**CSCI 2170 OLA6**

Write a C++ program named “**inventory.cpp**” that simulates inventory bins in a warehouse. Each bin holds a number of the same type of parts. The program should use a C++ **struct to** keep the following data:

Description of the part kept in the bin

Number of parts in the bin

This program requires that you use the **sorted linked list** to store the parts. Use typedef to make the ListItemType an alias for the struct type defined for the parts.

The parts carried by the warehouse are shown in the table below:

|  |  |
| --- | --- |
| **Part Description** | **Number of parts in the bin** |
| Valve | 10 |
| Bearing | 5 |
| Bushing | 15 |
| Coupling | 21 |
| Flange | 7 |
| Gear | 5 |
| Gear Housing | 5 |
| Vacuum Gripper | 25 |
| Cable | 18 |
| Rod | 12 |

Your program should read the information about the 10 parts carried by this warehouse from a data file named “inventory.dat”. Here is the format of the data file:

Valve

10

Bearing

5

Bushing

15

Coupling

21

Flange

7

Gear

5

Gear Housing

5

Vacuum Gripper

25

Cable

18

Rod

12

Because it is required that the parts be stored in a **sorted linked list**, *after the parts are inserted into the linked list, they should be sorted in alphabetical order based on the part names.*

The program simulates the maintenance operations performed at the warehouse. Assume that the maintain operations have been recorded in a data file named “operations.dat”. Part of the “operations.dat” file is shown below:

Add

Flange

3

Remove

Rod

2

Remove

Gear Housing

1

Add

Valve

5

…

When the program runs, it should repeat a loop that performs the following steps til the end of data is reached:

* Read the first line of an operation step
* If the operation is “Add”, read the part name and quantity, and **call a function named “Insert”** to add to the part count
* If the operation is “Remove”, read in the part name and quantity, and **call a function named “Remove”**  tosubtract the part count

In addition,you are required towrite a function “Release” to release the memory that was acquired dynamically before the program terminates.

Your program output should consist of the display of the table of parts and their quantities once before the operations are performed, and once after the operation steps are performed.

In summary, the program should have the following 4 functions:

* *Insert* – Increases a specific bin’s part count by a specified number.
* *R****emove*** – Decrease a specific bin’s part count by a specified number.
* ***Display*** – Prints the part name and number of parts in the bin in a table format (with 2 columns as shown above)
* ***Release*** – Release dynamically allocated memory space

Here is a hypothetical example output of the program:

Warehouse inventory before the operations

Part Quantity

|  |  |
| --- | --- |
| Bearing | 5 |
| Bushing | 15 |
| Cable | 18 |
| Coupling | 21 |
| Flange | 7 |
| Gear | 5 |
| Gear Housing | 5 |
| Rod | 12 |
| Vacuum Gripper | 25 |
| Valve | 10 |

Warehouse inventory after the operations

Part Quantity

|  |  |
| --- | --- |
| Bearing | 6 |
| Bushing | 5 |
| Cable | 8 |
| Coupling | 1 |
| Flange | 17 |
| Gear | 50 |
| Gear Housing | 28 |
| Rod | 24 |
| Vacuum Gripper | 19 |
| Valve | 15 |

**Program Submission:**

* You are required to create a **makefile** for this program
* Here are the steps to create the log file:

script ola6log

pr -n -e4 inventory.cpp

pr makefile

make

runOLA6 🡨 assume this is the executable file name created by your makefile

exit

Turn in the program and the log file as:

**handin ola6 inventory.cpp makefile ola6log**