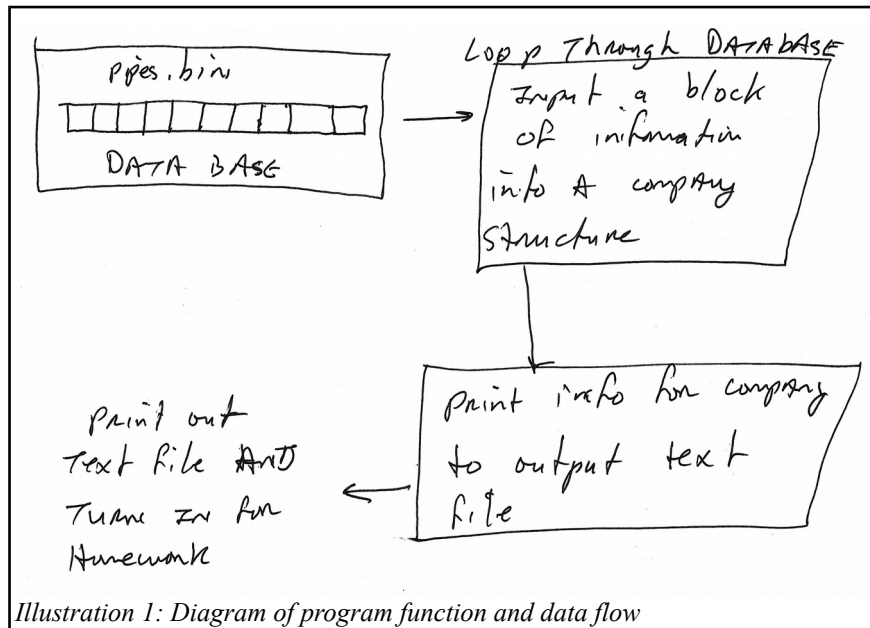


The contract called for creation of a random access database of plumbing shops within the near perimeter of FIU Engineering school. The database features a rating number from 1-10 to offer a guideline to the quality of service provided by the company.



Refer to the program outline shell program at the end of these instructions. Your assignment is to take that shell program and complete the code to read the company names from the pipes.bin database and print them to a text file. Once the output text file is created, it can be loaded into an editor and printed to be turned in as completion of this homework.

Create a CodeBlocks project to build a program to open the plumbing database and print each data block to a text file.

The database is located at <http://web.eng.fiu.edu/watsonh/EEL2880/pipesDatabase.html>

Download and save the binary database in the project folder you created for the program. When the program is compiled and executed from within CodeBlocks (Run), the default directory for the data file is where the main.cpp code is located.

The steps are as follows:

1. Create CodeBlocks project (C++ console application)
2. Download the binary database file to project folder
3. Copy the skeleton code to the project source file (main.cpp)
4. Fill in and complete the blanks for the source code
5. Compile and build the 'plumbing' program
6. Run the program: use the database as the input file (pipes.bin) and output to a text file

Macros:

Complete the lines of code to read the plumbing database using the following features:

Define in the program that there are three defined string lengths (lines 5-7)

```
STRSHORT = 15 chars
STRMED = 30 chars
STRLONG = 80 chars
```

Each block is location and contact information for a single plumbing company. Consider it similar to having a single index card for each company with the database being a collection of 10 cards.

The idea is to retrieve a single block from the database at a time and output all the information for each card to the output text file. Once all 10 blocks have been processed, all files can be closed. The output text file can be opened with an editor, printed, and turned in for the assignment.

Define data structure:

The structure for each company data block has the following members in the exact sequence (line 12):

```
char name [STRLONG];
char street [STRLONG];
char city [STRMED];
char state [STRMED];
char zip [STRSHORT];
char phone [STRSHORT];
char cell [STRSHORT];
int rating;
char contact [STRLONG];
```

Function Prototype:

One user defined function in the program has to have a prototype (line 15) as follows:

```
void printInfo(company* info, FILE* fp);
```

Declare file pointers:

Inside the program main function, two file pointers need to be declared: Input database file pointer line 19 and Output text file pointer line 20.

Declare data structure:

Declare an instance of the generic company data structure line 25 used to hold read file contents. Use the 'new' operator creating a block of memory returning a pointer to the block. The new operator is a C++ operator, so the project needs to be created as a C++ project if this is used. The pointer is named and is used throughout the program to reference the structure.

```
company* info= new company;
```

If the project is created as a C language project, then the following C language function is used

```
company* info = malloc(sizeof(company));
```

Line 8 would also have to be removed if a C language project is created. (remove: 8 using namespace std;)

Open input database file: fopen

Lines 27-29 prompt the user for the input file name. A char array for the input file name needs to be declared before it is used. The array can be declared with a long string constant [STRLONG]. The declaration should be placed at line 22. That input file name is then used on line 32 where the file is to be opened.

The 'if' expression should look like the following and specify read binary :

```
if ((file pointer = fopen (calling arguments)) == NULL)
```

Line 32 uses the fopen function http://www.acm.uiuc.edu/webmonkeys/book/c_guide/2.12.html#fopen .

Fill in the appropriate values used in the 'if' condition expression. Since the pipes database file is a binary database, the fopen specification should be to 'read binary' mode .

Open output text file: fopen

Lines 38-40 prompt the user for the output text file name. A char array for the output file name needs to be declared before it is used. The array can be declared with a long string constant [STRLONG]. The declaration should be placed at line 23. That output file name is then used on line 43 where the file is to be opened.

The 'if' expression should again look like the following except now specify write text:

```
if ((file pointer = fopen (calling arguments)) == NULL)
```

Use the fopen function referenced above to open the output text file. Fill in the appropriate values used in the 'if' condition expression. Since the output text file is composed of strings, the fopen specification should be set to 'write text' mode.

Line 48 sets up a loop to read each company information block one-at-a-time from the beginning of the file to the end. Each time a block is read, the contents are printed to the output text file.

Loop through the database:

First the database file needs to be positioned to the block corresponding to the current BlockNumber. In order to do that, the file is positioned to the block using fseek using the offset.

Line 51 is the 'if' statement used for fseek to position the file. The rest of the 'if' statement detects a failed seek condition and exits the program (lines 52-55).

Seek to data block:

The specification for the fseek function (other than the text book) can be found at

http://www.acm.uiuc.edu/webmonkeys/book/c_guide/2.12.html#fseek

Fill in the appropriate values used in the 'if' condition expression.

The 'if' expression should look like the following, but specify the condition expression
`if ((fseek(correct calling arguments), != 0)`

The input file stream returns the file pointer value when the file is successfully opened (line 32). The fseek offset is calculated from the beginning of the file, position 0 using the appropriate whence constant. The appropriate Block Number offset is computed as the BlockNumber*sizeof(company data structure).

Read the database block: fread

The specification for fread on line 57 function (other than the text book) can be found at

http://www.acm.uiuc.edu/webmonkeys/book/c_guide/2.12.html#fread

`fread(information structure pointer, sizeof(information structure), Number of blocks to read, input file pointer)`

Fread has 4 calling arguments, see the web reference or the text book. The first calling argument is the name of a pointer for the database information structure where the input data will be placed. The second calling argument has to specify how many bytes are to be read into the structure. Rather than counting the number of bytes, the computer will do that for you by using the function: sizeof(information structure). The third calling argument will be 1 (one) because only one block of data will be read. The last calling argument is the name of the file pointer for the input stream (line 19).

Print information for each company:

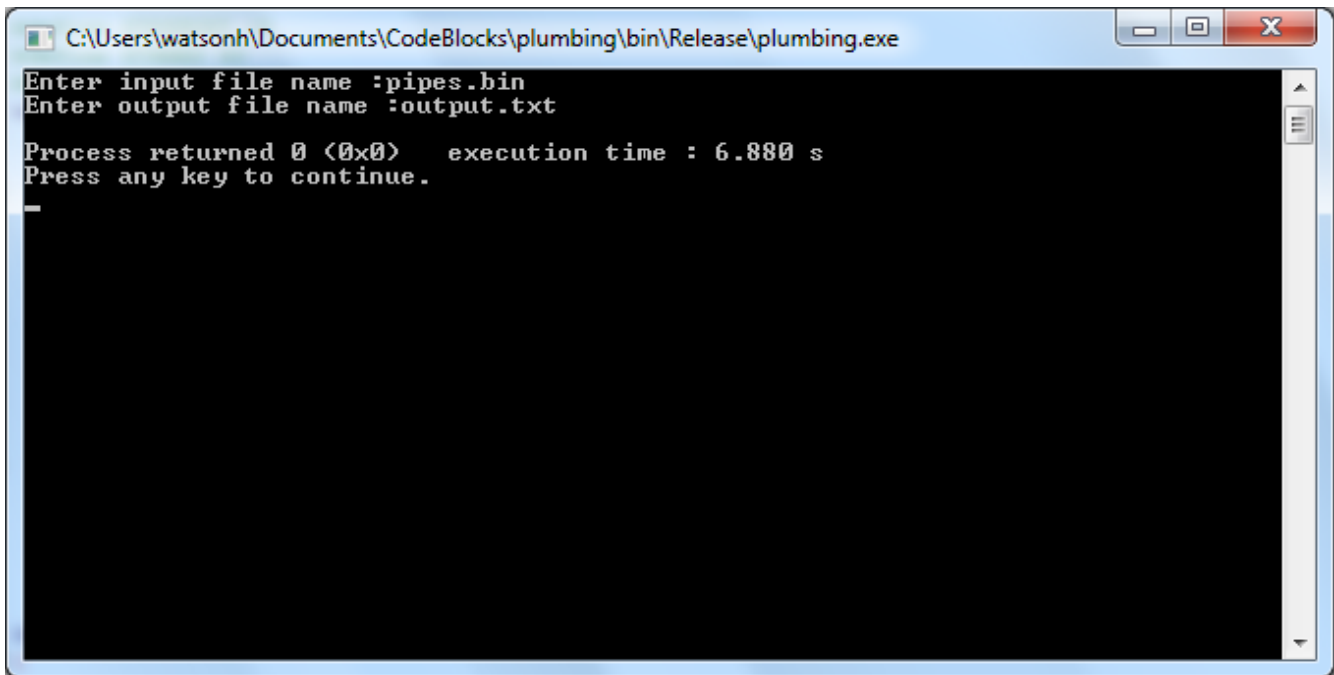
Call the user defined printInfo function line 60 with the appropriate calling arguments. Hint: both calling arguments are pointers.

End of loop through database**User defined printInfo function:**

Fill in the calling arguments on line 65. These will be pointers for the block of company information and the pointer for the output file where text is being written to.

Line 68 takes data from the company data structure to print company name to the output text file. Lines 69-76 need to be filled in with the corresponding statements to print each value of the company structure. Print out the rest of the information from the company data structure. Refer to the data structure declaration used on lines 10-13.

The following screen print is obtained from running the program.



```
C:\Users\watsonh\Documents\CodeBlocks\plumbing\bin\Release\plumbing.exe
Enter input file name :pipes.bin
Enter output file name :output.txt
Process returned 0 (0x0) execution time : 6.880 s
Press any key to continue.
-
```

Once this program has been run, there should be a file in the project folder with the name given for the output file name.

Open that output text file with an editor and print its contents on paper to turn in as the completion of the assignment.

Good luck team!

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 // set string lengths
5 #define ?? - see macros page 1
6 #define ??
7 #define ??
8 using namespace std; - created as a C++ CodeBlocks project
9
10 typedef struct company
11 {
12 ?? add member elements for structure from page 2
13 }company;
14
15 ?? function prototype to print information: printInfo (line 65)
16
17 int main()
18 {
19     declare input file pointer ??
20     declare output file pointer ??
21     int BlockNumber;
22     char ??[STRLONG]; //create an input file name (think up a name)
23     char ??[STRLONG]; //create an output file name
24
25     ?? Declare data structure:page 2 – use either new or malloc()
26
27     printf("Enter input file name :");
28     scanf("%s", input file name );
29     fflush(stdin); /* flush keyboard buffer */
30
31     // open binary data file
32     if ( fill in the fopen expression to open the database file )
33     {
34         fprintf(stderr, "Error opening input file.");
35         exit(1);
36     }
37
38     printf("Enter output file name :");
39     scanf("%s", output file name );
40     fflush(stdin); /* flush keyboard buffer */
41
42     // open output text file
43     if ( fill in the fopen expression to open the output text file page 2 )
44     {
45         fprintf(stderr, "Error opening output file.");
46         exit(1);
47     }
48     for (BlockNumber=0; BlockNumber<10; BlockNumber++)
49     {
50         /* Move the position indicator to the specified element. */
51         if ( fill in the fseek expression to position to BlockNumber page 3 )
52         {
53             fprintf(stderr, "\nError using fseek().");
54             exit(1);
55         }
56         /* Read in a single info. */
57         fread( fill in the fread expression to read data page 3);
58
59         fprintf(op, "\n\n\n Block %d\n\n", BlockNumber); /add header text
60         printInfo (fill in correct calling arguments);
61     }
62     return 0;
63 }
64
65 void printInfo(fill in calling arguments – company and output file pointers)
66 {
67     // print all the info values
68     fprintf (fp, "Name: %s \n", info->name);
69     fprintf ( fill in correct arguments );
70     fprintf ( fill in correct arguments );
71     fprintf ( fill in correct arguments );
72     fprintf ( fill in correct arguments );
73     fprintf ( fill in correct arguments );
74     fprintf ( fill in correct arguments );
75     fprintf ( fill in correct arguments );
76     fprintf ( fill in correct arguments );
77     return;
78 }

```

Illustration 2: Program shell to be completed