, your program should also display an error message. The user may enter either lowercase or uppercase letter and they should work the same. After these checks, if the user enters a proper letter, then the program displays all the words beginning with this letter and their counts, in alphabetical order. If there are no words beginning with the letter, an informing message should be displayed.

4. **Display words with the same initial letter (in reverse alphabetical order)**

This option works the same as option 3, but this time you have to display in reverse alphabetical order.

5. **Display all words (in alphabetical order)**

This option prints all the words and their counts in alphabetical order.

6. **Display all words (in reverse alphabetical order)**

This option prints all the words and their counts in reverse alphabetical order.

7. **Exit**

Program terminates when the user chooses this option. Your program should deallocate all the dynamic memory before termination.

The Data Structure

In this homework, you must store the words that have the same initial letter as one *doubly linked list*. Thus you have to use 26 doubly linked lists in the program.

The node type of the doubly linked list should have two node pointers: *next*, *previous*. These pointers are pointing to the previous and next words in the word list, respectively. Nodes must be ordered according to alphabetical order. In addition, the node structure should contain a string for the word and an integer for its count. For the sake of standardization, please use the following `struct` for the nodes. Of course, if you want you can add constructors for this `struct`.

```
struct node
{
     string word;
     int count;
     node * next;
     node * previous
}
```

You have to design and implement a class for all of the data and functions of this doubly linked list. Below is a reference implementation which you can make use of. You can add or remove methods from this class, if you need to do so. However, please do not forget that this is a doubly linked list and you need to keep both head and tail.

```
class wordlist
{
public:
     wordlist(); //constructor
     void AddWord(string);
     void RemoveWord(string);
     void DisplayWord(string);
     void PrintList();
     void ReversePrintList();
     void DeleteList();
     bool isEmpty();

private:
     node * head;
     node * tail;
};
```

Since one wordlist object is used to store words with the same initial letter, we need 26 of them. For this purpose please use a **dynamic array of wordlist objects with 26 elements**. For example, `wordlist * myWords  = new wordlist [26];`

No other multiple data containers (built-in array, dynamic array, vector, matrix, 2D array, etc.) can be used. The only exception is the wordlist array that is mentioned above.

An example of such structure is showed in Figure 2 for the data provided in Figure 1.

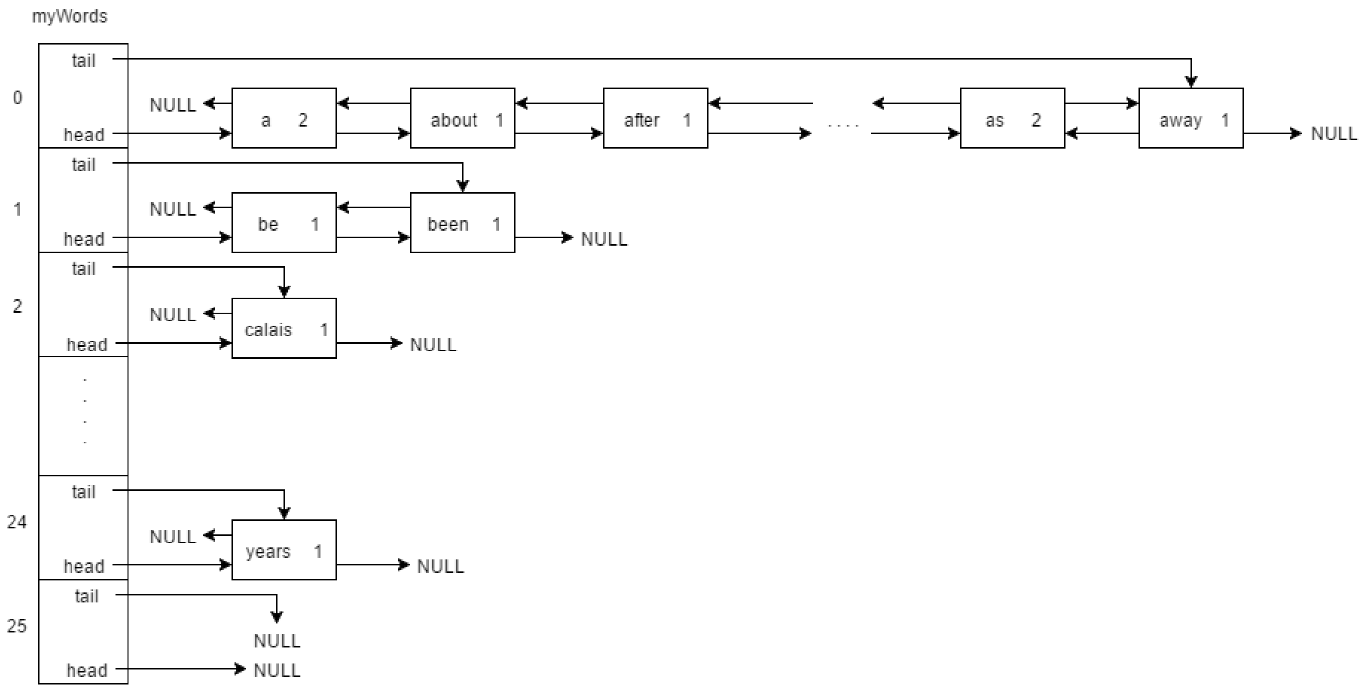

Figure 2. The Data Structure Sample

Sample Runs

Some sample runs are given below, but these are not comprehensive, therefore you have to consider **all possible cases** to get full mark.

Sample Run 1
File: file1.txt

```
I see a beautiful city and a brilliant
people rising from this abyss. I see the
lives for which I lay down my life,
peaceful, useful, prosperous and happy. I
see that I hold a sanctuary in their hearts,
and in the hearts of their descendants,
generations hence. It is a far, far better
```

```
thing that I do, than I have ever done; it
is a far, far better rest that I go to than
I have ever known.
```

```
Please enter the name of the input file: file.txt
File cannot be opened.
Please enter the name of the input file again: file1.txt
1ay cannot be added
1ife, cannot be added
File file1.txt has been read and processed

---------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 5
*******************************************
a 5
abyss 1
and 3
beautiful 1
better 2
brilliant 1
city 1
descendants 1
do 1
done 1
down 1
ever 2
far 4
for 1
from 1
generations 1
go 1
happy 1
have 2
hearts 2
hence 1
hold 1
i 9
in 2
is 2
it 2
known 1
lives 1
my 1
of 1
peaceful 1
people 1
prosperous 1
rest 1
rising 1
sanctuary 1
see 3
than 2
that 3
the 2
their 2
```

```
thing 1
this 1
to 1
useful 1
which 1

--------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 1

Please enter a word to find out its count: I
******************************************
i 9

--------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 1

Please enter a word to find out its count: little
******************************************
little does not exist.

--------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 1

Please enter a word to find out its count: aBySs
******************************************
abyss 1

--------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 2

Please enter a word to remove: hello
******************************************
```

```
hello does not exist.

---------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 2

Please enter a word to remove: 1rising
******************************************
The word contains an invalid first letter

---------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 2

Please enter a word to remove: rising
******************************************
rising has been deleted.

---------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 3

Please enter a letter: R
*******************************************
rest 1

------------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 2

Please enter a word to remove: REST
******************************************
rest has been deleted.

---------------------------------------------------------------
Please select one option:
```

```
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 6
*******************************************
which 1
useful 1
to 1
this 1
thing 1
their 2
the 2
that 3
than 2
see 3
sanctuary 1
prosperous 1
people 1
peaceful 1
of 1
my 1
lives 1
known 1
it 2
is 2
in 2
i 9
hold 1
hence 1
hearts 2
have 2
happy 1
go 1
generations 1
from 1
for 1
far 4
ever 2
down 1
done 1
do 1
descendants 1
city 1
brilliant 1
better 2
beautiful 1
and 3
abyss 1
a 5

--------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 7
Press any key to continue . . .
```

Sample Run 2
File: file2.txt

```
Batman hurries off. Limping into the shadows.

GORDON'S SON: Batman? Batman! Why is he running, Dad?

GORDON: Because we have to chase him.

As Cops race into the buildings the dogs get the scent
and pull away from the doorway, following the shadow
into the stacks of shipping containers...

James looks at his father, confused.

SON: He didn't do anything wrong.

Gordon stares after the Batman. The sound of the dogs
becoming louder and more ferocious.

GORDON: Because...

The Batman LURCHES between shipping containers.
Stumbling. Bleeding. He makes it to the bat-pod...

GORDON: ...he's the hero Gotham deserves, but not the
one it needs right now. So, we'll hunt him, because he
can take it. Because he's not our hero.

The bat-pod streaks through Gotham's underground
streets, the Batman's cape fluttering behind. A
wraith...

GORDON (cont'd): He's a silent guardian. A watchful
protector. A Dark Knight.

The Batman races up a ramp into a blinding light
```

```
Please enter the name of the input file: file2.txt
File file2.txt has been read and processed

------------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 1

Please enter a word to find out its count: batman
******************************************
```

```
batman 6

-----------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 3

Please enter a letter: b
*******************************************
bat-pod 2
batman 6
batman's 1
because 4
becoming 1
behind 1
between 1
bleeding 1
blinding 1
buildings 1
but 1

-----------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 2

Please enter a word to remove: 1bat-pod3
******************************************
The word contains an invalid first letter

-----------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 2

Please enter a word to remove: bat-pod3
******************************************
bat-pod3 does not exist.

-----------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
```

```
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 2

Please enter a word to remove: bat-pod
******************************************
bat-pod has been deleted.

-----------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 4

Please enter a letter: B
*******************************************
but 1
buildings 1
blinding 1
bleeding 1
between 1
behind 1
becoming 1
because 4
batman's 1
batman 6

-----------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 2

Please enter a word to remove: BUT
******************************************
but has been deleted.

-----------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 4

Please enter a letter: b
*******************************************
buildings 1
blinding 1
bleeding 1
```

```
between 1
behind 1
becoming 1
because 4
batman's 1
batman 6

-----------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 6
*******************************************
wrong 1
wraith 1
why 1
we'll 1
we 1
watchful 1
up 1
underground 1
to 2
through 1
the 17
take 1
stumbling 1
streets 1
streaks 1
stares 1
stacks 1
sound 1
son 2
so 1
silent 1
shipping 2
shadows 1
shadow 1
scent 1
running 1
right 1
ramp 1
races 1
race 1
pull 1
protector 1
our 1
one 1
off 1
of 2
now 1
not 2
needs 1
more 1
makes 1
lurches 1
louder 1
looks 1
limping 1
light 1
knight 1
james 1
it 3
```

```
is 1
into 4
hurries 1
hunt 1
his 1
him 2
hero 2
he's 3
he 4
have 1
guardian 1
gotham's 1
gotham 1
gordon's 1
gordon 5
get 1
from 1
following 1
fluttering 1
ferocious 1
father 1
doorway 1
dogs 2
do 1
didn't 1
deserves 1
dark 1
dad 1
cops 1
containers 2
cont'd 1
confused 1
chase 1
cape 1
can 1
buildings 1
blinding 1
bleeding 1
between 1
behind 1
becoming 1
because 4
batman's 1
batman 6
away 1
at 1
as 1
anything 1
and 2
after 1
a 6

-----------------------------------------------------------------
Please select one option:
1. Display the count of a specific word
2. Remove a specific word
3. Display words with the same initial letter (in alphabetical order)
4. Display words with the same initial letter (in reverse alphabetical order)
5. Display all words  (in alphabetical order)
6. Display all words  (in reverse alphabetical order)
7. Exit.

Your choice: 7
Press any key to continue . . .
```

Some Important Rules

In order to get a full credit, your programs must be efficient and well presented, presence of any redundant computation or bad indentation, or missing, irrelevant comments are going to decrease your grades. You also have to use understandable identifier names, informative introduction and prompts. Modularity is also important; you have to use functions wherever needed and appropriate. Since using classes is mandated in this homework, a proper object oriented design and implementation will also be considered in grading.

Since you will use dynamic memory allocation in this homework, it is very crucial to properly manage the allocated area and return the deleted parts to the heap whenever appropriate. Inefficient use of memory may reduce your grade.

When we grade your homework we pay attention to these issues. Moreover, in order to observe the real performance of your codes, we may run your programs in *Release* mode and **we may test your programs with very large test cases**. Of course, your program should work in *Debug* mode as well.

What and where to submit (PLEASE READ, IMPORTANT)

You should prepare (or at least test) your program using MS Visual Studio 2012 C++. We will use the standard C++ compiler and libraries of the abovementioned platform while testing your homework. It'd be a good idea to write your name and last name in the program (as a comment line of course).

Submissions guidelines are below. Some parts of the grading process are automatic. Students are expected to strictly follow these guidelines in order to have a smooth grading process. If you do not follow these guidelines, depending on the severity of the problem created during the grading process, 5 or more penalty points are to be deducted from the grade. Name your solution, project, cpp file that contains your main program using the following convention (the necessary file extensions such as .sln, .cpp, etc, are to be added to it):

“SUCourseUserName_YourLastname_YourName_HWnumber”

Your SUCourse user name is actually your SUNet user name which is used for checking sabanciuniv e-mails. Do NOT use any spaces, non-ASCII and Turkish characters in the file name. For example, if your SUCourse user name is cago, name is Çağlayan, and last name is Özbugsızkodyazaroğlu, then the file name must be:

Cago_Ozbugsizkodyazaroglu_Caglayan_hw3

In some homework assignments, you may need to have more than one .cpp or .h files to submit. In this case add informative phrases after the hw number. However, do not add any other character or phrase to the file names.

Now let us explain which files will be included in the submitted package. Visual Studio 2012 will create two *debug* folders, one for the solution and the other one for the project. You should delete these two *debug* folders. Moreover, if you have run your program in release mode, Visual Studio may create *release* folders; you should delete these as well. Apart from these, Visual Studio 2012 creates a file extension of *.sdf*; you will also delete this file. The remaining content of your solution folder is to be submitted after compression. Compress your solution and project folders using WINZIP or WINRAR programs. Please use "zip" compression. "rar" or another compression mechanism is NOT allowed. Our homework processing system works only with zip files. Therefore, make sure that the resulting compressed file has a zip extension. Check that your compressed file opens up correctly and it contains all of the solution, project and source code files that belong to the latest version of your homework. Especially double-check that the zip file contains your cpp and (if any) header files that you wrote for the homework.

Moreover, we strongly recommend you to check whether your zip file will open up and run correctly. To do so, unzip your zip file to another location. Then, open your solution by clicking the file that has a file extension of .sln. Clean, build and run the solution; if there is no problem, you could submit your zip file. Please note that the deleted files/folders may be regenerated after you build and run your program; this is normal, but do not include them in the submitted zip file.

You will receive no credits if your compressed zip file does not expand or it does not contain the correct files. The naming convention of the zip file is the same. The name of the zip file should be as follows:

SUCourseUserName_YourLastname_YourName_HWnumber.zip

For example zubzipler_Zipleroglu_Zubeyir_hw3.zip is a valid name, but

Hw3_hoz_HasanOz.zip, HasanOzHoz.zip

are **NOT** valid names.

Submit via SUCourse ONLY! You will receive no credits if you submit by other means (e-mail, paper, etc.).

Successful submission is one of the requirements of the homework. If, for some reason, you cannot successfully submit your homework and we cannot grade it, your grade will be 0.

Good Luck!

Albert Levi, Ömer Mert Candan