
Algorithms Level 3
PROGRAMMING ADVICES
LEARN THE RIGHT WAY
26+ Years of Experience
Mohammed Abu-Hadhoud
MBA, PMOC, PgMP®, PMP®, PMI-RMP®, CM, ITILF, MCPD, MCSD
حقوق النشر محفوظة، أسعار الكورسات في المنصة هي أسعار رمزيه جدا، ارجو عدم نشر هذه الوثيقة لان نشرها سيمنعنا من الاستمرار في تقديم العلم للآخرين
ارجو عدم استخدام هذه الوثيقة من غير وجه حق لأنك ستحرم الاف الناس من التعلم
ProgrammingAdvices.com

```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <iomanip>

using namespace std;
const string ClientsFileName = "Clients.txt";

struct sClient
{
    string AccountNumber;
    string PinCode;
    string Name;
    string Phone;
    double AccountBalance;
    bool MarkForDelete = false;
};

vector<string> SplitString(string S1, string Delim)
{
    vector<string> vString;

    short pos = 0;
    string sWord; // define a string variable

    // use find() function to get the position of the delimiters
    while ((pos = S1.find(Delim)) != std::string::npos)
    {
        sWord = S1.substr(0, pos); // store the word
        if (sWord != "")
        {
            vString.push_back(sWord);
        }

        S1.erase(0, pos + Delim.length());  /* erase() until
positon and move to next word. */
    }

    if (S1 != "")
    {
        vString.push_back(S1); // it adds last word of the string.
    }

    return vString;

}
```

```
sClient ConvertLinetoRecord(string Line, string Seperator =
"#//#")
{

    sClient Client;
    vector<string> vClientData;

    vClientData = SplitString(Line, Seperator);

    Client.AccountNumber = vClientData[0];
    Client.PinCode = vClientData[1];
    Client.Name = vClientData[2];
    Client.Phone = vClientData[3];
    Client.AccountBalance = stod(vClientData[4]);//cast string to
double

    return Client;
}

string ConvertRecordToLine(sClient Client, string Seperator =
"#//#")
{

    string stClientRecord = "";

    stClientRecord += Client.AccountNumber + Seperator;
    stClientRecord += Client.PinCode + Seperator;
    stClientRecord += Client.Name + Seperator;
    stClientRecord += Client.Phone + Seperator;
    stClientRecord += to_string(Client.AccountBalance);

    return stClientRecord;

}
```

```
vector <sClient> LoadCleintsDataFromFile(string FileName)
{

    vector <sClient> vClients;

    fstream MyFile;
    MyFile.open(FileName, ios::in);//read Mode

    if (MyFile.is_open())
    {

        string Line;
        sClient Client;

        while (getline(MyFile, Line))
        {

            Client = ConvertLinetoRecord(Line);

            vClients.push_back(Client);
        }

        MyFile.close();

    }

    return vClients;

}


void PrintClientCard(sClient Client)
{
    cout << "\nThe following are the client details:\n";
    cout << "\nAccout Number: " << Client.AccountNumber;
    cout << "\nPin Code     : " << Client.PinCode;
    cout << "\nName         : " << Client.Name;
    cout << "\nPhone        : " << Client.Phone;
    cout << "\nAccount Balance: " << Client.AccountBalance;
}
```

```
bool FindClientByAccountNumber(string AccountNumber, vector
<sClient> vClients, sClient& Client)
{

    for (sClient C : vClients)
    {

        if (C.AccountNumber == AccountNumber)
        {
            Client = C;
            return true;
        }

    }
    return false;

}


sClient ChangeClientRecord(string AccountNumber)
{
    sClient Client;

    Client.AccountNumber = AccountNumber;

    cout << "\n\nEnter PinCode? ";
    getline(cin >> ws, Client.PinCode);

    cout << "Enter Name? ";
    getline(cin, Client.Name);

    cout << "Enter Phone? ";
    getline(cin, Client.Phone);

    cout << "Enter AccountBalance? ";
    cin >> Client.AccountBalance;

    return Client;

}
```

```
vector <sClient> SaveCleintsDataToFile(string FileName, vector
<sClient> vClients)
{

    fstream MyFile;
    MyFile.open(FileName, ios::out);//overwrite

    string DataLine;

    if (MyFile.is_open())
    {

        for (sClient C : vClients)
        {

            if (C.MarkForDelete == false)
            {
                //we only write records that are not marked for
delete.
                DataLine = ConvertRecordToLine(C);
                MyFile << DataLine << endl;

            }

        }

        MyFile.close();

    }

    return vClients;

}
```

```
bool UpdateClientByAccountNumber(string AccountNumber, vector
<sClient> &vClients)
{

    sClient Client;
    char Answer = 'n';

    if (FindClientByAccountNumber(AccountNumber, vClients,
Client))
    {

        PrintClientCard(Client);
        cout << "\n\nAre you sure you want update this client? y/n
? ";
        cin >> Answer;
        if (Answer == 'y' || Answer == 'Y')
        {

            for (sClient & C : vClients)
            {
                if (C.AccountNumber == AccountNumber)
                {
                    C = ChangeClientRecord(AccountNumber);
                    break;
                }

            }

            SaveCleintsDataToFile(ClientsFileName, vClients);

            cout << "\n\nClient Updated Successfully.";
            return true;
        }

    }
    else
    {
        cout << "\nClient with Account Number (" << AccountNumber
<< ") is Not Found!";
        return false;
    }

}
```

```
string ReadClientAccountNumber()
{
    string AccountNumber = "";

    cout << "\nPlease enter AccountNumber? ";
    cin >> AccountNumber;
    return AccountNumber;

}

int main()

{
    vector <sClient> vClients =
LoadCleintsDataFromFile(ClientsFileName);
    string AccountNumber = ReadClientAccountNumber();

    UpdateClientByAccountNumber(AccountNumber, vClients);

    system("pause>0");
    return 0;
}
```