#include <iostream>

#include <string>

#include<conio.h>

#include<iomanip>

#include<fstream>

using namespace std;

class person

{

protected:

string name;

int gender;

int age;

public:

void get\_person\_info()

{

//STRINGS\*\*\*\*\*\*\*\*\*\*\*\*

cout << "name:";

cin.ignore();//to ignore the new line buffer

getline(cin, name);

cout << "age:";

cin >> age;

cout << "gender:\n1.male\n2.female" << endl;

cout << "enter your gender:";

cin >> gender;

}

void display\_peron\_info()

{

cout << "name: " << name << endl;

cout << "age: " << age << endl;

cout << "1.male\n2.female" << endl;

cout << "gender: " << gender << endl;

}

//friend function\*\*\*\*\*\*\*\*\*\*\*

friend int return\_gender(person p);

//to return gender which will be used furthe (while accessing OPTION 2) that is gym member login....

//and will accordingly provide data for dietplan and workout plan....

};

//inline fn\*\*\*\*\*\*\*\*\*\*\*\*//replace fn call with body...

inline int return\_gender(person p)//friend function

{

return p.gender;

}

class membership :public person

{

protected:

string membership\_type;

double fee;

string timeslot;

public:

//pure virtual function\*\*\*\*\*\*\*\*\*\*\*\*

//(membership)---->(gymmember)--->--->DIETPLAN AND WORKOUTPLAN

virtual void displayplan() = 0;

void get\_membership\_info()

{

int mem\_tp\_choice;

cout << endl;

cout << "ENTER MEMBERSHIP DETAILS" << endl;

cout << "1.basic-1000" << endl;

cout << ":one month regular strength\n" << endl;

cout << "2.premium-2000" << endl;

cout << ":two months + free one month regular strength" << endl;

cout << ":access to cardio machines\n" << endl;

cout << "3.vip-3000" << endl;

cout << ":three months + free one month regular strength" << endl;

cout << ":access to cardio machines" << endl;

cout << ":persnol gym trainer\n" << endl;

cout << "enter membership type:";

cin >> mem\_tp\_choice;

if (mem\_tp\_choice == 1)

{

membership\_type = "Basic";

fee = 1000;

}

else if (mem\_tp\_choice == 2)

{

membership\_type = "Premium";

fee = 2000;

}

else if (mem\_tp\_choice == 3)

{

membership\_type = "VIP";

fee = 3000;

}

else

{

cout << "Invalid input!" << endl;

cout << "setting your membership to basic!" << endl;

membership\_type = "Basic";

fee = 1000;

}

cout << "membership type:" << membership\_type << endl;

cout << "fee: " << fee << endl;

}

void display\_membership\_info()

{

cout << "membership type: " << membership\_type << endl;

cout << "fee: " << fee << endl;

}

void gym\_services\_info()

{

cout << "\nOUR SERVICES:" << endl;

cout << ":weight gain / weight loss" << endl;

cout << ":cardio" << endl;

cout << ":body fitness" << endl;

cout << ":personal training" << endl;

cout << ":personalized diet plan\n" << endl;

}

void gym\_timing\_info()

{

cout << "gym timing:" << endl;

cout << "MORNING: 6.00am-10.30am" << endl;

cout << "EVENING: 4.00pm-10.00pm" << endl;

cout << "sunday closed!\n" << endl;

}

};

//templates\*\*\*\*\*\*\*\*\*\*\*\*

//(inside the int main....while creating objects of the class...two different datamembers....int and string are passed)

template <typename T1, typename T2>

class trainer :public person

{

protected:

T1 trainer\_id;

T2 speacility;

public:

void get\_trainer\_info()

{

static int t\_id = 100;

get\_person\_info();

trainer\_id = t\_id;

t\_id++;

cout << "speacility: ";

cin.ignore();

getline(cin, speacility);

cout << "trainer ID assigned to you:" << trainer\_id << endl;

}

int get\_trainerid()const

{

return trainer\_id;

}

void display\_trainer\_info()

{

cout << "trainer id: " << trainer\_id << endl;

display\_peron\_info();

cout << "spacility: " << speacility << endl;

}

};

class suppliments\_and\_accessories

{

protected:

string sup\_name[5];

string acc\_name[5];

double sup\_price[5];

double acc\_price[5];

double total\_bill = 0;

public:

//constructor\*\*\*\*\*\*\*\*\*\*\*\*

//will be invoked automatically while creating objects of this class...

suppliments\_and\_accessories()

{

sup\_name[0] = "POTEIN POWDER";

sup\_name[1] = "CRETINE";

sup\_name[2] = "MULTIVITAMIN";

sup\_name[3] = "BCAA";

sup\_name[4] = "FISH OIL";

sup\_price[0] = 2000;

sup\_price[1] = 600;

sup\_price[2] = 300;

sup\_price[3] = 800;

sup\_price[4] = 400;

acc\_name[0] = "GYM\_TOWEL";

acc\_name[1] = "YOGAMAT";

acc\_name[2] = "RESISTANCE BAND";

acc\_name[3] = "WRIST SUPPORTER";

acc\_name[4] = "SHAKER BOTTLE";

acc\_price[0] = 200;

acc\_price[1] = 600;

acc\_price[2] = 400;

acc\_price[3] = 300;

acc\_price[4] = 800;

}

//destructor\*\*\*\*\*\*\*\*\*\*\*\*

~suppliments\_and\_accessories() { }

void display\_suppliments\_info()

{

for (int i = 0; i < 5; i++)

{

cout << i + 1 << " " << sup\_name[i] << " price: " << sup\_price[i] << endl;

}

}

void display\_accessories\_info()

{

for (int i = 0; i < 5; i++)

{

cout << i + 1 << " " << acc\_name[i] << " price: " << acc\_price[i] << endl;

}

}

void buy\_sup()

{

int choice, quantity;

int option;

do

{

display\_suppliments\_info();

cout << "Enter choice: ";

cin >> choice;

if (choice < 1 || choice > 5)

{

cout << "Invalid choice!" << endl;

continue;

}

cout << "Enter quantity: ";

cin >> quantity;

total\_bill = total\_bill + sup\_price[choice - 1] \* quantity;

cout << "Want to buy more supplements?\n1.yes\n2.no" << endl;

cout << "enter your choice:";

cin >> option;

} while (option == 1);

}

void buy\_acc()

{

int choice, quantity;

int optionnn;

do

{

display\_accessories\_info();

cout << "Enter choice: ";

cin >> choice;

if (choice < 1 || choice > 5)

{

cout << "Invalid choice!" << endl;

continue;

}

cout << "Enter quantity: ";

cin >> quantity;

total\_bill = total\_bill + acc\_price[choice - 1] \* quantity;

cout << "Want to buy more accessories?\n1.yes\n2.no" << endl;

cout << "enter your choice:";

cin >> optionnn;

} while (optionnn == 1);

}

//operator overloading\*\*\*\*\*\*\*\*\*\*\*\*

//(first obj will be used for storing bill info for suppliments)...(second object will be use for storing bill for accessories)...

//OBJECT\_3=OBJECT\_1+OBJECT\_2...

suppliments\_and\_accessories operator+(suppliments\_and\_accessories SECOND);

void display\_total\_bill()

{

cout << "\nTOTAL BILL: " << total\_bill << endl;

}

};

suppliments\_and\_accessories suppliments\_and\_accessories::operator+(suppliments\_and\_accessories SECOND)

{

suppliments\_and\_accessories temp;

temp.total\_bill = total\_bill + SECOND.total\_bill;

return temp;

}

class gymmember : public membership, public suppliments\_and\_accessories

{

protected:

int memberid;

public:

//files\*\*\*\*\*\*\*\*\*\*\*\*

//(consepts of files to write data inside a file)...

void writememberdata()

{

ofstream fout("MemberDetails.txt", ios::app);

if (fout.is\_open())

{

fout << "Member ID:" << memberid << endl;

fout << "Name:" << name << endl;

fout << "Age:" << age << endl;

fout << "Membership Type:" << membership\_type << endl;

fout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

fout.close();

cout << "Member data saved succesfully In File" << endl;

}

else

{

cout << "Error:Could not open file!!" << endl;

}

}

//(consepts of files to display data inside a file)...

void displaymemberdata()

{

ifstream fin("MemberDetails.txt");

if (fin.is\_open())

{

string line;

cout << "Member Details:\n";

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

while (getline(fin, line))

cout << line << endl;

fin.close();

}

else

{

cout << "Error:Could not open file!!\n";

}

}

void displayplan()//(part of virtual function)

{

}

void get\_members\_info()

{

static int m\_id = 1000;

memberid = m\_id;

m\_id++;

gym\_services\_info();

gym\_timing\_info();

cout << "ENTER DETAILS:" << endl;

get\_person\_info();

get\_membership\_info();

cout << "member id assigned to you:" << memberid << endl;

}

int get\_memberid()const

{

return memberid;

}

void display\_member\_info()

{

cout << "member id: " << memberid << endl;

display\_peron\_info();

display\_membership\_info();

}

};

class DietPlan :public gymmember

{

protected:

int currentWeight;

int targetWeight;

int goal;

public:

void input\_dietplan\_Details()

{

cout << "Enter your current weight:";

cin >> currentWeight;

cout << "Do you want to lose or gain weight?\n1.lose\n2.gain: ";

cin >> goal;

//exception handling\*\*\*\*\*\*\*\*\*\*\*\*

//(will throw error for invalid input)...

try

{

if (goal != 1 && goal != 2)

{

throw(2);

}

else if (goal == 1)

{

cout << "goal: weight loss" << endl;

}

else if (goal == 2)

{

cout << "goal: weight gain" << endl;

}

}

catch (int j)

{

cout << "invalid input for goal!" << endl;

}

cout << "Enter your target weight: ";

cin >> targetWeight;

cout << endl;

}

void displayplan(int var)

{

input\_dietplan\_Details();

if (goal == 1)

{

if (currentWeight > targetWeight)

{

if (var == 1)

{

cout << "male loss" << endl;

cout << "1. Calorie intake:\nreduce calorie intake by 800cal\nmaximun calorie intake:1800-2000 cal/day\n" << endl;

cout << "2. Diet:\nBREAKFAST:\n(veg)-(nonveg) 1 bowl of fruits, green tea\n";

cout << "LUNCH:\n(veg) 1 cup dal , rice , 50 gm paneer , 1 plate sala\n(nonveg) chicken rice , 1 plate salad\n";

cout << "DINNER:\n(veg) 2 roti , 1 bowl paneer bhurji , 1 cup lo-fat curd\n(nonveg) 2 roti , 1 bowl egg bhurji , 1 cup lowfat curd\n" << endl;

cout << "3. Exercise:\nWeight lifting 4-5 times a week\n";

cout << "stay hydrated! (3-4 liter) of water/day\n" << endl;

}

else if (var == 2)

{

cout << "female loss" << endl;

cout << "1. Calorie intake:\nreduce calorie intake by 600cal\nmaximun calorie intake:1500-1600 cal/day\n" << endl;

cout << "2. Diet:\nBREAKFAST:\n(veg)-(nonveg) 1 bowl of fruits, green tea\n";

cout << "LUNCH:\n(veg) 1 cup dal , rice , 50 gm paneer , 1 plate sala\n(nonveg) chicken rice , 1 plate salad\n";

cout << "DINNER:\n(veg) 2 roti , 1 bowl paneer bhurji , 1 cup lo-fat curd\n(nonveg) 2 roti , 1 bowl egg bhurji , 1 cup lowfat curd\n" << endl;

cout << "3. Exercise:\nWeight lifting 4-5 times a week\n";

cout << "stay hydrated! (3-4 liter) of water/day\n" << endl;

}

}

else

{

cout << "\nYour target weight should be less than your current weight\n";

}

}

else if (goal == 2)

{

if (currentWeight < targetWeight)

{

if (var == 1)

{

cout << "male gain" << endl;

cout << "1. Calorie intake:\nincrease calorie intake by 1000cal\nminimum calorie intake:2800-3000 cal/day\n" << endl;

cout << "2. Diet:\nBREAKFAST:\n(veg) 1 cup oats , 1 banana , 15-20 almonds\n(nonveg) 1 cup oats , 2 boiled eggs , 10-15 almonds\n";

cout << "LUNCH:\n(veg) 1/2 cup brown rice , 1 bowl rajma , 1 plate salad\n(nonveg) chicken rice , 1 plate salad\n";

cout << "DINNER:\n(veg) 2 roti , 1 bowl paneer bhurji , 1 cup yogurt\n(nonveg) 2 roti , 1 bowl egg bhurji , 1 cup yogurt\n" << endl;

cout << "3. Exercise:\nWeight lifting 4-5 times a week\n";

cout << "stay hydrated! (3-4 liter) of water/day\n" << endl;

}

else if (var == 2)

{

cout << "female gain" << endl;

cout << "1. Calorie intake:\nincrease calorie intake by 600cal\nminimum calorie intake:1800-2000 cal/day\n" << endl;

cout << "2. Diet:\nBREAKFAST:\n(veg) 1 cup oats , bowl of fruits\n(nonveg) 1 cup oats , 2 boiled eggs\n";

cout << "LUNCH:\n(veg) 1/2 cup brown rice , 1 bowl rajma\n(nonveg) chicken rice , 1 plate salad\n";

cout << "DINNER:\n(veg) 2 roti , 1 bowl paneer bhurji , 1 cup yogurt\n(nonveg) 2 roti , 1 bowl egg bhurji , 1 cup yogurt\n" << endl;

cout << "3. Exercise:\nWeight lifting 4-5 times a week\n";

cout << "stay hydrated! (2-2.5 liter) of water/day\n" << endl;

}

}

else

{

cout << "\nYour target weight should be greater than your current weight\n";

}

}

else

{

cout << "invalid input while entering data!" << endl;

}

}

};

class WorkoutPlan :public gymmember

{

protected:

int workoutType;

public:

void input\_workoutplan\_Details()

{

cout << "\nEnter workout plan type:\n";

cout << "1.Weight Training\n";

cout << "2.Cardio\n";

cout << "Enter your choice:";

cin >> workoutType;

//exception handling\*\*\*\*\*\*\*\*\*\*\*\*

//(will throw error for invalid input)...

try

{

if (workoutType != 1 && workoutType != 2)

{

throw (1);

}

else if (workoutType == 1)

{

cout << "selected weight training plan" << endl;

}

else if (workoutType == 2)

{

cout << "selected cardio training plan" << endl;

}

}

catch (int i)

{

cout << "invalid input for workout plan!" << endl;

}

cout << endl;

}

void displayplan(int var)

{

input\_workoutplan\_Details();

if (workoutType == 1)

{

if (var == 1)

{

cout << "weight training male" << endl;

cout << "Monday: Chest and Triceps\n1.cable chest flies\n2.inclined dumbel press\n3.declined barbell press\n4.tricep pushdown\n5.overhead tricep extension\n" << endl;

cout << "Tuesday: Back and Biceps\n1.lats pulldown\n2.back cable row\n3.pull ups\n4.dumbell bisep curls\n5.preacher curls\n" << endl;

cout << "Wednesday: Legs\n1.squats\n2.legs extension\n3.leg press\n4.calf raises\n" << endl;

cout << "Thursday: Shoulders\n1.shoulder press\n2.lateral raise\n3.shrugs\n4.rear delt raises\n" << endl;

cout << "Friday: Core\n1.crunches\n2.v ups\n3.planks\n4.leg raises\n" << endl;

cout << "Saturday: strength lifting\n1.deadlift\n2.squats\n3.bench press\n" << endl;

cout << "Sunday: Rest Day\n" << endl;

}

else if (var == 2)

{

cout << "weight training female" << endl;

cout << "Monday: Chest and Triceps\n1.cable chest flies\n2.inclined dumbel press\n3.declined barbell press\n4.tricep pushdown\n5.overhead tricep extension\n" << endl;

cout << "Tuesday: Back and Biceps\n1.lats pulldown\n2.back cable row\n3.pull ups\n4.dumbell bisep curls\n5.preacher curls\n" << endl;

cout << "Wednesday: Legs\n1.squats\n2.legs extension\n3.leg press\n4.calf raises\n" << endl;

cout << "Thursday: Shoulders\n1.shoulder press\n2.lateral raise\n3.shrugs\n4.rear delt raises\n" << endl;

cout << "Friday: Core\n1.crunches\n2.v ups\n3.planks\n4.leg raises\n" << endl;

cout << "Saturday: strength lifting\n1.deadlift\n2.squats\n3.bench press\n" << endl;

cout << "Sunday: Rest Day\n" << endl;

}

}

else if (workoutType == 2)

{

if (var == 1)

{

cout << "cardio male" << endl;

cout << "Monday-thursday:\n1.walking:30 min\n2.running 30 min\n" << endl;

cout << "Tuesday-friday:\n1.Cycling:1 hour\n2.core exercise 30 min\n" << endl;

cout << "Wednesday-saturday:\n1.sprint:100 meter-5 times\n2.rope jump 10 min\n" << endl;

cout << "Sunday: Rest Day\n" << endl;

}

else if (var == 2)

{

cout << "cardio female" << endl;

cout << "Monday-thursday:\n1.walking:30 min\n2.running 15 min\n" << endl;

cout << "Tuesday-friday:\n1.Cycling:45 min\n2.core exercise 10 min\n" << endl;

cout << "Wednesday-saturday:\n1.sprint:100 meter-2 times\n2.rope jump 20 min\n" << endl;

cout << "Sunday: Rest Day\n" << endl;

}

}

else

{

cout << "Invalid input while entering data!" << endl;

}

}

};

//default arguments\*\*\*\*\*\*\*\*\*\*\*\*()

int calculate\_weight\_difference(int cw = 0, int lw = 0);

//function overloading(for int)

int calculate\_weight\_difference(int cw, int lw)

{

return cw - lw;

}

//default arguments\*\*\*\*\*\*\*\*\*\*\*\*()

float calculate\_weight\_difference(float cw = 0, float lw = 0);

//function overloading()for float

float calculate\_weight\_difference(float cw, float lw)

{

return cw - lw;

}

void display\_result(float wd)

{

if (wd < 0)

{

cout << "you lost " << -wd << " kg" << endl;

}

else if (wd > 0)

{

cout << "you gained " << wd << " kg" << endl;

}

else

{

cout << "same weight!" << endl;

}

}

void main()

{

cout << setfill(' ') << setw(60) << "" << endl;

cout << setw(33) << "GYM MANAGEMENT SYSTEM" << setw(15) << "" << endl;

cout << setfill(' ') << setw(70) << "\n" << endl;

const int max\_members = 100;

const int max\_trainers = 20;

gymmember m[max\_members];

//templates(passing two different data types)

trainer<int, string> t[max\_trainers];

int members\_count = 0;

int trainers\_count = 0;

int choicee;

do

{

int choice = 0;

cout << endl;

cout << setfill('-') << setw(43) << "" << endl;

cout << setw(30) << "WELCOME TO USER MENU" << setw(13) << "" << endl;

cout << setfill('-') << setw(44) << "\n" << endl;

cout << setfill(' ') << setw(10) << "" << "1. Admin" << endl;

cout << setw(10) << "" << "2. Gym Member" << endl;

cout << setw(10) << "" << "3. Exit\n" << endl;

cout << setfill('-') << setw(43) << "" << endl;

cout << "Enter your choice: ";

cin >> choicee;

if (choicee == 1)

{

cout << endl;

cout << "WELCOME ADMIN\n";

while (choice != 11)

{

cout << endl;

cout << "1. register Gym Member" << endl;

cout << "2. Add Trainer" << endl;

cout << "3. Display Perticular Gym Member Info" << endl;

cout << "4. Display Perticular Trainer Info" << endl;

cout << "5. Display all registered member" << endl;

cout << "6. Display all added Trainers" << endl;

cout << "7. Delete Gym Member" << endl;

cout << "8. Delete Trainer" << endl;

cout << "9. Renewing Gym Membership " << endl;

cout << "10. Display File Content (will display deleted members tooooo)..." << endl;//(will display deleted members tooooo)...

cout << "11. Exit" << endl;

cout << "Enter Your Choice: ";

cin >> choice;

switch (choice)

{

case 1:

if (members\_count >= max\_members)

{

cout << "members full!" << endl;

}

else

{

m[members\_count].get\_members\_info();

//files(data of perticular member will be written inside a file)...

m[members\_count].writememberdata();

members\_count++;

}

break;

case 2:

if (trainers\_count >= max\_trainers)

{

cout << "trainers full!" << endl;

}

else

{

cout << "enter trainer deails: " << endl;

t[trainers\_count].get\_trainer\_info();

trainers\_count++;

}

break;

case 3:

if (members\_count == 0)

{

cout << "no members to display!" << endl;

}

else

{

int m\_id;

cout << "enter member id:";

cin >> m\_id;

int m\_found;

m\_found = 0;

for (int i = 0; i < members\_count; i++)

{

if (m[i].get\_memberid() == m\_id)

{

m[i].display\_member\_info();

m\_found = 1;

break;

}

}

if (m\_found == 0)

{

cout << "member with such id not found!" << endl;

}

}

break;

case 4:

if (trainers\_count == 0)

{

cout << "no trainers to display!" << endl;

}

else

{

int t\_id;

cout << "enter trainer id:";

cin >> t\_id;

int t\_found;

t\_found = 0;

for (int i = 0; i < trainers\_count; i++)

{

if (t[i].get\_trainerid() == t\_id)

{

t[i].display\_trainer\_info();

t\_found = 1;

break;

}

}

if (t\_found == 0)

{

cout << "trainer with such id not found!" << endl;

}

}

break;

case 5:

if (members\_count == 0)

{

cout << "no members to display!" << endl;

}

else

{

cout << "information: " << endl;

for (int i = 0; i < members\_count; i++)

{

m[i].display\_member\_info();

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

}

}

break;

case 6:

if (trainers\_count == 0)

{

cout << "no trainers to display!" << endl;

}

else

{

cout << "information: " << endl;

for (int i = 0; i < trainers\_count; i++)

{

t[i].display\_trainer\_info();

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

}

}

break;

case 7:

if (members\_count == 0)

{

cout << "no members to display!" << endl;

}

else

{

int d\_m\_id;

cout << "enter member id to delete:";

cin >> d\_m\_id;

int d\_m\_found;

d\_m\_found = 0;

for (int i = 0; i < members\_count; i++)

{

if (m[i].get\_memberid() == d\_m\_id)

{

for (int j = i; j < members\_count - 1; j++)

{

m[j] = m[j + 1];

}

members\_count--;

d\_m\_found = 1;

cout << "deleted!" << endl;

break;

}

}

if (d\_m\_found == 0)

{

cout << "member id not found!" << endl;

}

}

break;

case 8:

if (trainers\_count == 0)

{

cout << "no trainers to display!" << endl;

}

else

{

int d\_t\_id;

cout << "enter trainer id to delete:";

cin >> d\_t\_id;

int d\_t\_found;

d\_t\_found = 0;

for (int i = 0; i < trainers\_count; i++)

{

if (t[i].get\_trainerid() == d\_t\_id)

{

for (int j = i; j < trainers\_count - 1; j++)

{

t[j] = t[j + 1];

}

trainers\_count--;

d\_t\_found = 1;

cout << "deleted!" << endl;

}

}

if (d\_t\_found == 0)

{

cout << "trainer id not found!" << endl;

}

}

break;

case 9:

{

if (members\_count == 0)

{

cout << "no members to update!" << endl;

}

else

{

int update\_id;

cout << "enter id to update gym membership: ";

cin >> update\_id;

int update\_found;

update\_found = 0;

for (int i = 0; i < members\_count; ++i)

{

if (m[i].get\_memberid() == update\_id)

{

cout << "updating fmembership for: " << update\_id << endl;

m[i].display\_peron\_info();

m[i].get\_membership\_info();

update\_found = 1;

cout << "information updated!" << endl;

break;

}

}

if (update\_found == 0)

{

cout << "member id not found! " << endl;

}

}

break;

}

case 10:

{

m[0].displaymemberdata();//(display file content)...(will display the deleted members too)....

break;

}

case 11:

{

cout << "EXITING...." << endl;

break;

}

default:

cout << "invalid choice!" << endl;

}

}

}

else if (choicee == 2)

{

int var;

int select\_m\_id;

cout << "enter membe id to login:";

cin >> select\_m\_id;

int mm\_found;

mm\_found = 0;

for (int i = 0; i < members\_count; i++)

{

if (m[i].get\_memberid() == select\_m\_id)

{

cout << endl;

cout << "member info:" << endl;

m[i].display\_peron\_info();

mm\_found = 1;

if (return\_gender(m[i]) == 1)//friend function

{

var = 1;

}

else

{

var = 2;

}

}

}

if (mm\_found == 0)

{

cout << "member with such id not found!" << endl;

}

else

{

cout << "welcome!\n";

while (choice != 5)

{

cout << endl;

cout << "1. Buying Gym Suppliments And Accesories" << endl;

cout << "2. Dietplan " << endl;

cout << "3. Workoutplan" << endl;

cout << "4. calculate weight difference (weight difference for one month)" << endl;

cout << "5. Exit" << endl;

cout << "Enter Your Choice: ";

cin >> choice;

switch (choice)

{

case 1:

{

{

//operator overloading()...

suppliments\_and\_accessories c1, c2, c3;

cout << "AVAILABLE SUPPLIMENTS" << endl;

c1.buy\_sup();

int buyAccessories;

cout << "\nDo you want to buy gym accessories?\n1.yes\n2.no " << endl;

cout << "your choice: ";

cin >> buyAccessories;

if (buyAccessories == 1)

{

cout << "\nAVAILABLE ACCESSORIES" << endl;

c2.buy\_acc();

}

c3 = c1 + c2;

c3.display\_total\_bill();

}

break;

}

case 2:

{

if (var == 1)//(PVF)...

{

DietPlan dp;

int v = 1;

dp.displayplan(v);

break;

}

else

{

DietPlan dp;

int v = 2;

dp.displayplan(v);

break;

}

}

case 3://(PVF)...

{

if (var == 1)

{

WorkoutPlan wp;

int v = 1;

wp.displayplan(v);

break;

}

else

{

WorkoutPlan wp;

int v = 2;

wp.displayplan(v);

break;

}

}

case 4:

{

int ipt;

cout << "1.integer weight values\n2.floating weight values" << endl;

cin >> ipt;

if (ipt == 1)

{

int currentweight, lastmonthweight;

cout << "enter current month weight:";

cin >> currentweight;

cout << "enter last month weight:";

cin >> lastmonthweight;

int difference;

difference = calculate\_weight\_difference(currentweight, lastmonthweight);

display\_result(difference);

}

else if (ipt == 2)

{

float currentweight, lastmonthweight;

cout << "enter current month weight:";

cin >> currentweight;

cout << "enter last month weight:";

cin >> lastmonthweight;

float difference;

difference = calculate\_weight\_difference(currentweight, lastmonthweight);

display\_result(difference);

}

else

{

cout << "invalid input!" << endl;

}

break;

}

case 5:

{

cout << "EXITING...." << endl;

break;

}

default:

cout << "invalid choice!" << endl;

}

}

}

}

else if (choicee == 3)

{

cout << "logging off..... " << endl;

}

else

{

cout << "invalid input!" << endl;

}

} while (choicee != 3);

\_getch();

}