UNIVERSITY OF SWAZILAND

FACULTY OF SCIENCE

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

PROGRAMMING TECHNIQUES II

COURSE CODE – EE272

MAIN EXAMINATION

MAY 2011

DURATION OF THE EXAMINATION - 3 HOURS

INSTRUCTIONS TO CANDIDATES

1. There are **SIX** questions in this paper. Answer questions 1 & 2 and any other **TWO** questions.
2. Each question carries equal marks.
3. Show all your steps clearly in any calculations.
4. State clearly any assumptions made.
5. Start each new question on a fresh page.

Question 1

(i) Discuss four restrictions on operator overloading in C++. In each case give an example to illustrate. [4]

(ii) How is it that polymorphism enables programming "in the general" rather that "in the specific"? Discuss two advantages of programming "in the general". [4]

(iii) Distinguish between static binding and dynamic binding. Explain the use of virtual functions and the *vtable* in dynamic binding. [4]

(iv) In what way does polymorphism promote extensibility. [1]

(v) Discuss two problems of programming with the `switch` logic. Using an example, explain how polymorphism can be an effective alternative to switch logic. [4]

(vi) Using an example, explain the relationship between function templates and function overloading? [4]

(vii) Using examples, discuss four ways by which class templates and inheritance are related. [4]

Question 2

a) Information hiding is one of the key features that distinguish object-oriented programming from structured programming. Using an example, explain the rationale of information hiding and how it relates to the following object-oriented programming concepts: *abstraction*, *coupling*, and *cohesion*. [4]

b) Explain the following Object-Oriented programming terms:
   a. Polymorphism [2]
   b. Class Constructor [2]
   c. Interface [2]

c) Discuss the ways in which inheritance promotes software reuse, saves time during program development and helps prevent errors. [4]

d) Describe the ways by which a derived class may inherit from a base class. [6]

e) Explain the advantage of separating interface from implementation of a class. [2]

f) Explain the difference between local variable and a data member? [2]

g) Explain why a class might provide a *set* and a *get* function for a data member. [1]

Question 3

A bank requires a software system for managing customers' bank accounts. All customers at this bank can deposit into and withdraw from their accounts. The bank has two types of accounts: *savings* and *current*. A savings account earns interest on the money held meanwhile a current account charges a fee per transaction. The required software system consists of three classes related through an inheritance hierarchy containing base class *Account* and derived classes *SavingsAccount* and *CurrentAccount* that inherit from class *Account*.

Class *Account* has a data member of type double to represent the account balance. Its constructor receives and validates an initial balance and uses it to initialize the balance. *Account* also has three member functions: *credit*, *debit*, and *getBalance*. *Credit* adds an amount to the current balance and *debit* deducts an amount from the account and ensures that the debit amount does not exceed the account balance. G*etBalance* returns the current balance.

Class *SavingsAccount* inherits the functionality of *Account*, but also include a data member of type double indicating interest rate (percentage) assigned to the account. Its constructor receives the initial values for the balance and interest rate. The class also has a public member function, *calculateInterest,* which returns a double indicating the amount of interest earned obtained by multiplying the interest rate by the account balance.

Class *CurrentAccount* also inherits from *Account* and includes an additional data member of type double that represents the fee charged per transaction. Its constructor receives the initial balance, as well as a parameter indicating a fee amount. This class redefines functions *credit* and *debit* so that they deduct a fee from the account balance whenever a transaction is performed successfully. Its version of these functions invoke *Account's* version to perform the updates to an account balance and its debit function charge a fee only if money is actually withdrawn.

(i) Draw a class diagram depicting the three classes and their relationship. [3]
(ii) Write the C++ interface of each class. [6]
(iii) Write the C++ implementation of each class. [12]
(iv) Write a C++ program that creates objects of each class and tests their member functions. [4]

Question 4

(a) Declare a class named *Triple* with three private data members (floats) x, y, and z. Provide public functions for setting and getting values of all the private data members. Define a constructor that initializes the values to user-specified values or, by default, sets the values all equal to 0. Also overload the following operators:

i. *Addition* (+) so that corresponding elements are added together
ii. *Output* (<<) so that it displays the Triple in the form "The triple is (x,y,z)"
iii. *Assignment* (=) that copies x to z, y to x, and z to y.
iv. *Post-increment* (++) so that x and z are increased by one each.
v. *Input* (>>) such that it is possible to input a Triple in the form:
(x, y, z) e.g. (3, 2, 5)

[17]

(b) Analyse the following program and determine its output:

```
#include <iostream>

class A{
      protected:
            int a;
      public:
            A(int x=1) {a=x;}
            void f(){a+=2;}
            virtual g(){a+=1;}
            int h() {f(); return a;}
            int j() {g(); return a;}
};

class B: public A{
      private:
            int b;
      public:
            B(){int y=5){b=y;}
            void f(){b+=10;}
            void j(){a+=3;}
};

int main(){
      A obj1;
      B obj2;

      cout<<obj1.h()<<endl;
      cout<<obj1.g()<<endl;
      cout<<obj2.h()<<endl;
      cout<<obj2.g()<<endl;

      return 0;
}
```

[8]

Question 5

Specify a class for solving simultaneous equations. Call this class *EquationSolver*. Your call should provide private data members for coefficients. It should also provide four functions. The first function should be for reading the coefficients. Call this function *enterCofficients*. The second function solves the equation using substitution. The third function solves the equation using elimination. Call the second and third functions *solveBySubstitution* and *solveByEllimination*, respectively. Finally, the fourth function, printSolutions, should print both equations and the solutions. The output should look as follows:

Equations:

$x + y = 24$

$2x - y = -6$

Solutions:

$x = 6$

$y = 18$

Write a program that tests the functionality of *EquationSolver*. Your program should give the user the option of either solving by substitution or by elimination. When defining your class feel free to add any data members and member functions when necessary.

Allocation of Marks:

Program Clarity	[3]
Class interface definition	[3]
Class implementation	[16]
Test Program	[4]

Question 6

Create a class called `Invoice` that a hardware store might use to represent an invoice for an item sold at the store. An `Invoice` should include four pieces of information as data members – a part number (type string), a part name (type string), a quantity of the item being purchased (type double), a price per item (type double), and a value added tax (VAT) in percent (type double). Your class should have a constructor that initialises the four data members. Provide set and get functions for each data member. In addition, provide a member function named *InvoiceAmount* that calculates the invoice amount (i.e. multiplies the quantity by the price per item and then add the VAT), then returns the amount as a double value. If the quantity is not positive it should be set to 0.0. If the price per item is not positive it should be set to 0.0.

(i) Write an interface and implementation for this class.

[20]

(ii) Write a test program that creates two Invoice objects. The first object represent the purchase of two shovels (part # SH00101), each costing E250. The second object represent the purchase of nine bags of cement (part # CM01012), each costing E65. Assume that the current VAT set by the government is 14%. Your test program should print the values of each invoice.

[5]

END OF PAPER