

// project2.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include <iostream>

#include <string>

#include <fstream>

#include <iomanip>

#include <conio.h>

using namespace std;

struct MasterRecord

{

string customer\_No;

string customer\_Name;

string street;

string city;

string state;

string zipcode;

};

struct TransactionRecord

{

string tCustomer\_NO;

string transData;

string transCode;

string junk;

};

struct NewMasterRecord

{

string customer\_No;

string customer\_name;

string street;

string city;

string state;

string zipcode;

string cust\_NO;

string transData;

string transCode;

string junk;

};

// Declare functions

void openAllFiles(ifstream &, ifstream &, ofstream &);

void closeAllFiles(ifstream &, ifstream &, ofstream &, ofstream &);

void display();

int main()

{

string oldCustomer\_No, oldCustomer\_Name, oldStreet, oldCity, oldState, oldZipCode;

int masterCounter = 0, transCounter = 0, errorCounter = 0, newCounter = 0;

//declare the object ParentLine,DependentLine for the ParentRecord structure

MasterRecord MasterLine;

TransactionRecord TransactionLine;

NewMasterRecord NewMasterLine;

ifstream(Master);

ifstream(Transaction);

ofstream(Result);

ofstream(NewMaster);

// open files

openAllFiles(Master,Transaction,Result);

//Get the Data

getline(Master,MasterLine.customer\_No,'#');

getline(Master,MasterLine.customer\_Name,'#');

getline(Master,MasterLine.street,'#');

getline(Master,MasterLine.city,'#');

getline(Master,MasterLine.state,'#');

getline(Master,MasterLine.zipcode,'\n');

masterCounter++;

getline(Transaction,TransactionLine.tCustomer\_NO,'#');

getline(Transaction,TransactionLine.transData,'#');

getline(Transaction,TransactionLine.transCode,'#');

getline(Transaction,TransactionLine.junk,'\n');

transCounter++;

display();

//The main loop to process all the trans records

while (!Master.eof() && !Transaction.eof())

{

//If the trans is equal to the master

if ((MasterLine.customer\_No == TransactionLine.tCustomer\_NO) && (!Transaction.eof()))

{// Update the old master and write out the new master record

oldCustomer\_No = MasterLine.customer\_No;

oldCustomer\_Name = MasterLine.customer\_Name;

oldStreet = MasterLine.street;

oldCity = MasterLine.city;

oldState = MasterLine.state;

while ((MasterLine.customer\_No == TransactionLine.tCustomer\_NO) && (!Transaction.eof()))

{

int transCode = stoi(TransactionLine.transCode.c\_str());

switch (transCode)

{

case 1:

MasterLine.customer\_Name = TransactionLine.transData;

break;

case 2:

MasterLine.street = TransactionLine.transData;

break;

case 3:

MasterLine.city = TransactionLine.transData;

break;

default:

break;

}//end case transCode

cout << "Transaction " << oldCustomer\_No << " " << transCode << endl;

//Write to new file

NewMaster << '#' << MasterLine.customer\_No << '#';

NewMaster << '#' << MasterLine.customer\_Name << '#';

NewMaster << '#' << MasterLine.street << '#';

NewMaster << '#' << MasterLine.city << '#';

NewMaster << '#' << MasterLine.state << '#';

NewMaster << '#' << MasterLine.zipcode << '#';

newCounter++;

//read in the next transaction record

getline(Transaction,TransactionLine.tCustomer\_NO,'#');

getline(Transaction,TransactionLine.transData,'#');

getline(Transaction,TransactionLine.transCode,'#');

getline(Transaction,TransactionLine.junk,'\n');

transCounter++;

}//end while == w/ switch

cout << " Old M " << oldCustomer\_No << " " << oldCustomer\_Name << oldStreet << oldCity << oldState << endl;

cout << " new M " << oldCustomer\_No << " " << MasterLine.customer\_Name << MasterLine.street << MasterLine.city << endl;

}//end of ==

if ((MasterLine.customer\_No < TransactionLine.tCustomer\_NO) && (!Transaction.eof()))

{

//masterLine is unique, write it to the new master file

NewMaster << '#' << MasterLine.customer\_No << '#';

NewMaster << '#' << MasterLine.customer\_Name << '#';

NewMaster << '#' << MasterLine.street << '#';

NewMaster << '#' << MasterLine.city << '#';

NewMaster << '#' << MasterLine.state << '#';

NewMaster << '#' << MasterLine.zipcode << '#';

newCounter++;

// read in next masterLine

getline(Master,MasterLine.customer\_No,'#');

getline(Master,MasterLine.customer\_Name,'#');

getline(Master,MasterLine.street,'#');

getline(Master,MasterLine.city,'#');

getline(Master,MasterLine.state,'#');

getline(Master,MasterLine.zipcode,'\n');

masterCounter++;

}

if ((MasterLine.customer\_No > TransactionLine.tCustomer\_NO) && (!Transaction.eof()))

{

// tranLine is unique, create new record and write to file

// create blank fields of appropriate length for new record

NewMaster << '#' << " " << '#';

NewMaster << '#' << " " << '#';

NewMaster << '#' << " " << '#';

NewMaster << '#' << " " << '#';

NewMaster << '#' << " " << '#';

NewMaster << '#' << " " << '#';

NewMaster << '#' << " " << '#';

// determine category of update, move data to the new master from the trans record

int transCode = stoi(TransactionLine.transCode.c\_str());

switch (transCode)

{

case 1:

MasterLine.customer\_Name = TransactionLine.transData;

break;

case 2:

MasterLine.street = TransactionLine.transData;

break;

case 3:

MasterLine.city = TransactionLine.transData;

break;

default:;

}

// read in new transaction record

getline(Transaction,TransactionLine.tCustomer\_NO,'#');

getline(Transaction,TransactionLine.transData,'#');

getline(Transaction,TransactionLine.transCode,'#');

getline(Transaction,TransactionLine.junk,'\n');

transCounter++;

cout << "\*\* ERROR\*\* NO MATCH FOR CUSTOMER NO. " << NewMasterLine.customer\_No << " -REFERENCE NO. " << transCounter;

cout << endl;

errorCounter++;

}

}//end of while loop

//Write it to the Result file

Result << '#' << MasterLine.customer\_No << '#';

Result << '#' << MasterLine.customer\_Name << '#';

Result << '#' << MasterLine.street << '#';

Result << '#' << MasterLine.city << '#';

Result << '#' << MasterLine.state << '#';

Result << '#' << MasterLine.zipcode << '#';

//Close all files

closeAllFiles(Master,Transaction, Result, NewMaster);

//Output Counters

cout << " Total Transaction Records added : " << transCounter << endl;

cout << " Total Transaction Records in Error : " << errorCounter << endl;

cout << " Total Master Records : " << masterCounter << endl;

cout << " Total New Master Records : " << newCounter << endl;

system("pause");

} // end main()

//Functions to Open and Close Files

void openAllFiles(ifstream& PFile, ifstream& DFile, ofstream& OFile)

{

PFile.open("ParentFile.txt", ios::in);

DFile.open("DependentFile.txt", ios::in);

OFile.open("Result.txt", ios::out);

OFile.open("NewMaster.txt", ios::out);

}

void closeAllFiles(ifstream& PPFile, ifstream& DDFile, ofstream& OOFile,ofstream& nmFile)

{

PPFile.close();

DDFile.close();

OOFile.close();

nmFile.close();

}

void display()

{

cout << "Julian Barber June 25th, 2013\n\n";

cout << " Name and Address Update Journal\n\n";

cout << " CustNO Name Address City ST.\n\n";

}