// Declaration of header files  
#include <iostream.h>  
#include <fstream.h>  
#include <process.h>  
#include <string.h>  
#include <stdio.h>  
#include <ctype.h>  
#include <conio.h>  
#include <dos.h>  
#include <stdlib.h>  
#include <iomanip.h>  
#include <graphics.h>

typedef char option[15];  
const int ROW = 10, COL = 10;

int scan;    // To hold the special characters for moving the prompt in menu  
int ascii;

// To display the main menu options  
option a[]= {  
“NewCassPur”,  
“ListOfPurchase”,  
“DailyCassSales”,  
“SalesReport”,  
“AddOtherExp”,  
“OtherExpRep”,  
“ClosingStock”,  
“MonthlyProfit”,  
“Exit”  
};

// Function used to do screening  
class main\_menu  
{  
int i,done;

public:  
void normalvideo(int x,int y,char \*str);  
void reversevideo(int x,int y,char \*str);  
void box(int x1,int y1,int x2,int y2);  
char menu();  
void control\_menu();  
void help(void);  
};

/\* Class member functions for drawing boxes \*/  
class shape  
{  
public:  
void line\_hor(int, int, int, char);  
void line\_ver(int, int, int, char);  
void box(int, int, int, int, char);  
};

// Class contains the cassettes deposit of customers  
class cassettes  
{  
public:  
void new\_cassettes(void);    // Function to add new cassettes in music shop  
// For cassettes entry into the cassettes.dat data file  
void add\_to\_file(int, char t\_Cass\_name[30], char t\_Comp\_name[30], int, int, int, int, float);  
void display\_list(void); // Displaying the cassettes list  
void delete\_cassette(int); // Deletes the temporary cassette record  
int last\_cas\_code(void); // if the file is exist or not  
char \*return\_name(int); // Function for validation entry of name  
char \*return\_address(int); // Function for validation entry of address  
int recordno(int);  
void display(int); // To display the cassette information account

private:  
// Data Members of cassette.dat data file

int cas\_code;    // Cassette code – film, nonfilm, religious  
char Cass\_name[30];    // Title of the cassette  
char Comp\_name[30];    // Cassette company  
int tot\_cassette;    // Total cassette purchased  
int dd, mm, yy;    // Date of purchase  
float price;    // Price per cassette  
};

class balance  
{  
public:  
void add\_to\_file(int, int, int, int, int, float);    // Add record for balance cassette  
void balance\_cassette(void);    // Function to display the balance cassette  
void delete\_balance(int);    // Deleting the balance from Tbal.dat  
int give\_balance(int);    // returns the current balance of a particular  
// cassette code  
void Update\_balance(int, int, int , int , int, float); // Function to update balance after cassette sale  
float return\_price(int);    // Return the cassette price

void monthly\_profit(void);    // Monthly profit after every transaction

private:

// Data members of Tbal.dat data file  
int cas\_code;    // Cassette code to be balance  
int cas\_bal;    // Total number of cassettes in balance  
int dd;    // Balance date  
int mm;  
int yy;  
float price;    // Unit price of cassettes on code wise

};

// Class contains the customers daily cassette transaction entry  
class account  
{  
public:  
void new\_account(void);  // Function to sale cassettes  
void close\_account(void);    // Function to close the cassette purchase  
void clear(int, int); // Function to perform a clear screen function  
void Display\_sales(void);    // Displaying the sales report of cassettes  
int last\_cas\_code(void); // if the file is exist or not  
float sales\_status(int m1, int y1, float cod1, float cod2, float cod3);  
private:  
void add\_to\_file(int, char tP\_name[30], char tP\_address[30],int, int, int, int); // Function to add transaction records  
void delete\_account(int); // Function to delete a transaction record

// Data members of person.dat data file  
int cas\_code;        // Cassette code – film, nonfilm, religious  
char P\_name[30];    // Person Name  
char P\_address[30];    // Person Address  
int No\_cass;        // Number of cassette  
int dd, mm, yy;     // To store the system date/ purchase date  
};

class other  
{  
public :  
// Function to display other expense screen like salary,  
// electricity bill, telephone and miscellaneous expences  
void Expense\_other(void);

// Add the data to Other.dat data file  
void add\_to\_file(int, char tNat\_Expen[30], int, int, int, float);  
void display\_expense(void);    // Display the expenses  
float other\_status(int m1, int y1);

private :  
// Data members of other.dat data file  
int O\_code;        // Other expence code  
/\* The other expence code as :  
1     –    Salary to workers,  
2    –    Electricity bill  
3    –    Telephone bills  
4    –    Miscellaneous expenses    \*/  
char Nat\_Expen[30];    // Nature of expense description  
int dd, mm, yy;    // Expense date  
float amount;    // Expense amount  
};

// Function to displays all the menu prompt messages  
// from the pointer array of option a[]  
void main\_menu::normalvideo(int x,int y,char \*str)  
{  
gotoxy(x,y);  
cprintf(“%s”,str);  
}

// Function to move the cursor on the menu prompt  
// with a reverse video color  
void main\_menu::reversevideo(int x,int y,char \*str)  
{  
textcolor(5+143);  
textbackground(WHITE);  
gotoxy(x,y);  
cprintf(“%s”,str);  
textcolor(GREEN);  
textbackground(BLACK);  
}

void main\_menu::box(int x1,int y1,int x2,int y2)  
{  
for(int col=x1;col<x2;col++)  
{  
gotoxy(col,y1);  
cprintf(“%c”,196);  
gotoxy(col,y2);  
cprintf(“%c”,196);  
}

for(int row=y1;row<y2;row++)  
{  
gotoxy(x1,row);  
cprintf(“%c”,179);  
gotoxy(x2,row);  
cprintf(“%c”,179);  
}  
gotoxy(x1,y1);  
cprintf(“%c”,218);  
gotoxy(x1,y2);  
cprintf(“%c”,192);  
gotoxy(x2,y1);  
cprintf(“%c”,191);  
gotoxy(x2,y2);  
cprintf(“%c”,217);  
}

// Displaying the main menu of music system  
char main\_menu::menu()  
{  
clrscr();

textcolor(22);  
box(20, 6, 65, 20);  
box(18, 4, 67, 22);  
textcolor(5+143);  
gotoxy(36, 5);  
textbackground(BLUE);  
cprintf(“M U S I C”);  
textbackground(BLACK);  
textcolor(22);  
for(i = 1; i < 9; i++)  
normalvideo(32, i+10, a[i]);  
reversevideo(32, 10, a[0]);  
i = done = 0;  
\_setcursortype(\_NOCURSOR);  
do  
{  
int key = getch();

switch (key)  
{  
case 00:  
key = getch();

switch (key)  
{  
case 72:  
normalvideo(32, i+10, a[i]);  
i–;  
if (i == -1)  
i = 8;  
reversevideo(32,i+10, a[i]);  
break;  
case 80:  
normalvideo(32, i+10, a[i]);  
i++;  
if (i == 9)  
i = 0;  
reversevideo(32, i+10, a[i]);  
break;  
}  
break;  
case 13:  
done = 1;  
}  
} while (!done);

\_setcursortype(\_NOCURSOR);  
return(i+49);  
}

/\* The function main\_menu() is used to display the main menu system \*/  
void main\_menu::control\_menu()  
{

char choice;  
cassettes cas;  
account a;  
balance bal;  
other oth;

do  
{  
choice = menu();  
clrscr();

switch (choice)  
{  
case ‘1’:  
\_setcursortype(\_NORMALCURSOR);  
box(3, 1, 75, 24);  
box(5, 2, 73, 23);

cas.new\_cassettes(); // New cassette entry function  
break;  
case ‘2’:  
box(3, 1, 75, 24);  
box(5, 2, 73, 23);  
cassettes ini;  
ini.display\_list(); // Glogal list of cassettes function  
break;  
case ‘3’:  
box(3, 1, 75, 24);  
box(5, 2, 73, 23);

\_setcursortype(\_NORMALCURSOR);

a.new\_account(); // Enter the personwise sale cassette  
break;  
case ‘4’:  
box(3, 1, 75, 24);  
box(5, 2, 73, 23);

account a;  
\_setcursortype(\_NORMALCURSOR);  
a.Display\_sales(); // Displays the sales cassettes  
break;  
case ‘5’:  
box(3, 1, 75, 24);  
box(5, 2, 73, 23);

\_setcursortype(\_NORMALCURSOR);  
oth.Expense\_other(); // Entry of other expenses  
break;  
case ‘6’:  
box(3, 1, 75, 24);  
box(5, 2, 73, 23);  
gotoxy(10,10);  
oth.display\_expense();    // Displays the expenses  
break;  
case ‘7’:  
box(3, 1, 75, 24);  
box(5, 2, 73, 23);  
gotoxy(10,10);  
bal.balance\_cassette(); // Function to display the balance cassettes in shop  
break;  
case ‘8’ :  
box(3, 1, 75, 24);  
box(5, 2, 73, 23);  
gotoxy(10,10);  
// Monthly profit after every transaction  
bal.monthly\_profit();  
break;  
case ‘9’ :exit(0);  
}  
} while (choice != 8);  
}

/\* Function to draw horizontal line for menu\*/  
void shape::line\_hor(int column1, int column2, int row, char c)  
{  
for (column1; column1 <= column2; column1++)  
{  
gotoxy(column1, row);  
cout << c;  
}  
}

/\* Function to draw vertical line for menu \*/  
void shape::line\_ver(int row1, int row2, int column, char c)  
{  
for (row1; row1 <= row2; row1++)  
{  
gotoxy(column, row1);  
cout << c;  
}  
}

/\* Function for drawing boxes for menu \*/  
void shape::box(int column1, int row1, int column2, int row2, char c)  
{  
char ch = 218;  
char c1, c2, c3, c4;  
char l1 = 196, l2 = 179;  
if (c == ch)  
{  
c1 = 218;  
c2 = 191;  
c3 = 217;  
c4 = 217;  
l1 = 196;  
l2 = 179;  
}  
else  
{  
c1 = c;  
c2 = c;  
c3 = c;  
c4 = c;  
l1 = c;  
c2 = c;  
}

gotoxy(column1, row1);  
cout << c1;  
gotoxy(column2, row1);  
cout << c2;  
gotoxy(column1, row2);  
cout << c3;  
gotoxy(column2, row2);  
cout << c4;  
column1++;  
column2–;  
line\_hor(column1, column2, row1, l1); //Horizontal line  
line\_hor(column1, column2, row2, l1);  
column1–;  
column2++;  
row1++;  
row2–;  
line\_ver(row1, row2, column1, l2); // Vertical line  
line\_ver(row1, row2, column2, l2);  
}

/\* Function to display help about this project \*/  
void main\_menu::help(void)  
{  
clrscr();

setbkcolor(7);  
settextstyle(7,HORIZ\_DIR,5);  
outtextxy(70,20,”Welcome to Music Shop”);  
settextstyle(2,HORIZ\_DIR,5);  
outtextxy(60,100, “This project your can keep record of daily music cassettes “);  
delay(2);  
outtextxy(60,130, “transaction. This program is capable of holding any no. of cassettes.”);  
delay(2);  
outtextxy(60,160, “-In first option you can enter new cassette information”);  
delay(2);  
outtextxy(60,190, “-In second option you can see the list of all the cassettes”);  
delay(2);  
outtextxy(60,220,”-In third option you can sales the cassettes according to demand”);  
delay(2);  
outtextxy(60,250, “-Through fourth optiion you can see the sales status”);  
delay(2);  
outtextxy(60,280, “In the fifth option enter your daily other expenses”);  
delay(2);  
outtextxy(60,310, “-In fifth option you can take monthWise individual account report”);  
delay(2);  
outtextxy(60,340,”-In sixth option you can take reports of other expenses”);  
delay(2);  
outtextxy(60,370, “Note-: Enter only valid data while you maintain this package”);  
delay(2);  
outtextxy(60,400, “-And last option is Quit (Exit From Music Window)”);  
delay(2);

settextstyle(7,HORIZ\_DIR,4);  
outtextxy(80,420,”Press any key to continue…”);  
getch();

}

/\* Function for displaying a cassette information when required \*/  
void cassettes::display(int t\_cas\_code)  
{  
fstream file;  
file.open(“cassettes.dat”, ios::in);  
file.seekg(0, ios::beg);

// Displays the record contents matching  
// with t\_cas\_code from cassettes.dat data file  
while (file.read((char \*)this, sizeof(cassettes)))  
{  
if (t\_cas\_code == cas\_code)  
{  
gotoxy(8, 5);  
cout << “Account no. ” << cas\_code;  
gotoxy(10, 8);  
cout << “Name : “;  
puts(Cass\_name);  
gotoxy(10, 9);  
cout << “Comany Name : “;  
puts(Comp\_name);

gotoxy(10, 10);  
cout << “Price : ” << setw(15)    // setwidth  
<< setprecision(2)    // set position of decimal point  
<< setiosflags(ios::left)   // set left justified output  
<< setiosflags(ios::showpoint)  // always show decimal point  
<< setiosflags(ios::fixed)    // set fixed notation for display  
<< price;  
break;  
}  
}

file.close();  
}

/\* Function to find is their records in cassette.dat file or not \*/

int cassettes::last\_cas\_code(void)  
{  
fstream file;  
file.open(“cassettes.dat”, ios::in);  
file.seekg(0, ios::beg);  
int tcount = 0;  
int count = 0;

// Founds the last account no.  
while (file.read((char \*)this, sizeof(cassettes)))  
{  
tcount = cas\_code;  
count = count + tcount;  
}

file.close();  
return count;  
}

/\* Function to find is their records in file person.dat or not \*/

int account::last\_cas\_code(void)  
{  
fstream file;  
file.open(“Person.dat”, ios::in);  
file.seekg(0, ios::beg);  
int tcount = 0;  
int count = 0;  
// Founds the last account no.  
while (file.read((char \*)this, sizeof(account)))  
{  
tcount = cas\_code;  
count = count + tcount;  
}  
file.close();  
return count;  
}

/\* Function for returning balance cassette.  
This function returns the balance cassette on code wise  
to know the current balance from “Tbal.dat” data file. \*/

int balance::give\_balance(int t\_cas\_code)  
{  
fstream file;  
file.open(“Tbal.dat”, ios::in);  
file.seekg(0, ios::beg);  
int t\_balance = 0;

// Gives the last balance of an individual account  
while (file.read((char \*)this, sizeof(balance)))  
{  
if (file.eof())  
break;

if (cas\_code == t\_cas\_code)  
{  
t\_balance = cas\_bal;  
break;  
}  
}  
file.close();  
return t\_balance;  
}

// Function to return the cassette price  
float balance::return\_price(int t\_cas\_code)  
{  
float t\_price = 0.0;  
fstream file;  
file.open(“Tbal.dat”, ios::in);  
file.seekg(0, ios::beg);

// Gives the last balance of an individual account  
while (file.read((char \*)this, sizeof(balance)))  
{  
if (file.eof())  
break;

if (cas\_code == t\_cas\_code)  
{  
t\_price = price;  
break;  
}  
}  
file.close();  
return t\_price;  
}

/\* This function add\_to\_file() is used to create  
new/fresh record in the data file \*/

void cassettes::add\_to\_file(int t\_cas\_code, char t\_Cass\_name[30],  
char t\_Comp\_name[30], int t\_cas, int tdd, int tmm, int tyy, float t\_price)  
{  
cas\_code = t\_cas\_code;  
strcpy(Cass\_name, t\_Cass\_name);  
strcpy(Comp\_name, t\_Comp\_name);  
tot\_cassette = t\_cas;  
dd = tdd;  
mm = tmm;  
yy = tyy;  
price = t\_price;

fstream file;

// Appends new account record with the price into cassettes.dat data file  
file.open(“cassettes.dat”, ios::out|ios::app);  
file.write((char \*)this, sizeof(cassettes));  
file.close();  
}

/\* This function add\_to\_file() is used to  
create new/fresh record in the data file \*/

void balance::add\_to\_file(int t\_cas\_code,  
int t\_cas, int tdd, int tmm, int tyy, float t\_price)  
{  
int bal = give\_balance(t\_cas\_code);

if (bal > 0)  
{  
fstream file;  
file.open(“Tbal.dat”, ios::in);  
fstream temp;  
temp.open(“TEMP.dat”, ios::out);  
file.seekg(0,ios::beg);

// Uses a copy method to delete the cassette  
// from cassette.dat data file  
while (!file.eof())  
{  
file.read((char \*)this, sizeof(balance));  
if (file.eof())  
break;  
if (cas\_code != t\_cas\_code)  
temp.write((char \*)this, sizeof(balance));  
else  
{  
cas\_code = t\_cas\_code;  
cas\_bal = t\_cas + bal;  
dd = tdd;  
mm = tmm;  
yy = tyy;  
price = t\_price;  
temp.write((char \*)this, sizeof(balance));  
}  
}

file.close();  
temp.close();  
file.open(“Tbal.dat”, ios::out);  
temp.open(“TEMP.dat”, ios::in);  
temp.seekg(0, ios::beg);

// Copy the TEMP.dat contents into cassette.dat data file  
while (!temp.eof())  
{  
temp.read((char \*)this, sizeof(balance));  
if (temp.eof())  
break;  
file.write((char \*)this, sizeof(balance));  
}  
file.close();  
temp.close();  
}  
else  
{  
cas\_code = t\_cas\_code;  
cas\_bal = t\_cas;  
dd = tdd;  
mm = tmm;  
yy = tyy;  
price = t\_price;  
fstream file;

// Appends new account record with the  
// price into cassettes.dat data file  
file.open(“Tbal.dat”, ios::out|ios::app);  
file.write((char \*)this, sizeof(balance));  
file.close();  
}  
}

// Function for deleting a account from cassettes.dat file  
void cassettes::delete\_cassette(int t\_cas\_code)  
{  
fstream file;  
file.open(“cassettes.dat”, ios::in);  
fstream temp;  
temp.open(“TEMP.dat”, ios::out);  
file.seekg(0,ios::beg);

// Uses a copy method to delete the account  
// from cassette.dat data file  
while (!file.eof())  
{  
file.read((char \*)this, sizeof(cassettes));  
if (file.eof())  
break;  
if (cas\_code != t\_cas\_code)  
temp.write((char \*)this, sizeof(cassettes));  
}

file.close();  
temp.close();  
file.open(“cassettes.dat”, ios::out);  
temp.open(“TEMP.dat”, ios::in);  
temp.seekg(0, ios::beg);

// Copy the TEMP.dat contents into cassette.dat data file  
while (!temp.eof())  
{  
temp.read((char \*)this, sizeof(cassettes));  
if (temp.eof())  
break;  
if (cas\_code != t\_cas\_code)  
file.write((char \*)this, sizeof(cassettes));  
}

file.close();  
temp.close();  
}

// Function for deleting a balance from Tbal.dat file  
void balance::delete\_balance(int t\_cas\_code)  
{  
fstream file;  
file.open(“Tbal.dat”, ios::in);  
fstream temp;  
temp.open(“TEMP.dat”, ios::out);  
file.seekg(0,ios::beg);

// Uses a copy method to delete the account from Tbal.dat data file  
while (!file.eof())  
{  
file.read((char \*)this, sizeof(balance));  
if (file.eof())  
break;  
if (cas\_code != t\_cas\_code)  
temp.write((char \*)this, sizeof(balance));  
}  
file.close();  
temp.close();  
file.open(“Tbal.dat”, ios::out);  
temp.open(“TEMP.dat”, ios::in);  
temp.seekg(0, ios::beg);

// Copy the TEMP.dat contents into Tbal.dat data file  
while (!temp.eof())  
{  
temp.read((char \*)this, sizeof(balance));  
if (temp.eof())  
break;  
if (cas\_code != t\_cas\_code)  
file.write((char \*)this, sizeof(balance));  
}

file.close();  
temp.close();  
}

/\* Function for add an account details of  
daily tranaction into person.dat file. \*/

void account::add\_to\_file(int t\_cas\_code, char tP\_name[30],  
char tP\_address[30],int tNo\_cass, int d1, int m1, int y1)  
{

cas\_code = t\_cas\_code;  
strcpy(P\_name, tP\_name);  
strcpy(P\_address, tP\_address);  
No\_cass = tNo\_cass;  
dd = d1;  
mm = m1;  
yy = y1;

fstream file;

// Appends the transaction record into person.dat data file  
file.open(“person.dat”, ios::out|ios::app);  
file.write((char \*)this, sizeof(account));  
file.close();  
}

/\* Function for deleting an account from person.dat file. \*/  
void account::delete\_account(int t\_cas\_code)  
{  
fstream file;  
file.open(“person.dat”, ios::in); // Open to read records  
fstream temp;  
temp.open(“TEMP.dat”, ios::out); // Open to write records  
file.seekg(0, ios::beg);    // Positioned from begining of the file

// Uses the copy method for deleting the  
// transaction record from person.dat data file  
while (!file.eof())  
{  
file.read((char \*)this, sizeof(account));  
if (file.eof())  
break;  
if (cas\_code != t\_cas\_code)  
temp.write((char \*)this, sizeof(account));  
}

file.close();  
temp.close();  
file.open(“person.dat”, ios::out);  
temp.open(“TEMP.dat”, ios::in);  
temp.seekg(0, ios::beg);

// Uses copy method to transfer the record  
// from TEMP.dat file to person.dat data file  
while (!temp.eof())  
{  
temp.read((char \*)this, sizeof(account));  
if (temp.eof())  
break;  
if (cas\_code != t\_cas\_code)  
file.write((char \*)this, sizeof(account));  
}

file.close();  
temp.close();  
}

// Function for add an other expenses details in other.dat data file  
void other::add\_to\_file(int tO\_code, char tNat\_Expen[30],  
int d1, int m1,  int y1, float tamount)  
{  
O\_code = tO\_code;  
strcpy(Nat\_Expen, tNat\_Expen);  
dd = d1;  
mm = m1;  
yy = y1;  
amount = tamount;

fstream file;

// Appends the transaction record into other.dat data file  
file.open(“other.dat”, ios::out|ios::app);  
file.write((char \*)this, sizeof(other));  
file.close();  
}

/\* Function for displaying an account from “cassettes.dat”. \*/  
void cassettes::display\_list(void)  
{  
clrscr();  
int flag;  
float tot\_price = 0.0, tot\_balance = 0.0;  
fstream file;  
gotoxy(25,2);

cout << “List of cassettes in Music Shop”;  
gotoxy(25, 3);  
cout << “===============================”;  
int d1, m1, y1;  
struct date d;        // For extracting system date  
getdate(&d);  
d1 = d.da\_day;  
m1 = d.da\_mon;  
y1 = d.da\_year;  
gotoxy(62, 3);  
cout << “Date: ” << d1 << “/” << m1 << “/” << y1;  
gotoxy(1, 4);  
for (int j = 1; j <= 79; j++)  
cout << “=”;

gotoxy(1, 5);  
cout << “Code #”;  
gotoxy(9, 5);  
cout << “Cassette Name”;  
gotoxy(34, 5);  
cout << “Company Name”;  
gotoxy(55, 5);  
cout << “Qty”;  
gotoxy(61, 5);  
cout << “Price”;  
gotoxy(72, 5);  
cout << “Total”;  
gotoxy(1, 6);  
for (j = 1; j <= 79; j++)  
cout << “=”;

file.open(“cassettes.dat”, ios::in);  
file.seekg(0,ios::beg);  
int row = 7;

// Reads all the records to display on the screen  
while (file.read((char \*)this, sizeof(cassettes)))  
{  
tot\_price = 0.0;  
flag = 0;  
delay(2);  
gotoxy(3, row);  
cout << cas\_code;  
gotoxy(9, row);  
puts(Cass\_name);  
gotoxy(34, row);  
puts(Comp\_name);  
gotoxy(55, row);  
cout << tot\_cassette;

gotoxy(61, row);  
cout <<  setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< price;

tot\_price = tot\_cassette \* price;

gotoxy(72, row);  
cout <<  setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< tot\_price;

tot\_balance = tot\_balance + tot\_price;

row++;  
if (row > 23)  
{  
flag = 1;  
row = 6;  
gotoxy(4, 24);  
cout << “Press any key to continue…. “;  
getch();  
clrscr();  
}  
}

gotoxy(1, row);  
for (j = 1; j <= 79; j++)  
cout << “=”;  
row++;  
gotoxy(3, row);  
cout << “Total price of Cassettes is : “;

gotoxy(72, row);  
cout << setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< tot\_balance;

file.close();  
if (!flag)  
{  
gotoxy(4, 24);  
cout << “Press any key to continue…”;  
getch();  
}  
getch();  
}

/\* Function for displaying an cassette sales on day wise \*/  
void account::Display\_sales(void)  
{  
clrscr();  
int flag;  
fstream file;

int tcas\_code;  
int xdd, xmm, xyy;  
clrscr();  
gotoxy(10, 8);  
cout << “Enter the Cassette Code “;  
gotoxy(10, 9);  
cout <<“As 1 or 2 or 3 “;  
gotoxy(10, 11);  
cout << “Enter the date ” ;  
gotoxy(30, 9);  
cin >> tcas\_code;  
gotoxy(30, 11);  
cin >> xdd;  
gotoxy(32, 11);  
cout << “-“;  
gotoxy(33, 11);  
cin >> xmm;  
gotoxy(35, 11);  
cout << “-“;  
gotoxy(36, 11);  
cin >> xyy;  
if (tcas\_code < 1)  
{  
gotoxy(5, 23);  
cout << “Your entry is not valid”;  
getch();  
return;  
}  
else  
if (tcas\_code > 3)  
{  
gotoxy(5, 23);  
cout << “Your entry is not valid”;  
getch();  
return;  
}

clrscr();  
gotoxy(25, 2);  
cout << “Sales Report in Music Shop”;  
gotoxy(25, 3);  
cout << “==========================”;  
int d1, m1, y1;  
struct date d;        // For extracting system date  
getdate(&d);  
d1 = d.da\_day;  
m1 = d.da\_mon;  
y1 = d.da\_year;  
gotoxy(62, 3);  
cout << “Date: ” << d1 << “/” << m1 << “/” << y1;  
gotoxy(1, 5);  
for (int j = 1; j <= 79; j++)  
cout << “=”;

gotoxy(1, 6);  
cout << “Code #”;  
gotoxy(9, 6);  
cout << “Name”;  
gotoxy(34, 6);  
cout << “Address”;  
gotoxy(55, 6);  
cout << “Qty”;  
gotoxy(61, 6);  
cout << “Date”;

gotoxy(1, 7);  
for (j = 1; j <= 79; j++)  
cout << “=”;  
file.open(“Person.dat”, ios::in);  
file.seekg(0,ios::beg);  
int row = 8;

// Reads all the records to display on the screen  
while (file.read((char \*)this, sizeof(account)))  
{  
if ((tcas\_code == cas\_code) && (xmm == mm) && (xyy == yy))  
{  
flag = 0;  
delay(2);  
gotoxy(3, row);  
cout << cas\_code;  
gotoxy(9, row);  
puts(P\_name);  
gotoxy(34, row);  
puts(P\_address);  
gotoxy(55, row);  
cout << No\_cass;

gotoxy(61, row);  
cout << dd << “-” << mm << “-” << yy;  
++row;  
if (row > 23)  
{  
flag = 1;  
row = 6;  
gotoxy(4, 24);  
cout << “Press any key to continue…. “;  
getch();  
clrscr();  
}  
}  
}

++row;

gotoxy(1, row);  
for (j = 1; j <= 79; j++)  
cout << “=”;  
row++;

file.close();  
if (!flag)  
{  
gotoxy(4, 24);  
cout << “Press any key to continue…”;  
getch();  
}  
getch();  
}

/\* Function for displaying other expenses in shop \*/  
void other::display\_expense(void)  
{  
clrscr();  
int flag;  
fstream file;

/\* The other expence code as :  
1     –    Salary to workers,  
2    –    Electricity bill  
3    –    Telephone bills  
4    –    Miscellaneous expenses    \*/

int tO\_code;  
int xdd, xmm, xyy;  
clrscr();  
gotoxy(10, 8);  
cout << “Enter the Expense Code “;  
gotoxy(10, 9);  
cout <<“1 Salary to Worker “;  
gotoxy(10, 10);  
cout <<“2 Electricity bill “;  
gotoxy(10, 11);  
cout <<“3 Telephone bill “;  
gotoxy(10, 12);  
cout <<“4 Miscellaneous “;

gotoxy(10, 13);  
cout <<“Enter your Choice”;

gotoxy(10, 14);  
cout << “Enter the date ” ;  
gotoxy(30, 13);  
cin >> tO\_code;  
gotoxy(30, 14);  
cin >> xdd;  
gotoxy(32, 14);  
cout << “-“;  
gotoxy(33, 14);  
cin >> xmm;  
gotoxy(35, 14);  
cout << “-“;  
gotoxy(36, 14);  
cin >> xyy;  
if (tO\_code < 1)  
{  
gotoxy(5, 23);  
cout << “Your entry is not valid”;  
getch();  
return;  
}  
else  
if (tO\_code > 4)  
{  
gotoxy(5, 23);  
cout << “Your entry is not valid”;  
getch();  
return;  
}

clrscr();

gotoxy(25, 2);  
cout << “Other Expenses in Music Shop”;  
gotoxy(25, 3);  
cout << “============================”;  
int d1, m1, y1;  
struct date d;        // For extracting system date  
getdate(&d);  
d1 = d.da\_day;  
m1 = d.da\_mon;  
y1 = d.da\_year;

gotoxy(62, 3);  
cout << “Date: ” << d1 << “/” << m1 << “/” << y1;

gotoxy(1, 5);  
for (int j = 1; j <= 79; j++)  
cout << “=”;

gotoxy(1, 6);  
cout << “Code #”;  
gotoxy(9, 6);  
cout << “Nature Expense”;  
gotoxy(34, 6);  
cout << “Date”;  
gotoxy(55, 6);  
cout << “Amount”;

gotoxy(1, 7);  
for (j = 1; j <= 79; j++)  
cout << “=”;

file.open(“other.dat”, ios::in);  
file.seekg(0,ios::beg);  
int row = 8;

// Reads all the records to display on the screen  
while (file.read((char \*)this, sizeof(other)))  
{

if ((tO\_code == O\_code) && (xmm == mm) && (xyy == yy))  
{  
flag = 0;  
delay(2);  
gotoxy(3, row);

cout << O\_code;  
gotoxy(9, row);  
puts(Nat\_Expen);  
gotoxy(34, row);  
cout << dd << “-” << mm << “-” << yy;  
gotoxy(55, row);  
cout << setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< amount;

++row;  
if (row > 23)  
{  
flag = 1;  
row = 6;  
gotoxy(4, 24);  
cout << “Press any key to continue…. “;  
getch();  
clrscr();  
}  
}  
}

++row;  
gotoxy(1, row);  
for (j = 1; j <= 79; j++)  
cout << “=”;  
row++;

file.close();  
if (!flag)  
{  
gotoxy(4, 24);  
cout << “Press any key to continue…”;  
getch();  
}  
getch();  
}

/\* Function for clearing specified row and column. \*/  
void account::clear(int col, int row)  
{  
for (int j = col; j <= 79; j++)  
{  
gotoxy(j, row);  
cout << ” “;  
}  
}

/\* Function for return name of the account  
holder from cassettes.dat. \*/  
char \*cassettes::return\_name(int t\_cas\_code)  
{  
fstream file;  
file.open(“cassettes.dat”, ios::in);  
file.seekg(0, ios::beg);  
char t\_Cass\_name[30];

// Return the name to display at report screen if found  
while (file.read((char \*)this, sizeof(cassettes)))  
{  
if (cas\_code == t\_cas\_code)  
{  
strcpy(t\_Cass\_name, Cass\_name);  
break;  
}  
}  
file.close();  
return t\_Cass\_name;  
}

/\* Function for return Comp\_name of the account holder from cassettes.dat. \*/  
char \*cassettes::return\_address(int t\_cas\_code)  
{  
fstream file;  
file.open(“cassettes.dat”, ios::in);  
file.seekg(0, ios::beg);  
char t\_Comp\_name[30];

// Return the Comp\_name to display at report screen if found  
while (file.read((char \*)this, sizeof(cassettes)))  
{  
if (cas\_code == t\_cas\_code)  
{  
strcpy(t\_Comp\_name, Comp\_name);  
break;  
}  
}  
file.close();  
return t\_Comp\_name;  
}

void balance::monthly\_profit()  
{

clrscr();  
int flag;  
fstream file;  
gotoxy(25,1);

cout << “Monthly Profit Statement”;  
gotoxy(25, 2);  
cout << “========================”;

int d1, m1, y1;  
struct date d;        // For extracting system date  
getdate(&d);

d1 = d.da\_day;  
m1 = d.da\_mon;  
y1 = d.da\_year;

float T\_bal = 0.0;  
float G\_bal = 0.0;  
gotoxy(55, 3);  
cout << “As on Date: ” << d1 << “/” << m1 << “/” << y1;

gotoxy(1, 4);  
for (int j = 1; j <= 79; j++)  
cout << “=”;

gotoxy(1, 5);  
cout << “Code #”;  
gotoxy(15, 5);  
cout << “Total Balance”;  
gotoxy(40, 5);  
cout << “Date”;

gotoxy(50, 5);  
cout << “Unit Price”;

gotoxy(65, 5);  
cout << “Total Amount”;

gotoxy(1, 6);  
for (j = 1; j <= 79; j++)  
cout << “=”;

file.open(“Tbal.dat”, ios::in);  
file.seekg(0,ios::beg);  
int row = 7;

float cod1=0.0;  
float cod2=0.0;  
float cod3=0.0;  
// Reads all the records to display on the screen  
while (file.read((char \*)this, sizeof(balance)))  
{  
flag = 0;  
if (cas\_code == 1)  
cod1 = price;  
else  
if (cas\_code == 2)  
cod2 = price;  
else  
cod3 = price;  
T\_bal = cas\_bal \* price;  
delay(2);  
gotoxy(3, row);  
cout << cas\_code;  
gotoxy(20, row);  
cout << cas\_bal;  
gotoxy(40, row);  
cout << dd <<“-“<< mm << “-” << yy;  
gotoxy(53, row);  
cout << setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< price;  
G\_bal = G\_bal + T\_bal;

gotoxy(68, row);  
cout << setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< T\_bal;  
T\_bal = 0.0;  
row++;  
if (row > 23)  
{  
flag = 1;  
row = 6;  
gotoxy(4, 24);  
cout << “Press any key to continue…. “;  
getch();  
clrscr();  
}  
}

gotoxy(1, row);  
for (j = 1; j <= 79; j++)  
cout << “=”;  
row++;  
gotoxy(50, row);  
cout << “Grand Total”;  
gotoxy(68, row);  
cout << setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< G\_bal;

file.close();  
row++;  
float saltotal = 0.0;  
account a;  
saltotal = a.sales\_status(m1, y1, cod1, cod2, cod3);    // Will display the sales status  
row++;

gotoxy(1, row);  
for (j = 1; j <= 79; j++)  
cout << “=”;  
row++;  
gotoxy(5, row);  
cout << “This month’s Sales Amount”;  
gotoxy(50, row);  
cout << setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< saltotal;

float gother = 0.0;  
other oth;  
gother =  oth.other\_status(m1, y1);  
row = row + 2;  
gotoxy(5, row);  
cout << “Other expenses “;  
gotoxy(50, row);  
cout << setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< gother;

row++;

gotoxy(1, row);  
for (j = 1; j <= 79; j++)  
cout << “=”;  
row++;  
gotoxy(5, row);  
cout << “Net Profit”;

float net\_bal = 0.0;  
net\_bal = (G\_bal + saltotal) – gother;  
gotoxy(50, row);  
cout << setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< net\_bal;

if (!flag)  
{  
gotoxy(4, 24);  
cout << “Press any key to continue…”;  
getch();  
}  
getch();  
}

// Function to find the sales total value  
float account::sales\_status(int m1, int y1, float cod1, float cod2, float cod3)  
{  
fstream file;  
file.open(“Person.dat”, ios::in);  
file.seekg(0,ios::beg);  
// Reads all the records to display on the screen  
float ts = 0.0;  
float gts = 0.0;  
while (file.read((char \*)this, sizeof(account)))  
{  
if ((m1 == mm) && (y1 == yy))  
{  
if (cas\_code == 1)  
ts = No\_cass \* cod1;  
else  
if (cas\_code == 2)  
ts = No\_cass \* cod2;  
else  
ts = No\_cass \* cod3;  
gts = gts + ts;  
ts = 0;  
}  
}  
file.close();  
return gts;  
}

// Function to find the other status  
float other::other\_status(int m1, int y1)  
{  
fstream file;  
file.open(“other.dat”, ios::in);  
file.seekg(0,ios::beg);  
// Reads all the records to display on the screen  
float goth = 0.0;  
while (file.read((char \*)this, sizeof(other)))  
{  
if ((m1 == mm) && (y1 == yy))  
{  
goth = goth + amount;  
}  
}  
file.close();  
return goth;  
}

// Function to display the balance cassette in music shop

void balance::balance\_cassette(void)  
{

clrscr();  
int flag;  
fstream file;  
gotoxy(25,1);

cout << “List of Balance Cassettes”;  
gotoxy(25, 2);  
cout << “=========================”;  
int d1, m1, y1;  
struct date d;        // For extracting system date  
getdate(&d);  
d1 = d.da\_day;  
m1 = d.da\_mon;  
y1 = d.da\_year;

float T\_bal = 0.0;  
float G\_bal = 0.0;  
gotoxy(55, 3);  
cout << “As on Date: ” << d1 << “/” << m1 << “/” << y1;

gotoxy(1, 4);  
for (int j = 1; j <= 79; j++)  
cout << “=”;

gotoxy(1, 5);  
cout << “Code #”;  
gotoxy(15, 5);  
cout << “Total Balance”;  
gotoxy(40, 5);  
cout << “Date”;

gotoxy(50, 5);  
cout << “Unit Price”;

gotoxy(65, 5);  
cout << “Total Amount”;

gotoxy(1, 6);  
for (j = 1; j <= 79; j++)  
cout << “=”;

file.open(“Tbal.dat”, ios::in);  
file.seekg(0,ios::beg);  
int row = 7;

// Reads all the records to display on the screen  
while (file.read((char \*)this, sizeof(balance)))  
{  
flag = 0;  
T\_bal = cas\_bal \* price;  
delay(2);  
gotoxy(3, row);  
cout << cas\_code;  
gotoxy(20, row);  
cout << cas\_bal;  
gotoxy(40, row);  
cout << dd <<“-“<< mm << “-” << yy;  
gotoxy(53, row);  
cout << setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< price;  
G\_bal = G\_bal + T\_bal;

gotoxy(68, row);  
cout << setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< T\_bal;  
T\_bal = 0.0;  
row++;  
if (row > 23)  
{  
flag = 1;  
row = 6;  
gotoxy(4, 24);  
cout << “Press any key to continue…. “;  
getch();  
clrscr();  
}  
}

gotoxy(1, row);  
for (j = 1; j <= 79; j++)  
cout << “=”;  
row++;  
gotoxy(50, row);  
cout << “Grand Total”;  
gotoxy(68, row);  
cout << setw(15)  
<< setprecision(2)  
<< setiosflags(ios::left)  
<< setiosflags(ios::showpoint)  
<< setiosflags(ios::fixed)  
<< G\_bal;

file.close();  
if (!flag)  
{  
gotoxy(4, 24);  
cout << “Press any key to continue…”;  
getch();  
}  
getch();  
}

/\* Function for recording the new cassettes in shop \*/  
void cassettes::new\_cassettes(void)  
{  
char ch;  
int i, valid;  
clrscr();

// Declaring the class objects for different operations with  
// member functions

account a;  
balance bal;  
cassettes ini;

shape s;  
s.box(2, 1, 79, 25, 218);  
s.box(25, 2, 54, 4, 219);

gotoxy(65, 2);  
cout << “<0>=Exit”;

gotoxy(3,3);  
for (i = 1; i<= 76; i++)  
cprintf(” “);  
textbackground(BLACK);  
textcolor(BLACK+BLINK);  
textbackground(WHITE);  
gotoxy(30, 3);  
cprintf(“Add New Cassette”);  
textcolor(LIGHTGRAY);  
textbackground(BLACK);  
int d1, m1, y1;        // Declare the variable for purchase date of cassette  
struct date d;        // For extracting system date

getdate(&d);        // Extract the system.date  
d1 = d.da\_day;  
m1 = d.da\_mon;  
y1 = d.da\_year;

// Declaring the temporary variables for cassette entry

char c\_code[10];

int t\_cas\_code;  
char t\_Cass\_name[30];  
char t\_Comp\_name[30];  
int t\_cas;  
int tdd, tmm, tyy;  
float t\_price;

t\_cas\_code = ini.last\_cas\_code();

// Appends and deletes a false record  
// to create primary position in data files  
if (t\_cas\_code == 0)  
{  
ini.add\_to\_file(t\_cas\_code, “abc”, “xyz”, 1, 1, 1, 1, 1.1);  
ini.delete\_cassette(t\_cas\_code);  
bal.add\_to\_file(t\_cas\_code, 0, 0, 0, 0, 0.0);  
bal.delete\_balance(t\_cas\_code);  
}

gotoxy(4, 6);  
cout << “Date: ” << d1 << ‘/’ << m1 << ‘/’ << y1;

gotoxy(5, 8);  
cout << “Cassette Code # “;  
gotoxy(5, 10);  
cout << “Cassette Name : “;  
gotoxy(5, 12);  
cout << “Company Name : “;  
gotoxy(5, 14);  
cout << “Total Cassettes: “;  
gotoxy(5, 16);  
cout << “Individual Cassette price : “;  
gotoxy(5, 18);  
cout << “Enter the Date “;

// Steps to enter the cassette code either 1 – 2 – 3.  
do  
{  
a.clear(22, 8);  
a.clear(5, 23);        // Clears the buttom message  
gotoxy(5, 23);  
cout << “Enter cassettes Code either 1/2/3 “;  
valid = 1;  
gotoxy(22, 8);  
gets(c\_code);  
t\_cas\_code = atoi(c\_code);

if (t\_cas\_code <= 0)  
{  
valid = 0;  
a.clear(5, 23);  
gotoxy(5, 23);  
cprintf(“\7Should not other than 1, 2, 3”);  
getch();  
gotoxy(5, 23);  
cout << “Enter cassettes Code either 1/2/3 “;

}  
} while (!valid);  
a.clear(5, 23);

// Steps to enter the Cassette name  
do  
{  
a.clear(22, 10);  
a.clear(5, 23);  
gotoxy(5, 23);  
cout << “Enter Name of the Cassette”;  
valid = 1;  
gotoxy(22, 10);  
gets(t\_Cass\_name);  
strupr(t\_Cass\_name);  
if (t\_Cass\_name[0] == ‘0’)  
return;  
if (strlen(t\_Cass\_name) == 0 || strlen(t\_Cass\_name) > 25)  
{  
valid = 0;  
gotoxy(5, 23);  
cprintf(“\7Cassette Name should not greater than 25”);  
getch();  
}  
}while (!valid);  
a.clear(5, 23);

// Steps to enter company name  
do  
{  
a.clear(22, 12);  
a.clear(5, 23);  
gotoxy(5, 23);  
cout << “Enter Company Name “;  
valid = 1;  
gotoxy(22, 12);  
gets(t\_Comp\_name);  
strupr(t\_Comp\_name);  
if (t\_Comp\_name[0] == ‘0’)  
return;  
if (strlen(t\_Comp\_name) == 0 || strlen(t\_Comp\_name) > 25)  
{  
valid = 0;  
gotoxy(5, 23);  
cprintf(“\7Company Name should not greater than 25”);  
getch();  
}  
}while (!valid);  
a.clear(5, 23);

// Steps to enter the total cassettes purchased  
do  
{  
a.clear(22, 14);  
a.clear(5, 23);  
gotoxy(5, 23);  
cout << “Enter Total number of cassettes purchased”;  
valid = 1;  
gotoxy(22, 14);  
gets(c\_code);  
t\_cas = atoi(c\_code);  
if (t\_cas == 0)  
{  
valid = 0;  
a.clear(5, 23);  
gotoxy(5, 23);  
cprintf(“\7Enter valid cassette number”);  
getch();  
gotoxy(5, 23);  
cout << “Enter Total number of cassettes purchased”;

}  
}while (!valid);  
a.clear(5, 23);

// Steps to enter the cassette/price  
do  
{  
a.clear(35, 16);  
a.clear(5, 23);  
gotoxy(5, 23);  
cout << “Enter individual cassette price”;  
valid = 1;  
gotoxy(35, 16);  
gets(c\_code);  
t\_price = atof(c\_code);  
if (t\_price <= 0)  
{  
valid = 0;  
gotoxy(5, 23);  
cprintf(“\7Enter valid price for cassette”);  
getch();  
}  
}while (!valid);  
a.clear(5, 23);

// Steps to enter the purchase date  
a.clear(22, 18);  
a.clear(5, 23);  
gotoxy(5, 23);  
cout << “Enter date on which the cassette is purchase”;  
valid = 1;  
gotoxy(22, 18);  
cin >> tdd;  
gotoxy(25, 18);  
cout << “-“;  
gotoxy(26, 18);  
cin >> tmm;  
gotoxy(28, 18);  
cout << “-“;  
gotoxy(29, 18);  
cin >> tyy;  
a.clear(5, 23);

do  
{  
a.clear(5, 20);  
valid = 1;  
gotoxy(5, 20);  
cout << “Do you want to save the record <Y/N>: “;  
ch = getche();  
if (ch == ‘0’)  
return;  
ch = toupper(ch);  
}while (ch != ‘N’ && ch != ‘Y’);

if (ch == ‘N’)  
return;

// Appends the records contents into  
// both cassettes.dat and person.dat data files  
ini.add\_to\_file(t\_cas\_code, t\_Cass\_name, t\_Comp\_name,  
t\_cas, tdd, tmm, tyy, t\_price);

bal.add\_to\_file(t\_cas\_code, t\_cas, tdd, tmm, tyy, t\_price);  
}

/\* This function update\_balance() is used to update the  
balance cassette in Tbal.dat data file \*/

void balance::Update\_balance(int t\_cas\_code,  
int t\_cas, int tdd, int tmm, int tyy, float t\_price)  
{  
int bal = give\_balance(t\_cas\_code);

if (bal > 0)  
{  
fstream file;  
file.open(“Tbal.dat”, ios::in);  
fstream temp;  
temp.open(“TEMP.dat”, ios::out);  
file.seekg(0,ios::beg);

// Uses a copy method to delete the account from Tbal.dat data file  
while (!file.eof())  
{  
file.read((char \*)this, sizeof(balance));  
if (file.eof())  
break;  
if (cas\_code != t\_cas\_code)  
temp.write((char \*)this, sizeof(balance));  
else  
{  
cas\_code = t\_cas\_code;  
cas\_bal = bal – t\_cas;  
dd = tdd;  
mm = tmm;  
yy = tyy;  
price = t\_price;  
temp.write((char \*)this, sizeof(balance));  
}  
}

file.close();  
temp.close();  
file.open(“Tbal.dat”, ios::out);  
temp.open(“TEMP.dat”, ios::in);  
temp.seekg(0, ios::beg);

// Copy the TEMP.dat contents into Tbal.dat data file  
while (!temp.eof())  
{  
temp.read((char \*)this, sizeof(balance));  
if (temp.eof())  
break;  
file.write((char \*)this, sizeof(balance));  
}

file.close();  
temp.close();  
}  
else  
{  
cas\_code = t\_cas\_code;  
cas\_bal = t\_cas;  
dd = tdd;  
mm = tmm;  
yy = tyy;  
price = t\_price;  
fstream file;

// Appends new account record with the price  
// into cassettes.dat data file  
file.open(“Tbal.dat”, ios::out|ios::app);  
file.write((char \*)this, sizeof(balance));

file.close();  
}  
}

/\* Function for creating new account for new customer. \*/  
void account::new\_account(void)  
{  
char ch;  
int i, valid;  
clrscr();  
balance bal;

shape s;  
s.box(2, 1, 79, 25, 218);  
s.box(25, 2, 54, 4, 219);

gotoxy(65, 2);  
cout << “<0>=Exit”;

gotoxy(3,3);  
for (i = 1; i<= 76; i++)  
cprintf(” “);  
textbackground(BLACK);  
textcolor(BLACK+BLINK);  
textbackground(WHITE);  
gotoxy(30, 3);  
cprintf(“Sales of Cassettes”);  
textcolor(LIGHTGRAY);  
textbackground(BLACK);  
int d1, m1, y1;  
struct date d;        // For extracting system date  
getdate(&d);  
d1 = d.da\_day;  
m1 = d.da\_mon;  
y1 = d.da\_year;  
int t\_cas\_code;

int C\_bal = 0;  
float t\_price;  
char c\_code[2];  
char tP\_name[30];    // Person Name  
char tP\_address[30];    // Person Address  
int tNo\_cass;        // Number of cassette

t\_cas\_code = last\_cas\_code();

// Appends and deletes a false record to create  
// primary position in data files  
if (t\_cas\_code == 0)  
{  
add\_to\_file(t\_cas\_code, “xxxx”, “xxxxx”, 1, 1, 1, 1);  
delete\_account(t\_cas\_code);  
}

gotoxy(5, 6);  
cout << “Date: ” << d1 << ‘/’ << m1 << ‘/’ << y1;  
gotoxy(5, 8);  
cout << “Cassette Code “;  
gotoxy(5, 10);  
cout << “Name of Person : “;  
gotoxy(5, 12);  
cout << “Address : “;

gotoxy(5, 14);  
cout << “Total Cassette : “;

// Steps to enter the cassette code either 1 – 2 – 3.  
do  
{  
clear(22, 8);  
clear(5, 23);        // Clears the buttom message  
gotoxy(5, 23);  
cout << “Enter cassettes Code either 1/2/3 “;  
valid = 1;  
gotoxy(22, 8);  
gets(c\_code);  
t\_cas\_code = atoi(c\_code);

if (t\_cas\_code <= 0)  
{  
valid = 0;  
clear(5, 23);  
gotoxy(5, 23);  
cprintf(“\7Should not other than 1, 2, 3”);  
getch();  
gotoxy(5, 23);  
cout << “Enter cassettes Code either 1/2/3 “;  
}  
else  
{  
C\_bal = bal.give\_balance(t\_cas\_code);  
t\_price = bal.return\_price(t\_cas\_code);  
if (C\_bal > 0)  
{  
gotoxy(40, 8);  
cout << “Total cassettes in Shop : ” << C\_bal;  
break;  
}  
else  
{  
cout << “There is no cassette in shop”;  
return;  
}  
}  
} while (!valid);  
clear(5, 23);

// Steps to enter the name of the purchase person  
do  
{  
clear(22, 10);  
clear(5, 23);  
gotoxy(5, 23);  
cout << “Enter Name of the Person”;  
valid = 1;  
gotoxy(22, 10);  
gets(tP\_name);  
strupr(tP\_name);  
if (tP\_name[0] == ‘0’)  
return;  
if (strlen(tP\_name) == 0 || strlen(tP\_name) > 25)  
{  
valid = 0;  
gotoxy(5, 23);  
cprintf(“\7Name should not greater than 25”);  
getch();  
clear(5, 23);  
}

}while (!valid);  
clear(5, 23);

// Steps to enter the address of the purchase person  
do  
{  
clear(22, 12);  
clear(5, 23);  
gotoxy(5, 23);  
cout << “Enter Name of the Person”;  
valid = 1;  
gotoxy(22, 12);  
gets(tP\_address);  
strupr(tP\_address);  
if (tP\_address[0] == ‘0’)  
return;  
if (strlen(tP\_address) == 0 || strlen(tP\_address) > 25)  
{  
valid = 0;  
gotoxy(5, 23);  
cprintf(“\7Address should not greater than 25”);  
getch();  
clear(5, 23);  
}

}while (!valid);  
clear(5, 23);

do  
{  
clear(22, 14);  
clear(5, 23);  
gotoxy(5, 23);  
cout << “No. of cassette “;  
valid = 1;  
gotoxy(22, 14);  
cin >> tNo\_cass;  
if (tNo\_cass > C\_bal)  
{  
valid = 0;  
gotoxy(5, 23);  
cprintf(“\7The availability of cassette is not sufficient”);  
getch();  
}  
}while (!valid);  
clear(5, 23);

do  
{  
clear(5, 17);  
valid = 1;  
gotoxy(5, 17);  
cout << “Do you want to save the record <Y/N>: “;  
ch = getche();  
if (ch == ‘0’)  
return;  
ch = toupper(ch);  
}while (ch != ‘N’ && ch != ‘Y’);

if (ch == ‘N’)  
return;

// Updates the new balance after deducting the current sales

// Appends the records contents into both person.dat data files  
add\_to\_file(t\_cas\_code, tP\_name, tP\_address, tNo\_cass, d1, m1,  y1);

bal.Update\_balance(t\_cas\_code, tNo\_cass, d1, m1, y1, t\_price);  
}

/\* Function for enter the other expences in other.dat data file. \*/  
void other::Expense\_other(void)  
{  
char ch;  
int i, valid;  
clrscr();  
account a;

shape s;  
s.box(2, 1, 79, 25, 218);  
s.box(25, 2, 54, 4, 219);

gotoxy(65, 2);  
cout << “<0>=Exit”;  
gotoxy(3,3);  
for (i = 1; i<= 76; i++)  
cprintf(” “);  
textbackground(BLACK);  
textcolor(BLACK+BLINK);  
textbackground(WHITE);  
gotoxy(30, 3);  
cprintf(“Other Expence in Shop”);  
textcolor(LIGHTGRAY);  
textbackground(BLACK);  
int d1, m1, y1;  
struct date d;        // For extracting system date  
getdate(&d);  
d1 = d.da\_day;  
m1 = d.da\_mon;  
y1 = d.da\_year;

char c\_code[2];  
int tO\_code;        // Other expence code  
char tNat\_Expen[30];    // Nature of expense description  
float tamount;    // Expense amount

gotoxy(5, 6);  
cout << “Date: ” << d1 << ‘/’ << m1 << ‘/’ << y1;  
gotoxy(5, 8);  
cout << “Expense Code “;  
gotoxy(5, 10);  
cout << “Nature of Expense : “;  
gotoxy(5, 12);  
cout << “Amount : “;

// Steps to enter the other expense code  
do  
{  
a.clear(22, 8);  
a.clear(5, 23);        // Clears the buttom message  
gotoxy(5, 23);  
cout << “Enter Expense code Code either 1/2/3/4 “;  
valid = 1;  
gotoxy(22, 8);  
gets(c\_code);  
tO\_code = atoi(c\_code);

if (tO\_code <= 0)  
{  
valid = 0;  
a.clear(5, 23);  
gotoxy(5, 23);  
cprintf(“\7Expence code should not other than 1/2/3/4”);  
getch();  
gotoxy(5, 23);  
cout << “Enter other expense Codes either 1/2/3/4 “;  
}  
} while (!valid);  
a.clear(5, 23);

// Steps to enter the nature of expense  
do  
{  
a.clear(22, 10);  
a.clear(5, 23);  
gotoxy(5, 23);  
cout << “Enter the Nature of expense”;  
valid = 1;  
gotoxy(22, 10);  
gets(tNat\_Expen);  
strupr(tNat\_Expen);  
if (tNat\_Expen[0] == ‘0’)  
return;  
if (strlen(tNat\_Expen) == 0 || strlen(tNat\_Expen) > 25)  
{  
valid = 0;  
gotoxy(5, 23);  
cprintf(“\7Nature of expense not greater than 25”);  
getch();  
a.clear(5, 23);  
}

}while (!valid);  
a.clear(5, 23);

gotoxy(22, 12);  
cin >> tamount;

do  
{  
a.clear(5, 17);  
valid = 1;  
gotoxy(5, 17);  
cout << “Do you want to save the record <Y/N>: “;  
ch = getche();  
if (ch == ‘0’)  
return;  
ch = toupper(ch);  
}while (ch != ‘N’ && ch != ‘Y’);

if (ch == ‘N’)  
return;

// Appends the records contents into both person.dat data files  
if (tamount > 0)  
add\_to\_file(tO\_code, tNat\_Expen, d1, m1,  y1, tamount);  
}

/\* Function for returning the record no. for updating price \*/  
int cassettes::recordno(int t\_cas\_code)  
{  
fstream file;  
file.open(“cassettes.dat”, ios::in);  
file.seekg(0, ios::beg);  
int count = 0;

// Finds the record position in cassettes.dat data file  
while (file.read((char \*)this, sizeof(cassettes)))  
{  
count++;  
if (t\_cas\_code == cas\_code)  
break;  
}  
file.close();  
return count;  
}

/\* Function for closing any account after inputing account number. \*/  
void account::close\_account(void)  
{  
clrscr();  
char t\_acc[10];  
int t, t\_cas\_code;  
gotoxy(71, 1);  
cout << “<0>=Exit”;  
gotoxy(5, 5);  
cout << “Enter the account no. “;  
gets(t\_acc);  
t = atoi(t\_acc);  
t\_cas\_code = t;  
if (t\_cas\_code == 0)  
return;  
clrscr();  
cassettes ini;  
balance bal;

if (!bal.give\_balance(t\_cas\_code))  
{  
gotoxy(5, 5);  
cout << “\7Account not found “;  
getch();  
return;  
}  
gotoxy(71, 1);  
cout << “<0>=Exit”;  
gotoxy(3, 3);  
textbackground(WHITE);  
for (int i = 1; i <= 76; i++)  
cprintf(” “);  
textbackground(BLACK);  
textcolor(BLACK+BLINK);  
textbackground(WHITE);  
gotoxy(30, 3);  
cprintf(“Close account screen”);  
textcolor(LIGHTGRAY);  
textbackground(BLACK);  
int d1, m1, y1;  
struct date d;  
getdate(&d);  
d1 = d.da\_day;  
m1 = d.da\_mon;  
y1 = d.da\_year;  
gotoxy(5, 6);  
cout << “Date: ” << d1 << “/” << m1 << “/” << y1;  
char ch;

ini. display(t\_cas\_code);

do  
{  
clear(5, 15);  
gotoxy(5, 15);  
cout << “Close this account <y/n?? “;  
ch = getche();  
if (ch == ‘0’)  
return;  
ch = toupper(ch);  
}while (ch != ‘N’ && ch != ‘Y’);

if (ch == ‘N’)  
return;

// Function calls to delete the existing account no.  
ini.delete\_cassette(t\_cas\_code);  
delete\_account(t\_cas\_code);  
gotoxy(5, 20);  
cout << “\7Account Deleted”;  
gotoxy(5, 23);  
cout << “Press any key to continue…”;  
getch();  
}

// Main program logic which control the class members and member functions.  
void main(void)  
{  
main\_menu  m\_menu;  
int gdriver = DETECT, gmode, errorcode;  
initgraph(&gdriver, &gmode, “c:\\tc\\bgi”);

m\_menu.help();  
closegraph();  
m\_menu.control\_menu();  
}