Assignment 3(Inheritence)

Muhammad Sharjeel Husnain

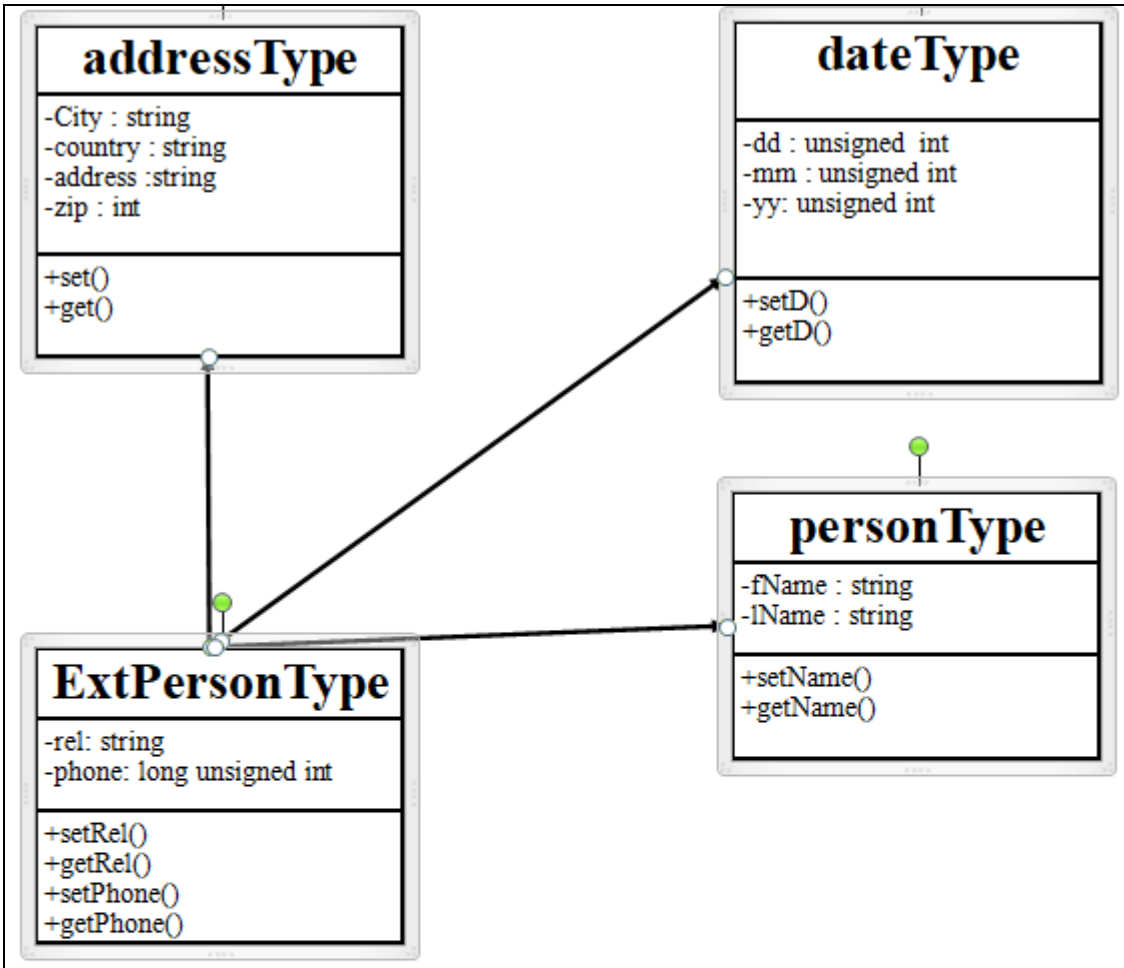
12/12/2022

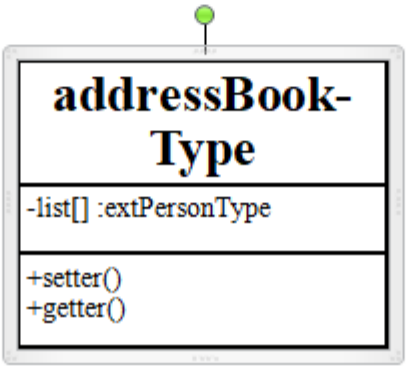
—

OOP

—

Prof. Muhammad Nadeem


addressType
-City : string
-country : string
-address :string
-zip : int
+set()
+get()
dateType
-dd : unsigned int
-mm : unsigned int
-yy: unsigned int
+setD()
+getD()
personType
-fName : string
-lName : string
+setName()
+getName()
ExtPersonType
-rel: string
-phone: long unsigned int
+setRel()
+getRel()
+setPhone()
+getPhone()


addressBook-
Type
-list[] :extPersonType
+setter()
+getter()

/*

```
* Muhammad Sharjeel
* 4345
* compiler used Microsoft VS (Community)
question 6: Using classes, design an online address book to keep track of the names,
addresses, phone numbers, and dates of birth of family members, close
friends, and certain business associates. Your program should be able to
handle a maximum of 500 entries.
a. Define a class, addressType, that can store a street address, city,
state, and ZIP code. Use the appropriate functions to print and store the
address. Also, use constructors to automatically initialize the member
variables.
b. Define a class extPersonType using the class personType (as
defined in Example 12-9, Chapter 12), the class dateType (as designed
in this chapter's Programming Exercise 2), and the class addressType.
Add a member variable to this class to classify the person as a family
788 | Chapter 13: Inheritance and Composition
member, friend, or business associate. Also, add a member variable to store
the phone number. Add (or override) the functions to print and store the
appropriate information. Use constructors to automatically initialize the
member variables.
c. Define the class addressBookType using the previously defined
classes. An object of the type addressBookType should be able to
process a maximum of 500 entries.
The program should perform the following operations:
i. Load the data into the address book from a disk.
ii. Sort the address book by last name.
iii. Search for a person by last name.
iv. Print the address, phone number, and date of birth (if it exists) of a
given person.
v. Print the names of the people whose birthdays are in a given month.
vi. Print the names of all of the people between two last names.
vii. Depending on the user's request, print the names of all family
members, friends, or business associates*/#include <iostream>
#include <string>
using namespace std;
#define SIZE 500


class addressType {

	string city, country, address;
	int zip;
public:
	addressType() {
		city = "\0";
		country = "\0";
		address = "\0";
		zip = 0;
	}
	void set();
	void get();
};

void addressType::set() {
	cout << "Enter address e.g street#, house#:" << endl;
	cin.ignore();
	getline(cin, address);

	cout << "Enter city name:" << endl;
	cin.ignore();
		getline(cin, city);
```

```
	cout << "Enter country name:" << endl;
	cin.ignore();
	getline(cin, country);
	cout << "Enter Zip code:" << endl;
	cin >> zip;
}
void addressType::get() {
	cout << "Address:" << address << endl;
	cout << "City:" << city << endl;
	cout << "State:" << country << endl;
	cout << "Zip Code:" << zip << endl;
}

class dateType {
protected:
	unsigned int dd, mm, yy;
public:
	dateType() {
		dd = 0;
		mm = 0;
		yy = 0;

	}
	void setD();
	void getD();
};
void dateType::setD()
{
	cout << "Enter Date Of birth dd/mm/yyyy" << endl;
	cin >> dd;
	cout << "/";
	cin >> mm;
	cout << "/";
	cin >> yy;
	cout << endl;

}
void dateType::getD()
{
	cout << "Date of birth is: " << dd << "/" << mm << "/" << yy << endl;
}

class personType {

	string fName, lName;
public:
	personType() {
		fName = lName = "\0";


	}
	void setName();
	void getName();
};

void personType::setName() {
	cout << "Enter First Name:" << endl;
	cin.ignore();
	getline(cin, fName);
	cout << "Enter your Last Name:" << endl;
```

```
	cin.ignore(0);
	getline(cin, lName);

}
void personType::getName() {
	cout << "first name :" << fName << endl;
	cout << "last name:" << lName << endl;
}

class extPersonType :virtual public addressType, virtual public personType, virtual public
dateType
{
	string rel;
	long unsigned phone;
public:
	extPersonType()
	{
		rel = "\0";
		phone = 0;
	}
	void setRel();
	void getRel();
	void setPhone();
	void getPhone();
};
void extPersonType::setRel()
{
	cout << "What is Relation? a family member, friend, or business friend ? Enter Below" <<
endl;
	char choice;
	cout << "1. Family\n"
		<< "2. Friend\n"
		<< "3. Business Friend " << endl;
	cin >> choice;
	switch (choice)
	{
	case '1':
		rel = "Family";
		break;
	case '2':
		rel = "Friend";
		break;
	case '3':
			rel = "Business Friend";
			break;

	}
}
void extPersonType::getRel() {
	cout << "Relation is " << rel << endl;
}
void extPersonType::setPhone() {
	cout << "Enter phone No" << endl;
	cin >> phone;
}
void extPersonType::getPhone() {
	cout << "Phone Number : " << phone << endl;
}

class addressBookType {
	extPersonType list[SIZE];
```

```
public:
	void setter(int);
	void getter(int);
};
void addressBookType::setter(int n)
{
	for (int i = 0; i < n; i++)
	{
		list[i].setName();
		list[i].set();
		list[i].setD();
		list[i].setRel();
		list[i].setPhone();
		system("pause");
		system("cls");

	}
}
void addressBookType::getter(int n)
{
	for (int i = 0; i < n; i++)
	{
		list[i].getName();
		list[i].get();
		list[i].getD();
		list[i].getRel();
		list[i].getPhone();

		cout<<"\n\n";
	}
}
int main()
{
	addressBookType p;
	int n = 0;
	char choice=0;
	do {
		cout << "***Main Menu***\n"
			<< "Press 1: Create Address Book\n"
			<< "Press 2: Display Address Book\n"
			<< "Press 9: Exit" << endl;
		cin >> choice;
		switch (choice)
		{
		case '1':
			cout << "how many address you want to store?" << endl;
			cin >> n;
			p.setter(n);

			break;

		case '2':
			p.getter(n);
			break;
		}
		system("pause");
		system("cls");
	} while (true);
	return 0;
}
```

C:\Users\Shoaib\source\repos\ch 13, Question 6, DS Malik\x64\Debug\ch 13, Que

```
***Main Menu***
Press 1: Create Address Book
Press 2: Display Address Book
Press 9: Exit
```

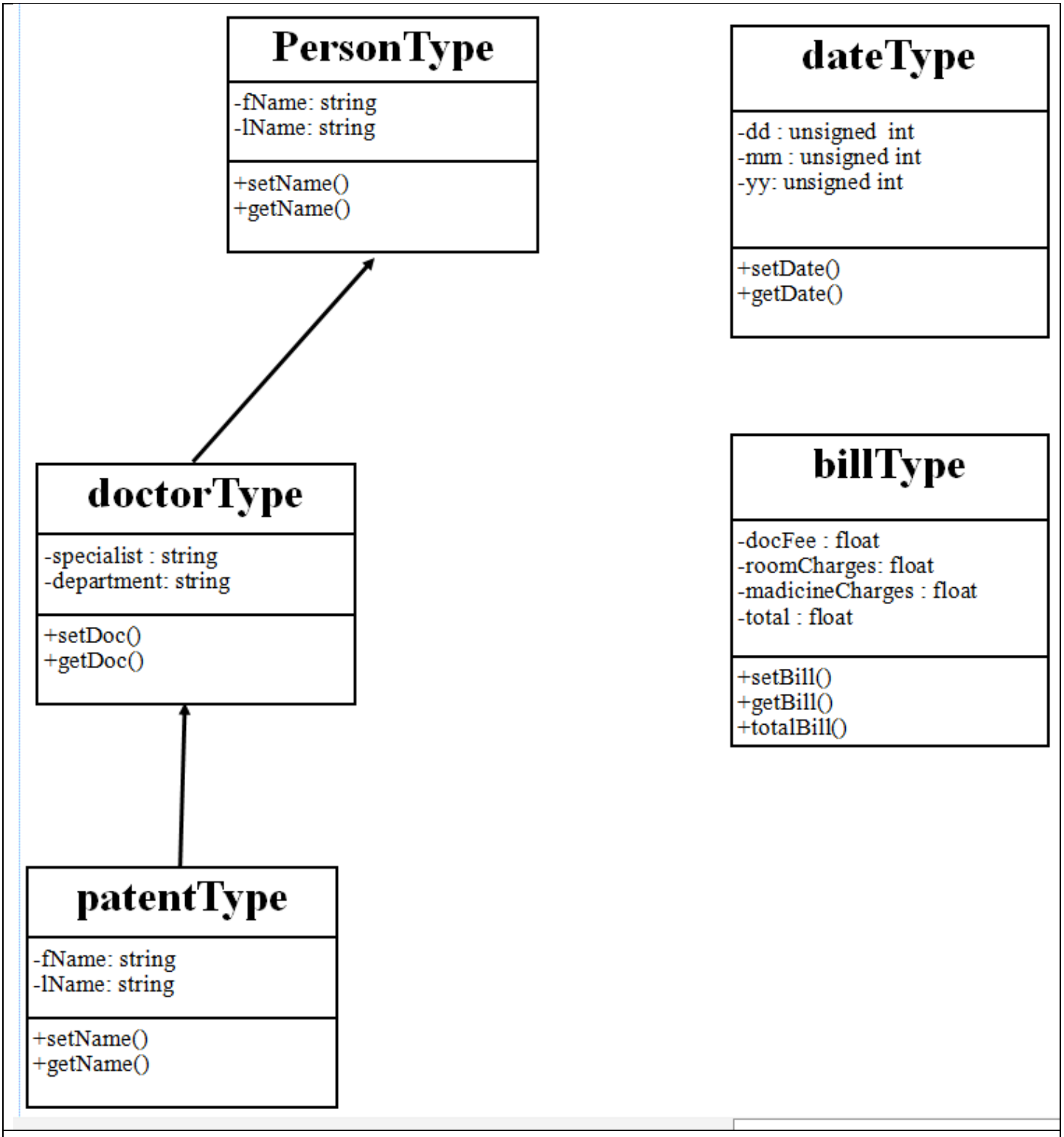
C:\Users\Shoaib\source\repos\ch 13, Question 6, DS Malik\x64\Debug\ch 13, Question 6, DS Malik.exe

```
Press 1: Create Address Book
Press 2: Display Address Book
Press 9: Exit
1
how many address you want to store?
1
Enter First Name:
 Sharjeel
Enter your Last Name:
 Husnain
Enter address e.g street#, house#:
 4
Enter city name:
 wah
Enter country name:
 Pakistan
Enter Zip code:
 47000
Enter Date Of birth dd/mm/yyyy
12
/12
/2001

What is Relation? a family member, friend, or business friend ? Enter Below
1. Family
2. Friend
3. Business Friend
1
Enter phone No
 03335577435
```

```
C:\Users\Shoaib\source\repos\ch 13, Question 6, DS Malik\x64\Debug\ch 13, Question 6, DS Malik.exe
***Main Menu***
Press 1: Create Address Book
Press 2: Display Address Book
Press 9: Exit
2
first name : Sharjeel
last name: Husnain
Address:4
City:wah
State:Pakistan
Zip Code:47000
Date of birth is: 12/12/2001
Relation is Family
Phone Number : 3335577435


Press any key to continue . . .
```




```
/*Question 12:
In this exercise, you will design various classes and write a program to
computerize the billing system of a hospital.
a. Design the class doctorType, inherited from the class
personType, defined in Chapter 12, with an additional data member
```

```
to store a doctor’s speciality. Add appropriate constructors and mem_ber functions to
initialize, access, and manipulate the data members.
b. Design the class billType with data members to store a patient’s ID
and a patient’s hospital charges, such as pharmacy charges for medicine,
doctor’s fee, and room charges. Add appropriate constructors and
member functions to initialize and access and manipulate the data
members.
c. Design the class patientType, inherited from the class
personType, defined in Chapter 12, with additional data members
to store a patient’s ID, age, date of birth, attending physician’s name,
the date when the patient was admitted in the hospital, and the date
when the patient was discharged from the hospital. (Use the class
dateType to store the date of birth, admit date, discharge date, and
the class doctorType to store the attending physician’s name.)
Add appropriate constructors and member functions to initialize,
access, and manipulate the data members.
Muhammad Sharjeel Husnain
4345
compiiler used MS Visual Studio (Community Eddition)

*/
#include <iostream>
#include <string>
using namespace std;
class personType {
	string fName, lName;
public:
	personType(string a = "\0", string b = "\0") :fName(a), lName(b) {}
	void setName();
	void getName();
};
void personType::setName()
{
	cout << "Enter First Name" << endl;
	cin.ignore(3, '\n');
	getline(cin, fName);
	cout << "Enter Last Name" << endl;
	getline(cin, lName);
}
void personType::getName()
{
	cout << "first name : " << fName << endl;

		cout << "Last name : " << lName << endl;
}
class doctorType : public personType {

	string specialist, department;
public:
	doctorType() {

		specialist = "\0";
		department = "\0";
	}
	void setDoc();
	void getDoc();
};
void doctorType::setDoc() {
	cout << "Enter Doctor's Specialism:" << endl;
	cin.ignore();
	getline(cin, specialist);
```

```
	cout << "Enter Doctor's Department:" << endl;

	getline(cin, department);
}
void doctorType::getDoc() {
	cout << "Doctor's Specialism:" << specialist << endl;
	cout << "Doctor's Department:" << department << endl;
}
class dateType {

	unsigned int d1, m1, y1;

public:
	dateType() {
		d1  = m1 = y1 = 0;
	}
	void setDate();
	void getDate();
};
void dateType::setDate() {
	cout << "Enter Date dd/mm/yy:" << endl;
	cin >> d1 >> m1 >> y1;

}
void dateType::getDate() {
	cout << d1 << "/" << m1 << "/" << y1 << endl;

}
class billType {
	float docFee, roomCharges, madicineCharges, total;
public:
	billType() {
		docFee = roomCharges = madicineCharges = total = 0.0;
	}
	void setBill();
	void getBill();
	float totalBill();
};
void billType::setBill()
{
	cout << "Enter Doctor Fees " << endl;
	cin >> docFee;
	cout << "Enter Room Charges " << endl;
	cin >> roomCharges;
	cout << "Enter Madicine Charges:" << endl;
	cin >> madicineCharges;

}
float billType::totalBill()
{
	total = docFee + roomCharges + madicineCharges;
	return total;
}
void billType::getBill()
{
	cout << "Doctor Fees is: " << docFee << endl;
	cout << "Room Charges are " << roomCharges << endl;
	cout << "Madicine Charges are :" << madicineCharges << endl;
	cout << "Total Bill is: " << totalBill() << endl;
}
class patentType :public doctorType
```

```
{
	string pId;
	unsigned int age;
public:
	patentType()
	{
		pId = "\0";
		age = 0;

	}
	void getPatent();
	void setPatent();
};
void patentType::setPatent() {
	cout << "Enter Patent ID" << endl;
	cin.ignore();
	getline(cin, pId);
	cout << "Enter Patent Age" << endl;
	cin >> age;
}
void patentType::getPatent() {
	cout << "Patent ID is: " << pId << endl;
	cout << "Patent age is :" << age << endl;
}
int main()
{
	dateType date;
	patentType person;
	billType bill;
	char choice;
	do {
		cout << "Main Menu" << endl;
		cout << "Press 1: Doctor" << endl;
		cout << "Press 2: Patient" << endl;
		cout << "Press 3: Exit" << endl;
		cout << "Select Option:" << endl;
		cin >> choice;
		switch (choice)
		{
		case '1':
			cout << "Press 1: Set data." << endl;
			cout << "Press 2: Get data." << endl;
			cin >> choice;
			switch (choice)
			{
			case '1':
				person.setName();
				person.setDoc();
				break;
			case '2':
				person.getName();
				person.getDoc();
				break;
			}
			break;
		case '2':
			cout << "Press 1: Set data." << endl;
			cout << "Press 2: Get data." << endl;
			cin >> choice;
			switch (choice)
			{
```

```
                case '1':
                        person.setName();


                                person.setPatent();
                        cout << "Enter Patent's Doctor Details " << endl;
                        person.setName();
                        person.setDoc();
                        cout << "Enter date of birth" << endl;
                        date.setDate();
                                cout << "Enter date of Admission" << endl;
                        date.setDate();
                        cout << "Enter date of discharge" << endl;
                        date.setDate();
                        bill.setBill();
                        break;
                case '2':
                        person.getName();
                        person.getPatent();
                        cout << "Patent's Doctor Details " << endl;
                        person.getName();
                        person.getDoc();
                        cout << " date of birth is" << endl;
                        date.getDate();
                        cout << " date of Admission is " << endl;
                        date.getDate();
                        cout << " date of discharge is" << endl;
                        date.getDate();
                        bill.getBill();

                        break;
                }
                break;
        case '3':
                exit(0);
        }
        system("pause");
        system("cls");
    } while (true);
    return 0;
}
```

C:\Users\SHOAIB\source\repos\q 12\x64\Debug\q 12.exe

```
Main Menu
Press 1: Doctor
Press 2: Patient
Press 3: Exit
Select Option:
```

C:\Users\SHOAIB\source\repos\q 12\x64\Debug\q 12.exe

```
Main Menu
Press 1: Doctor
Press 2: Patient
Press 3: Exit
Select Option:
1
Press 1: Set data.
Press 2: Get data.
1
Enter First Name
 Sharjeel
Enter Last Name
Husnain
```

```
C:\Users\SHOAIB\source\repos\q 12\x64\Debug\q 12.exe
Main Menu
Press 1: Doctor
Press 2: Patient
Press 3: Exit
Select Option:
1
Press 1: Set data.
Press 2: Get data.
1
Enter First Name
 Sharjeel
Enter Last Name
Husnain
Enter Doctor's Specialism:
 Medical
Enter Doctor's Department:
 Hospital
Press any key to continue . . .
```

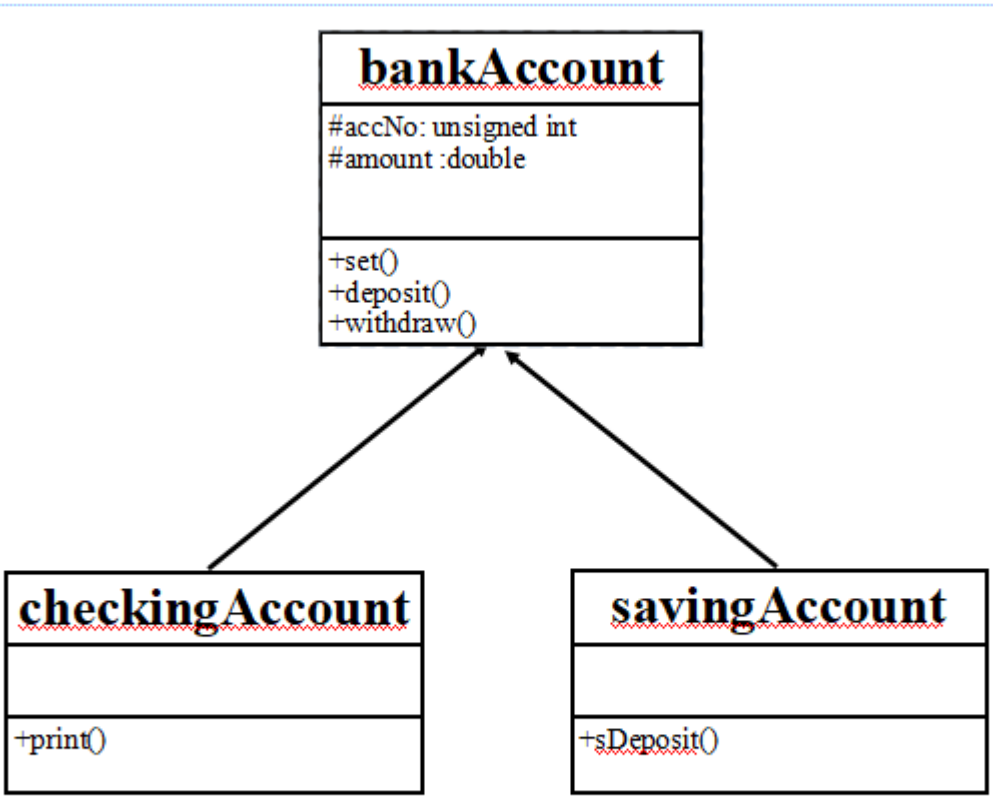
```
5            break;
```

```
C:\Users\SHOAIB\source\repos\q 12\x64\Debug\q 12.exe
Main Menu
Press 1: Doctor
Press 2: Patient
Press 3: Exit
Select Option:
1
Press 1: Set data.
Press 2: Get data.
2
first name :  Sharjeel
Last  name : Husnain
Doctor's Specialism:Medical
Doctor's Department: Hospital
Press any key to continue . . .
```

```
C:\Users\SHOAIB\source\repos\q 12\x64\Debug\q 12.exe
Press 1: Set data.
Press 2: Get data.
1
Enter First Name
 Sharjeel
Enter Last Name
Husnain
Enter Patent ID
 45
Enter Patent Age
22
Enter Patent's Doctor Details
Enter First Name
 Musab
Enter Last Name
 Akram
Enter Doctor's Specialism:
 Medical
Enter Doctor's Department:
 Hosp
Enter DoB:
Date:
11
Month:
12
Year:
2001
Enter Date of Admition:
Date:
```

```
C:\Users\SHOAIB\source\repos\q 12\x64\Debug\q 12.exe
Enter Doctor's Department:
 Hosp
Enter DoB:
Date:
11
Month:
12
Year:
2001
Enter Date of Admition:
Date:
12
Month:
11
Year:
2020
Enter Discharge Date:
Date:
12
Month:
4
Year:
2021
Enter Doctor Fees
500
Enter Room Charges
2000
Enter Madicine Charges:
1000
Press any key to continue . . .
```

```
C:\Users\SHOAIB\source\repos\q 12\x64\Debug\q 12.exe
Main Menu
Press 1: Doctor
Press 2: Patient
Press 3: Exit
Select Option:
2
Press 1: Set data.
Press 2: Get data.
2
first name :  Musab
Last  name :  Akram
Patent ID is: 45
Patent age is :22
Patent's Doctor Details
first name :  Musab
Last  name :  Akram
Doctor's Specialism:Medical
Doctor's Department: Hosp
date of Birth     is: 11/12/2001
date of Admition  is: 12/11/2020
date of Discharge is: 12/4/2021
Doctor Fees is: 500
Room Charges are 2000
Madicine Charges are :1000
Total Bill is: 3500
Press any key to continue . . .
```




```
/*Question 14
```

```
Define the class bankAccount to store a bank customer's account
number and balance. Suppose that account number is of type int, and
balance is of type double. Your class should, at least, provide the
following operations: set the account number, retrieve the account
number, retrieve the balance, deposit and withdraw money, and print
account information. Add appropriate constructors.
1
3
Programming Exercises | 791
b. Every bank offers a checking account. Derive the class
checkingAccount from the class bankAccount (designed in
part (a)). This class inherits members to store the account number
and the balance from the base class. A customer with a checking
account typically receives interest, maintains a minimum balance,
and pays service charges if the balance falls below the minimum
balance. Add member variables to store this additional information.
In addition to the operations inherited from the base class, this class
should provide the following operations: set interest rate, retrieve
interest rate, set minimum balance, retrieve minimum balance, set
service charges, retrieve service charges, post interest, verify if the
balance is less than the minimum balance, write a check, withdraw
(override the method of the base class), and print account informa_tion. Add appropriate
constructors.
c. Every bank offers a savings account. Derive the class
savingsAccount from the class bankAccount (designed in part
(a)). This class inherits members to store the account number and
the balance from the base class. A customer with a savings account
typically receives interest, makes deposits, and withdraws money. In
addition to the operations inherited from the base class, this class
should provide the following operations: set interest rate, retrieve
interest rate, post interest, withdraw (override the method of the
base class), and print account information. Add appropriate con_structors.
Muhammad Sharjeel Husnain
4345
compiler used MS Visual Studio (Community Eddition)
*/
#include <iostream>
#include <string>
using namespace std;
class bankAccount {
protected:
	unsigned int accNo;
	double amount;
public:
	bankAccount() {
		accNo = 0;
		amount = 0.0;
	}
	void set();
	void deposit();
	void withdraw();
};
void bankAccount::set()
{
	cout << "enter account number" << endl;
	cin >> accNo;
}
void bankAccount::deposit()
{
	double fund;
	cout << "enter amount to deposit" << endl;
```

```
	cin >> fund;
	amount = amount + fund;
	cout << "Deposited" << endl;
}
void bankAccount::withdraw()
{
	double fund;
	cout << "enter amount to withdraw" << endl;
	cin >> fund;
	amount = amount - fund;
	cout << "withdraw successful!" << endl;
}
class checkingAccount :public virtual bankAccount {
public:
	checkingAccount() {
		accNo = 0;
		amount = 0.0;
	}
	void printAcc();
};


void checkingAccount::printAcc()
{
	cout << "Account Number is :" << accNo << endl;
	cout << "Account Balance is :" << amount << endl;
}
class savingAccount :public virtual bankAccount {
public:
	savingAccount() {
		accNo = 0;
		amount = 0.0;
	}
	void sDeposit();
};
void savingAccount::sDeposit() {
	double sFund;
	cout << "enter ammount to deposit" << endl;
	cin >> sFund;
	double intrest;
	intrest = (2 * sFund) / 100;
	amount = amount + sFund + intrest;
	cout << "2% Profit added " << endl;
}
int main() {
	checkingAccount p;
	savingAccount s;
	char choice;
	do {
		cout << "Press 1: Current Account" << endl;
		cout << "Press 2: Saving Account" << endl;
		cout << " Press 3: Exit" << endl;
		cin >> choice;
		switch (choice)
		{
		case '1':
			cout << "Press 1: Open Account" << endl;
			cout << "Press 2: Deposit" << endl;
			cout << "Press 3: Withdraw " << endl;
			cout << "Press 4: check Account " << endl;
			cin >> choice;
```

```
                switch (choice)
                {
                case '1':
                        p.set();
                        p.deposit();
                        break;
                case '2':
                        p.deposit();
                        break;
                case '3':
                        p.withdraw();
                        break;
                case '4':
                        p.printAcc();
                        break;
                }
                break;

        case '2':
                cout << "Press 1: Open Account" << endl;
                cout << "Press 2: Deposit" << endl;
                cout << "Press 3: Withdraw " << endl;
                cout << "Press 4: check Account " << endl;
                cin >> choice;
                switch (choice)
                {
                case '1':
                        p.set();
                        s.sDeposit();
                        break;
                case '2':
                        s.sDeposit();
                        break;
                case '3':
                        p.withdraw();
                        break;
                case '4':
                        p.printAcc();
                        break;
                }
                break;
        case '3':
                exit(0);
        }
        system("pause");
        system("cls");
    } while (true);
    return 0;
}
```

```
C:\Users\SHOAIB\source\repos\Q 14\x64\Debug\Q 14.exe
Stack Frame (Ctrl+7)
Press 1: Current Account
Press 2: Saving Account
 Press 3: Exit
```

```
C:\Users\SHOAIB\source\repos\Q 14\x64\Debug\Q 14.exe
Stack Frame (Ctrl+7)
Press 1: Current Account
Press 2: Saving Account
 Press 3: Exit
1
Press 1: Open Account
Press 2: Deposit
Press 3: Withdraw
Press 4: check Account
```

```
C:\Users\SHOAIB\source\repos\Q 14\x64\Debug\Q 14.exe
Press 1: Current Account
Press 2: Saving Account
 Press 3: Exit
2
Press 1: Open Account
Press 2: Deposit
Press 3: Withdraw
Press 4: check Account
2
enter ammount to deposit
100
2% Profit added
Press any key to continue . . .
```

```
C:\Users\SHOAIB\source\repos\Q 14\x64\Debug\Q 14.exe
Press 1: Current Account
Press 2: Saving Account
 Press 3: Exit
2
Press 1: Open Account
Press 2: Deposit
Press 3: Withdraw
Press 4: check Account
3
enter amount to withdraw
100
withdraw successful!
Press any key to continue . . .
```

```
C:\Users\SHOAIB\source\repos\Q 14\x64\Debug\Q 14.exe
Press 1: Current Account
Press 2: Saving Account
 Press 3: Exit
2
Press 1: Open Account
Press 2: Deposit
Press 3: Withdraw
Press 4: check Account
4
Account Number is :45
Account Balance is :0
Press any key to continue . . .
```

```
C:\Users\SHOAIB\source\repos\Q 14\x64\Debug\Q 14.exe
Press 1: Current Account
Press 2: Saving Account
 Press 3: Exit
1
Press 1: Open Account
Press 2: Deposit
Press 3: Withdraw
Press 4: check Account
1
enter account number
45
enter amount to deposit
100
```