SEHH2042 Computer Programming

Tutorial 7 – Object Oriented Programming

Q1. Given the code below (available on Moodle), fill in the missing parts so that the program is compilable and executable. Use **Date** as the name of the class, and declare three integer-typed data members for the class: **day, month, year**. What is the output of the program?

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


_______________
{
public:
      void setDate(int d, int m, int y)
      {
            day = d;
            month = m;
            year = y;
      }

      void print()
      {
            cout << day << "-" << month << "-" << year;
      }


____________
      // Declare private data members
      _________________
      _________________
      _________________
};

int main(){
      // Create an object called xmas
      ___________________
      xmas.setDate(25, 12, 2013);
      cout << "xmas is: ";
      // Use xmas object to call print()
      ___________________
      cout << endl;
      return 0;
}
```

Continue from the above program, do the following updates:

(a) **Add a constructor function** in the **Date** class. The constructor accepts three integers **d**, **m**, and **y** as arguments, which will be the values of **day**, **month**, and **year** of the object that is created.

(b) Note the **syntax error** in the object creation statement in the main function. Explain why such error occurs. (*If you don't have syntax error, you should have added extra constructor in (i), remove it.*)

(c) **Modify** the codes in the main function, so that the date is set at the time when the object **xmas** is created, instead of using **setDate** afterwards in main.

Q2. Consider the partially-completed **Car** class below (available on Moodle). Complete the program as follows:

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

class Car
{
```

```
public:
    Car() {
        // Your code for part (a) should be inserted here
    }

    void showSpeed() {
        cout << "The car is moving at " << speed << " km/h." << endl;
    }

    void accelerate(int a) {
        cout << "Accelerating ... " << endl;
        // Your code for part (b) should be inserted here
    }

    // Your code for part (c) should be inserted here

    // Your code for part (d) should be inserted here

private:
    int speed;   // speed of the car, in km/h
};

int main(){
    Car myCar;

    myCar.showSpeed();
    myCar.accelerate(70);
    myCar.showSpeed();
    myCar.decelerate(20);
    myCar.showSpeed();
    myCar.accelerate(120);
    myCar.showSpeed();
    myCar.decelerate(100);
    myCar.showSpeed();
    myCar.stop();
    myCar.showSpeed();

    return 0;
}
```

(a) In the constructor, initialize the **speed** of the car to 0.
(b) In the member function **accelerate()**, update the speed of the car by increasing it by **a** (parameter of function, integer type). Note that the speed of the car should not exceed 150 km/h.
(c) Similar to **accelerate()**, implement a public member function **decelerate()**, which decreases the speed of the car. Note that the speed of the car should not be negative.
(d) Implement a public member function **stop()**, which sets the speed of the car to 0.

Sample result:

```
The car is moving at 0 km/h.
Accelerating ...
The car is moving at 70 km/h.
Decelerating ...
The car is moving at 50 km/h.
Accelerating ...
The car is moving at 150 km/h.
Decelerating ...
The car is moving at 50 km/h.
Stopping ...
The car is moving at 0 km/h.
```

Q3. Create a class called **Phone** that includes three pieces of information as data members—the brand name (type string), the os type (type string), and the price (type double). The **constructor** has two parameters that uses the parameters to initialize the brand name and the os type. Provide a *set* function for the **price**, and ensure that the price value is non-negative. Also, provide a member function **showConfig** that displays the configuration and price of the phone. To demonstrate the requirements, the sample results will be produced if the following main function is executed.

Note: You need to include header file <string> for displaying value of string variables.

```
int main(){
      Phone iPhone("Apple", "iOS version 6");
      Phone noteTwo("Samsung", "Android 4.1");

      iPhone.setPrice(5588);
      noteTwo.setPrice(4630);

      cout << "Specification of iPhone:" << endl;
      iPhone.showConfig();
      cout << "\nSpecification of Note 2:" << endl;
      noteTwo.showConfig();

      return 0;
}
```

Sample result:

```
Specification of iPhone:
Brand:  Apple
OS:     iOS version 6
Price:  $5588.00

Specification of Note 2:
Brand:  Samsung
OS:     Android 4.1
Price:  $4630.00
```

Q4. The following program demonstrates simple operations of bank accounts. You need to

(a) Create a class called **Account** which has a data member **balance** (of type *double*) that represents the account balance; a **constructor** with one double type parameter that initializes the balance; and three member functions:
   (i) **credit** – has one double type parameter and adds the amount to the current balance;
   (ii) **debit** – has of one double type parameter and deducts the amount from the current balance;
   (iii) **getBalance** – that returns the value of data member balance to the caller.

(b) Implement the function **transfer( )** that performs money transaction from account **a1** to account **a2**. The function starts by prompting for user input of the amount of transaction. If there is enough balance, then the transaction is performed. Otherwise, an error message is displayed.

The sample result is produced after the codes for parts (a) and (b) are filled in correctly.

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

// (a) Implementation of class Account should be inserted here

// Transfer money from a1 to a2, why pass-by-reference?
void transfer(Account &a1, Account &a2)
{
      // (b) code for function body should be inserted here
```

```
}

int main()
{
      Account peter(1000), mary(1500);
      int option;

      cout << fixed << setprecision(2);   // 2 decimal places

      do {
            cout << "\n";
            cout << "Peter's balance: " << peter.getBalance() << endl;
            cout << "Mary's balance:  " << mary.getBalance() << endl;
            cout << "------\n";
            cout << "(1) Transfer money from Peter to Mary\n";
            cout << "(2) Transfer money from Mary to Peter\n";
            cout << "(3) Quit\n";
            cout << "Option: ";
            cin >> option;

            switch (option) {
            case 1: transfer(peter, mary); break;
            case 2: transfer(mary, peter); break;
            case 3: cout << "Bye Bye.\n"; break;
            default: cout << "Incorrect option.\n";
            }
      } while (option != 3);

      return 0;
}
```

Sample result:

```
Peter's balance: 1000.00
Mary's balance:  1500.00
------
(1) Transfer money from Peter to Mary
(2) Transfer money from Mary to Peter
(3) Quit
Option: 1
How much to transfer: 5000
Insufficient balance.

Peter's balance: 1000.00
Mary's balance:  1500.00
------
(1) Transfer money from Peter to Mary
(2) Transfer money from Mary to Peter
(3) Quit
Option: 1
How much to transfer: 123.45
Transfer completed.

Peter's balance: 876.55
Mary's balance:  1623.45
------
(1) Transfer money from Peter to Mary
(2) Transfer money from Mary to Peter
(3) Quit
Option: 3
Bye Bye.
```