2.5 Example to more than one class lecture seven

EX: c++ program to make registration at the parking

```
#include <iostream>
#include<string>
using namespace std;

class parking
{
        private:
        int valume;
        int now;
        int price;
        int time;
        public:
        int costing();
        int getnumnow();
        int getvalume();
        void incrementnow();
        void setpinfo(int,int,int);
        void settime(int);
};

class person
{
        private:
        string name;
        string number;
        int start;
        int end;
        public:
        void set(string,string,int);
        void setn(int );
   void gettime(int&,int&);
   void getinfo(string&,string&);
};
void register1(person&,parking&);
void print(person&,parking&);
int parking::costing()
{   int c;
    c=time*price;
        return c;
}
int parking::getnumnow()
{
        return now;
}

int parking::getvalume()
{
        return valume;
}
void parking::settime(int t)
{
        time=t;
}
void parking::incrementnow()
{
        now++;
}
void parking::setpinfo(int v,int n,int p)
{       valume=v;
        now=n;
        price=p;
}
void person::set(string st,string st2,int
i)
{
        name=st;
        number=st2;
        start=i;
}
void person::setn(int i)
{end=i;
}
void person::gettime(int& s ,int& e)
{       s=start;
```

```
        e=end;}
void person::getinfo(string& s,string&
n)
{   s=name;
        n=number;}
void print(person& PO,parking& PK)
{
        string n,num;
        int s,e,c;
        PO.getinfo(n,num);
        PO.gettime(s,e);
        PK.settime(e-s);
        c=PK.costing();
        cout<<n<<" car number is:
"<<num<<endl;
        cout<<"strt regestration at:
"<<s<<" and end at: "<<e<<endl;
        cout<<"Please pay this:
"<<c<<"$"<<endl;
        cout<<"*_*_*_*_*_*_*_*_*_*
_*_*_*_*"<<endl; }
void register1(person& PO,parking&
PK)
{   string n,s1;
        int st;
        if((PK.getvalume()-
PK.getnumnow())>0)
        {cout<<"please enter your
information:"<<endl;
        cin>>n>>s1;
        cin>>st;
        PO.set(n,s1,st);}
     else
            cout<<"sory the parking is
full";}
 int ending()
 {  int e;
   cout<<"time of leaving is: ";
        cin>>e;
        return e;
 }
 int main()
 {
 parking p1;
 p1.setpinfo(500,200,20);
 person A;
 register1(A,p1);
 A.setn(ending());
 print(A,p1);
 }
```

```
please enter your information:
ali AA1234
2
time of leaving is: 6
ali car number is: AA1234
strt regestration at: 2 and end at: 6
Please pay this: 80$
*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*_*
Press any key to continue . . .
```

EX: program showing the interaction between the seller and the juicer using two different class.

```
#include <iostream>
using namespace std;
class cashRegister
{
public:
int getCurrentBalance() ;

void acceptAmount(int amountIn);

cashRegister(int cashIn = 500);

private:
int cashOnHand;
};
class dispenserType
{
public:
int getNoOfItems();
int getCost() const;
void makeSale();
dispenserType(int setNoOfItems = 50,
int setCost = 50);
private:
int numberOfItems;
int cost;
};
using namespace std;
int cashRegister::getCurrentBalance()
{
return cashOnHand;
}
void cashRegister::acceptAmount(int
amountIn)
{
cashOnHand = cashOnHand +
amountIn;
}
cashRegister::cashRegister(int cashIn)
{
if (cashIn >= 0)
cashOnHand = cashIn;
else
cashOnHand = 500;
}
int dispenserType::getNoOfItems()
{
return numberOfItems;
}
int dispenserType::getCost()
{
return cost;
}
void dispenserType::makeSale()
{
numberOfItems--;
}
dispenserType::dispenserType(int
setNoOfItems, int setCost)
{
if (setNoOfItems >= 0)
numberOfItems = setNoOfItems;
else
```

```
numberOfItems = 50;
if (setCost >= 0)
cost = setCost;
else
cost = 50;
}
void showSelection();
void sellProduct(dispenserType&
product,cashRegister& pCounter);
int main()
{
cashRegister counter;
dispenserType orange(100, 50);
dispenserType apple(100, 65);
dispenserType mango(75, 80);
dispenserType strawberrybanana(100,
85);
int choice;
showSelection();
cin >> choice;
while (choice != 9)
{
switch (choice)
{
case 1:
sellProduct(orange, counter);
break;
case 2:
sellProduct(apple, counter);
break;
case 3:
sellProduct(mango, counter);
break;
case 4:
sellProduct(strawberrybanana,
counter);
break;
default:
cout << "Invalid selection." << endl;
}//end switch
showSelection();
cin >> choice;
}//end while
return 0;
}//end main
void showSelection()
{
cout << "*** Welcome to Shelly's
Juice Shop ***" << endl;
cout << "To select an item, enter " <<
endl;
cout << "1 for orange juice" << endl;
cout << "2 for apple juice" << endl;
cout << "3 for mango juice" << endl;
cout << "4 for strawberry banana" <<
endl;
cout << "9 to exit" << endl;
}//end showSelection
```

```
void sellProduct(dispenserType& product,cashRegister& pCounter)
{
int amount;
int amount2;
if (product.getNoOfItems() > 0)
{
cout << "Please deposit " << product.getCost()
<< " cents" << endl;
cin >> amount;
if (amount < product.getCost())
{
cout << "Please deposit another "
<< product.getCost()- amount
<< " cents" << endl;
cin >> amount2;
amount = amount + amount2;
}
if (amount >= product.getCost())
{
pCounter.acceptAmount(amount);
product.makeSale();
cout << "Collect your item at the bottom and "
<< "enjoy." << endl;
}
else
cout << "The amount is not enough. "
<< "Collect what you deposited." << endl;
cout << "*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*"
<< endl << endl;
}
else
cout << "Sorry, this item is sold out."
<< endl;
}//end sellProduct
```

```
*** Welcome to Shelly's Juice Shop ***
To select an item, enter
1 for orange juice
2 for apple juice
3 for mango juice
4 for strawberry banana
9 to exit
1
Please deposit 50 cents
60
Collect your item at the bottom and enjoy.
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

*** Welcome to Shelly's Juice Shop ***
To select an item, enter
1 for orange juice
2 for apple juice
3 for mango juice
4 for strawberry banana
9 to exit
```

2.6 Composition (Aggregation)

Composition (aggregation) is another way to relate two classes. In composition (aggregation), one or more members of a class are objects of another class type. Composition is a ‘‘has-a’’ relation; for example, ‘‘every person has a date of birth.

EX: program showing the personal information using class, it used two other class in main class(class to represent name of person and class to represent the birthdate.

```
#include <iostream>
#include<string>
using namespace std;
class personType
{private:
string firstName; //variable to store the first name
string lastName; //variable to store the last name

public:
void print();
void setName(string, string );
string getFirstName();
string getLastName();
personType(string first = "", string last = "");
};
void personType::print()
{
cout << firstName << " " << lastName;
}
void personType::setName(string first, string last)
{
firstName = first;
lastName = last;
}
string personType::getFirstName()
{
return firstName;
}
string personType::getLastName()
{
return lastName;
}
//constructor
personType::personType(string first, string last)
{
firstName = first;
lastName = last;
}
class dateType
{
private:
int dMonth; //variable to store the month
int dDay; //variable to store the day
int dYear; //variable to store the year
public:
void setDate(int, int, int );
int getDay() ;
int getMonth() ;
int getYear();
void printDate();
dateType(int month = 1, int day = 1, int year = 1900);
```

```
};
void dateType::setDate(int month, int
day, int year)
{
dMonth = month;
dDay = day;
dYear = year;
}
int dateType::getDay()
{
return dDay;
}
int dateType::getMonth()
{
return dMonth;
}
int dateType::getYear()
{
return dYear;
}
void dateType::printDate()
{
cout << dMonth << "-" << dDay << "-
" << dYear;
}
//Constructor with parameters
dateType::dateType(int month, int
day, int year)
{
dMonth = month;
dDay = day;
dYear = year;
}
class personalInfo
{
private:
personType name;
dateType bDay;
int personID;
public:
void setpersonalInfo(string , string ,
int,int , int , int );
void printpersonalInfo ();
//Function to print the personal
information.
personalInfo(string first = "", string
last = "",int month = 1, int day = 1, int
year = 1900,int ID = 0);

};
void
personalInfo::setpersonalInfo(string
first, string last,
int month, int day, int year, int ID)
{
name.setName(first,last);
bDay.setDate(month,day,year);
personID = ID;
}
void personalInfo::printpersonalInfo()
{
name.print();
cout << "'s date of birth is ";
bDay.printDate();
cout << endl;
cout << "and personal ID is " <<
personID;
}
personalInfo::personalInfo(string first,
string last, int month,
int day, int year, int ID): name(first,
last), bDay(month, day, year)
{
personID = ID;
}
int main()
{
        personalInfo
p("ziad","omar",12,7,1999,10);
        p.printpersonalInfo();}
```

2.6 Composition (Aggregation)

Composition (aggregation) is another way to relate two classes. In composition (aggregation), one or more members of a class are objects of another class type. Composition is a ‘‘has-a’’ relation; for example, ‘‘every person has a date of birth.

EX: program showing the personal information using class, it used two other class in main class(class to represent name of person and class to represent the birthdate.

```
#include <iostream>
#include<string>
using namespace std;
class personType
{private:
string firstName; //variable to store the first name
string lastName; //variable to store the last name

public:
void print();
void setName(string, string );
string getFirstName();
string getLastName();
personType(string first = "", string last = "");
};
void personType::print()
{
cout << firstName << " " << lastName;
}
void personType::setName(string first, string last)
{
firstName = first;
lastName = last;
}
string personType::getFirstName()
{
return firstName;
}
string personType::getLastName()
{
return lastName;
}
//constructor
personType::personType(string first, string last)
{
firstName = first;
lastName = last;
}
class dateType
{
private:
int dMonth; //variable to store the month
int dDay; //variable to store the day
int dYear; //variable to store the year
public:
void setDate(int, int, int );
int getDay() ;
int getMonth() ;
int getYear();
void printDate();
dateType(int month = 1, int day = 1, int year = 1900);
};
```

```
void dateType::setDate(int month, int
day, int year)
{
dMonth = month;
dDay = day;
dYear = year;
}
int dateType::getDay()
{
return dDay;
}
int dateType::getMonth()
{
return dMonth;
}
int dateType::getYear()
{
return dYear;
}
void dateType::printDate()
{
cout << dMonth << "-" << dDay << "-
" << dYear;
}
//Constructor with parameters
dateType::dateType(int month, int
day, int year)
{
dMonth = month;
dDay = day;
dYear = year;
}
class personalInfo
{
private:
personType name;
dateType bDay;
int personID;
public:
void setpersonalInfo(string , string ,
int,int , int , int );
void printpersonalInfo ();
//Function to print the personal
information.
personalInfo(string first = "", string
last = "",int month = 1, int day = 1, int
year = 1900,int ID = 0);

};
void
personalInfo::setpersonalInfo(string
first, string last,
int month, int day, int year, int ID)
{
name.setName(first,last);
bDay.setDate(month,day,year);
personID = ID;
}
void personalInfo::printpersonalInfo()
{
name.print();
cout << "'s date of birth is ";
bDay.printDate();
cout << endl;
cout << "and personal ID is " <<
personID;
}
personalInfo::personalInfo(string first,
string last, int month,
int day, int year, int ID): name(first,
last), bDay(month, day, year)
{
personID = ID;
}
int main()
{
        personalInfo
p("ziad","omar",12,7,1999,10);
        p.printpersonalInfo()
```