Assignment 5

Please answer the following questions.

Q 1: Choose the correct answer.

1) Which of the following is an **incorrect** constructor type?

A.	**Friend**	**B.**	**Default**	**C.**	**Default Copy**	**D.**	**Parameterized**

2) What is the number of parameters that a default constructor requires?

A.	**1**	**B.**	**3**	**C.**	**0**	**D.**	**2**

3) Which of the following operators cannot be overloaded?

A.	**>**	**B.**	**%**	**C.**	**>>**	**D.**	**None of them**

4) A member function uses object and ..... operator to access private members of a class.

A.	**dot**	**B.**	**Conditional**	**C.**	**Scope resolution**	**D.**	**Size of**

5) Constant objects can be used only with ..... functions

A.	**Friend**	**B.**	**Virtual**	**C.**	**Static**	**D.**	**const**

6) For declaring a constant function, the keyword const is placed after function .....

A.	**Definition**	**B.**	**Declaration**	**C.**	**Body**	**D.**	**Call**

7) ..... is a programming mechanism that binds together code and the data it manipulates.

A.	**Polymorphism**	**B.**	**Data abstraction**	**C.**	**Data Hiding**	**D.**	**Encapsulation**

8) How can the concept of encapsulation be achieved in C++?

A.	**By Access specifier**	**B.**	**By Abstraction**	**C.**	**By private members**	**D.**	**By Inheritance**

9) The object cannot be passed .....

A.	**By copy**	**B.**	**As function**	**C.**	**By value**	**D.**	**By reference**

10) Which of the following operators cannot be overloaded?

A. ?: **B. %=** **C. . (dot operator)** **D. A and C**

Q 2: Complete each of the following questions with correct word.

1) The combination of abstraction of the data and code is viewed in the .....

2) The special character related to destructor is: telda(~)

3) The private keyword is called access Specifier

4) In C++, the feature of OOP which derives the class from another class is called Inheritance

5) The member function that gets called when an object is being created is the Constructor

6) A destructor is used to destroy the object(member data) that has been created by a constructor.

7) The feature of OOP that describes the reusability of code is called inheritance

8) Destructor is executed automatically when the control reaches end of the class scope.

9) Object is said to be an instance of the class.

10) Variables and constants of a class are called data Member Data

Q 3: Answer the following question.

John has two accounts in a bank, a saving account with account number 30010020 and balance 5000$ and
a current account with account number 40010020 and balance 2000$. Write a C++ code to do the following:

1. Create a class for each account.
2. Create a constructor for each account that requires a customer name, account number, and balance.

3. A method to transfer an amount of money from the current account to the saving account.
4. Write a test program to do the following.

- Save the above data to John's accounts.
- A transaction to transfer 300$ from the current account to the saving account.
- Display the balance of each account after the transaction.

Accounts.cpp

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

#define ll long long

class CurrentAccount;
class SavingAccount{
    private:
        string _name;
        ll int _number;
        long double _balance;
    public:
        // Constructor
        SavingAccount(string na, ll int nu, long double ba) : _name(na), _number(nu), _balance(ba) {

        }

        // Functionality
        friend void transfer(CurrentAccount&, SavingAccount&, long double);
        void display(){
            printf("SavingAccount(balance = $%.2Lf)\n", _balance);
        }
};

class CurrentAccount{
    private:
        string _name;
        ll int _number;
        long double _balance;
    public:
        // Constructor
        CurrentAccount(string na, ll int nu, long double ba) : _name(na), _number(nu), _balance(ba) {

        }

        // Functionality
        friend void transfer(CurrentAccount&, SavingAccount&, long double);
        void display(){
            printf("CurrentAccount(balance = $%.2Lf)\n", this->_balance);
        }
};


void transfer(CurrentAccount &C, SavingAccount &S, long double money){
    C._balance -= money;
    S._balance += money;
}

int main(){
    SavingAccount SAVING_Account("John", 30010020, 5000);
    CurrentAccount CURRENT_Account("John", 40010020, 2000);

    // transfer
    transfer(CURRENT_Account, SAVING_Account, 300);

    // dislay
    SAVING_Account.display();
    CURRENT_Account.display();
    return 0;
}

/* output
    PS GU\OOP\LEC\Assignments\#5\code> c++ main.cpp -o main.exe; .\main.exe
    SavingAccount(balance = $5300.00)
    CurrentAccount(balance = $1700.00)
*/
```