**Design Document**

1. Title  
   Informations’ Database , Melanie Woe, 105557790, 4/5/2016.
2. Problem Description  
   This program is a simple database system which handles multiple records. A database is a collection of information that one can update, organize, sort, search through, and print as needed. This program will store its information to a file, addition and deletion of records, field modifications, and it will allow users to sort records based on the selected keys, and produce reports (output) according to predefined criteria. This program will have a database for an address book, in which, the users should be able to browse their contacts by any field from any database and organize their chosen contacts to their liking and print out a report.
3. Overall Software Architecture

The program will use a text-based menu for users to choose available features and it will have a main menu at the beginning and also sub menus.

The users of the program will be able to read and write data to a text-based database file with the format (information is stored In ASCII text format, | divides the records, field is distinguished by line)

The program will test the input file when reading it to make sure that the data is of valid format and the contacts data will be inserted into the avl tree.

The program is primarily organized with ID# as a key. (AVL tree is the base data structure).

The program will provide appropriate user prompts and on-screen directions.

In this program, it will have a main function where it will have a menu where the users will be able to choose from. And then it will read in and output data using ifstream and ofstream.

The program uses vector to store the data. vector<ItemType> myvector;

I chose to use vector because elements in the vector are guaranteed to be stored in contiguous storage locations in the same order as represented by the vector. And it is easier to be accessed.

The same for the affiliates, I also use vector.

It will use Avltree.h and Avltree.cpp for basic data base creation.

Then records.h and records.cpp where it will have the setID, setfirstname, setlastname, setcompanyname, set homephone, etc. and then it will also return them (getId, getfirstname, etc.)

And Database.h and database.cpp where it will give the functions of deleting the record, modify the record, insert new record. (This function will uses the data from the records)

Edit data

Sort file data

Read in the file data (string types)

Display data

Insert the file data into AVL tree

Search data

Add data

1. Input Requirements

The program will have an input file format such as:

ID#, first name, middle name, last name, company name, home phone, office phone, email, mobile number, street address, city, state, zip code, country, Affiliate name 1, Affiliate name 2, Affiliate name 3, and Affiliate name 4.

The user will choose of the options and they should be able to search records based on the field information in two modes: exact and contains. For example search by first name “Jacob”, and then in the sub menu, users have to pick the search mode and field. And the user will have to type in the search item.

Data type :

Everything will be in string type.

String Id (9 digits), string first name, string middle name, string last name, string company name, string home phone, string office phone, string email, string mobile number, string address, string city, string state, string zip code, string country, string affiliates.

| = will be the record divider

A detailed list of all external inputs (from files or keyboard) including a description of the data type and range of valid values for each input. For input file format and interactive user input, you need to write what data type is used for every field and valid value and length.

1. Output Requirements

The output generation will be responsible for the re-organizing the final result to a user’s liking by displaying the fields and also sorted records. The output will be in ASCII-text format. There will be a list of contacts as output. And from the list of contacts, users will be able to further organize them and convey them to the output by choosing only the fields that they want to print in desired order. Secondary sorting will also be performed with the second sort key. The output file generation in text file format (ASCII text format).

Ouput: string id, string first name, string middle name, string last name, string company name, string home phone, string office phone, string email, string mobile number, string address, string city, string state, string zip code, string country, string affiliates.

My program’s result:

The program compiles and runs fine although it has few warnings and it outputs all the choice menu. I think that the programs cannot insert the database.txt to file and that’s why I cannot decide if my other codes for sorting, deleting, and adding works or not. But I am pretty sure that my “adding record” code can work because I tried to add the names.

1. Problem Solution Discussion

The program uses vector to store the data for affiliates.

vector<ItemType> myvector;

I chose to use vector because elements in the vector are guaranteed to be stored in contiguous storage locations in the same order as represented by the vector. And it is easier to be accessed.

It will use Avltree.h and Avltree.cpp for basic data base creation. AVL tree is the basic data structure to store data.

1. Data Structures

As in the project description, we need to use the AVL tree as our basic data structure for database. String will be the type of the records.

1. User Interface Scheme

The menu will be like this:

Please choose one of the option below by entering the number:

"1. Display all contacts”

"2. Sort and display all”

"3. Search record

"4. Modify Contact"

"5. Save Report"

"6. Add record"

"7. Exit"

You choose number : \_

After that, there will be more sub menu