## 

## 

Data Structure & Algorithm

# **Assignment - 2**

**Dr. Norsham Binti Idris**

## Phone stock management system

## Group Members and Their Tasks:

|  |  |  |
| --- | --- | --- |
| **No.** | **Name & Matric No.** | **Tasks** |
| 1 | Aaraf Islam (A20EC4001 ) | Idea Implementation, Code writing, Editing report |
| 2 | Mir Tamzid Hasan (A20EC4037) | Idea implementation, writing code, finishing the report |
| 3 | Islam Mohammed Ruzhan (A20EC4028) | Drawing Class diagram, idea implementation, writing code |

## Class diagram:

<<struct>>

phone

stock

- count : int

- Phones : struct phone

setCount(int) : void

getCount() : int

getCountA() : int

getCountB() : int

getCountC() : int

getPhone(int) : struct phone

addPhone(struct phone): void

setPhone(int, struct phone) : void

display\_all(): void

display\_byModel(string) : void

display\_byFeature(string) : void

display\_byPrice(string) : void

display\_Sort() : void

model : string

feature : string

price : string

1

\*

## Source Code Documentation:

#include <iostream>

#include <fstream>

using namespace std;

struct phone

{

string model;

string battery;

string price;

};

class stock

{

private:

int count;

struct phone phones[100];

public:

stock() {

count = 0;

}

~stock() {

}

void setCount(int c)

{

count = c;

}

int getCount()

{

return count;

}

int getCountA()

{

int c=0;

for(int i=0; i<getCount(); i++)

if(phones[i].model=="Samsung")

c++;

return c;

}

int getCountB()

{

int c=0;

for(int i=0; i<getCount(); i++)

if(phones[i].model=="Xiaomi")

c++;

return c;

}

int getCountC()

{

int c=0;

for(int i=0; i<getCount(); i++)

if(phones[i].model=="Oneplus")

c++;

return c;

}

struct phone getPhone(int n) {

return phones[n];

}

void addPhone(struct phone p) {

int c = getCount();

c++;

setCount(c);

phones[c-1] = p;

}

void setPhone(int n, struct phone t)

{

phones[n].model = t.model;

phones[n].battery = t.battery;

phones[n].price = t.price;

}

void display\_all()

{

for(int i=0; i<getCount(); i++)

{

cout<<i+1<<": "<<phones[i].model<<" "<<phones[i].battery<<" "<<phones[i].price<<endl;

}

}

void display\_byModel(string m)

{

cout<< "Model battery Price"<<endl;

for(int i=0; i<getCount(); i++)

{

if(phones[i].model == m){

cout<<phones[i].model<<" "<<phones[i].battery<<" "<<phones[i].price<<endl;}

}

}

void display\_bybattery(string m)

{

for(int i=0; i<getCount(); i++)

{

if(phones[i].battery == m)

cout<<phones[i].model<<" "<<phones[i].battery<<" "<<phones[i].price<<endl;

}

}

void display\_byPrice(string m)

{

for(int i=0; i<getCount(); i++)

{

if(phones[i].price == m)

cout<<phones[i].model<<" "<<phones[i].battery<<" "<<phones[i].price<<endl;

}

}

void display\_Sort()

{

cout<<"No Model battery Price"<<endl;

for( int i=0; i<getCount()-1; i++)

{

for ( int j=i; j<getCount(); j++)

{

if(phones[j].model == "Samsung")

{

if(phones[i].model != "Samsung")

{

struct phone tmp;

tmp.model = phones[i].model;

tmp.battery = phones[i].battery;

tmp.price = phones[i].price;

phones[i].model = phones[j].model;

phones[i].battery = phones[j].battery;

phones[i].price = phones[j].price;

phones[j].model = tmp.model;

phones[j].battery = tmp.battery;

phones[j].price = tmp.price;

}

}

}

}

for( int i=0; i<getCount()-1; i++)

{

for ( int j=i; j<getCount(); j++)

{

if(phones[j].model == "Xiaomi")

{

if(phones[i].model == "Oneplus")

{

struct phone tmp;

tmp.model = phones[i].model;

tmp.battery = phones[i].battery;

tmp.price = phones[i].price;

phones[i].model = phones[j].model;

phones[i].battery = phones[j].battery;

phones[i].price = phones[j].price;

phones[j].model = tmp.model;

phones[j].battery = tmp.battery;

phones[j].price = tmp.price;

}

}

}

}

display\_all();

}

};

int main()

{

stock p;

struct phone t;

struct phone b;

int go\_on;

struct phone new\_phone;

ifstream myfile ("stock.txt");

if (myfile.is\_open())

{

while(1)

{

getline(myfile, new\_phone.model);

getline(myfile, new\_phone.battery);

getline(myfile, new\_phone.price);

p.addPhone(new\_phone);

if(myfile.eof())

break;

}

myfile.close();

}

do{

cout << "Are you Customer(1) or Shop Owner(2)?";

int choice;

cin>>choice;

switch(choice)

{

case 2:

cout<<"1. Display phone list sorted by model"<<endl;

cout<<"2. Order Phone"<<endl;

cout<<"3. Receive stock update info"<<endl;

cout<<"4. View a phone info"<<endl;

int c2;

cin>>c2;

switch(c2)

{

case 1:

p.display\_Sort();

break;

case 2:

cout<<"Available list(model): "<<endl;

if(p.getCountA()<10)

cout<<"Samsung :"<<p.getCountA()<<endl;

if(p.getCountB()<10)

cout<<"Xiaomi :"<<p.getCountB()<<endl;

if(p.getCountC()<10)

cout<<"Oneplus :"<<p.getCountC()<<endl;

cout<<"Add a phone"<<endl;

cout<<"model: ";

cin>>t.model;

cout<<"battery: ";

cin>>t.battery;

cout<<"price: ";

cin>>t.price;

p.addPhone(t);

break;

case 3:

cout<<"Recently added phone"<<endl;

cout<<"model: "<<t.model<<" ";

cout<<"battery: "<<t.battery<<" ";

cout<<"price: "<<t.price<<endl;

break;

case 4:

cout<<"---Total count of phones in the stock now:"<<p.getCount()<<"---"<<endl;

cout<<"Input No of phone you want to see:";

int n;

cin>>n;

cout<<p.getPhone(n-1).model<<" ";

cout<<p.getPhone(n-1).battery<<" ";

cout<<p.getPhone(n-1).price<<endl;

break;

default:

break;

}

break;

case 1:

cout<<"(1) Search by model" <<endl;

cout << "(2) battery"<<endl;

cout<< "(3) price:"<<endl;

cout<<"(4) Buy a phone"<<endl;

cout<<"(5) Get invoice"<<endl;

int c;

string m,f,pr;

cin>>c;

switch(c)

{

case 1:

cout<<"model:";

cin>>m;

p.display\_byModel(m);

break;

case 2:

cout<<"battery:";

cin>>f;

p.display\_bybattery(f);

break;

case 3:

cout<<"price:";

cin>>pr;

p.display\_byPrice(pr);

break;

case 4:

cout<<"Phone list:"<<endl;

p.display\_Sort();

cout<<"Input the No you are gonna buy:"<<endl;

int n;

cin>>n;

b.model = p.getPhone(n-1).model;

b.battery = p.getPhone(n-1).battery;

b.price = p.getPhone(n-1).price;

for( int i = n; i<p.getCount(); i++)

{

p.setPhone(i, p.getPhone(i-1));

}

break;

case 5:

cout<<"Invoice:"<<endl;

cout<<"Model:"<<b.model<<" "<<"battery:"<<b.battery<<" "<<"price:"<<b.price<<endl;

break;

default:

break;

}

break;

}

cout<<"Continue?(yes:1, No:0)";

cin>>go\_on;

}while(go\_on);

return 0;

}

## Explanation of the sorting/searching techniques used:

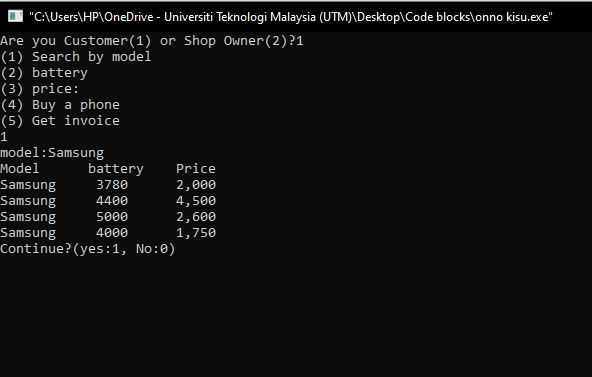
The sorting technique used in this program is the Bubble sort. It is used to sort an array. It is a simple sorting algorithm that works by repeatedly swapping the adjacent elements if they are used in the wrong order. If there are ‘n’ elements in an array, the number of passes in a Bubble sort is n-1. Here, in our code we used bubble sorting to sort the phone numbers according to their model. The sorting function was implemented so that the owner can sort the available phones the way he wanted.

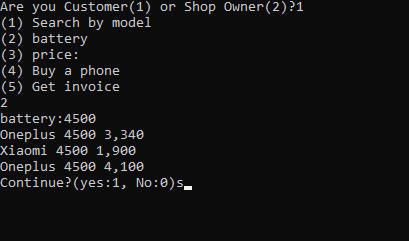
For searching the added phones in the file, we had to use a searching technique. The searching technique that we used was linear searching. It is a searching algorithm where a sequential search is made over all items one by one. Every item is checked and if a match is found then that particular item is returned, otherwise the search continues till the end of the data collection.

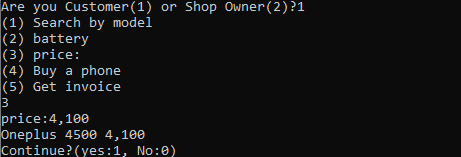
The searching option is for the customer and the searching is done based on the model, battery and price of the phones.

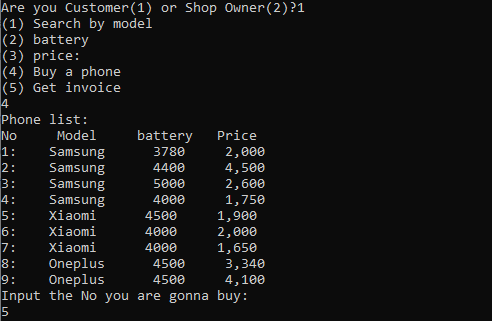
## Screenshot of Output:

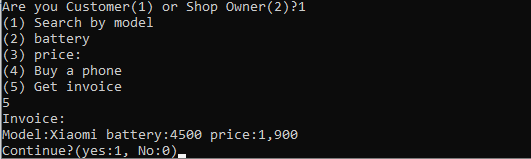
## **Customer Output:**











## **Owner Output:**

