UNIVERSITY OF SWAZILAND

FACULTY OF SCIENCE AND ENGINEERING

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

MAIN EXAMINATION 2015

Title of Paper: Programming Techniques II

Course Number: EE272

Time Allowed: 3 hrs

Instructions:

1. There are **five** (5) questions in this paper. Answer question 1 and any other **three** (3) questions.
2. Each question carries 25 marks.
3. State clearly any assumptions made.
4. Start each new question on a fresh page

THIS PAPER SHOULD NOT BE OPENED UNTIL PERMISSION HAS BEEN GIVEN BY THE INVIGILATOR.

This paper contains nine (8) pages including this page.

Question 1

a) Explain why a class might provide a *set* and *get* functions for a data member? [2]

b) What is the role of an abstract class? [2]

c) Using an example, explain the difference between an *object pointer* and an *object reference.* [2]

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

e) Describe the ways by which a derived class may inherit from a base class. [5]

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

g) Explain how polymorphism promotes extensibility of software design? [4]

h) The definition of classes should promote encapsulation. Explain. [2]

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

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

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

f) Explain 3 ways in which the members of a class can be accessed in the class's clients? [3]

g) Using an example, explain where you would use a unary scope resolution operator. [2]

Question 3

Analyse the following two programs and determine their outputs.

a) **[15]**

```
#include <iostream>
using namespace std;
class poly
{
    protected:
    int width, height;
    public:
    void set_values(int a, int b)
    {
        width = a; height = b;
    }
};
class Coutput
{
    public:
    void output (int i);
};
void Coutput::output(int i)
{
    cout « i;
}
class rect:public poly, public Coutput
{
```

```
    public:
    int area ()
    {
        return(width * height);
    }
};

class tri:public poly, public Coutput
{
    public:
    int area ()
    {
        return(width * height / 2);
    }
};
int main ()
{
     rect rect;
     tri trgl;
     rect.set_values(3, 4);
     trgl.set_values(4, 5);
     rect.output(rect.area());
     trgl.output(trgl.area());
     return 0;
}
```

b) [10]

```
#include <iostream>

using namespace std;

class Box

{
    double width;
    public:
    friend void printWidth( Box box );
    void setWidth( double wid );


};

void Box::setWidth( double wid)

{

    width = wid;

}

void printWidth( Box box)

{

    box.width = box. width * 2;

    cout « "Width of box :" « box.width « endl;

}

int main ( )

{


    Box box;


    box.setWidth(10.0);


    printWidth( box );


    return 0;

}
```

Question 4

Create a class called *Employee* that includes three pieces of information as data members—a first name (type string), a last name (type string) and a monthly salary (type float). Your class should have a constructor that initializes the three data members. Provide a *set* and a *get* function for each data member. If the monthly salary is not positive, set it to 0.

(i) Write an interface and implementation for this class. [15]

(ii) Write a test program that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10 percent raise and display each Employee's yearly salary again. [10]

Question 5

Package-delivery services, such as FedEx, DHL, and UPS, offer a number of different shipping options, each with specific costs associated.

Create an inheritance hierarchy in the form of a class diagram to represent the various types of packages. Use `Package` as the base class of the hierarchy, then include classes `TwoDayPackage` and `Overnight` that derive from `Package`. Base class `Package` should include data members representing the *name, address, city,* and *region* for both the sender and the recipient of the package, in addition to data members that store the *weight* (in kilograms) and *cost per kilogram* to ship the package. Package's constructor should initialise these data members. Ensure that the weight and cost per kilogram contain positive values.

Package should provide a public member function `calculateCost` that returns a double indicating the cost associated with shipping the package. `Package`'s `calculateCost` function should determine the cost by multiplying the weight by the cost per kilogram. Derived class `TwoDayPackage` should inherit the functionality of base class `Package`, but also include a data member that represents a flat fee that the shipping company charges for two-day-delivery service. `TwoDayPackage`'s constructor should receive a value to initialise this data member. `TwoDayPackage` should redefine member function `calculateCost` so that it computes the shipping cost by adding the flat fee to the weight-based cost calculated by base class `Package`'s `calculateCost` function.

Class `OverNightPackage` should inherit directly from class `Package` and contain an additional data member representing an additional fee per kilogram charged for overnight-delivery service. `OverNightPackage` should redefine member function `calculateCost` so that it adds the additional fee per kilogram to the standard cost per kilogram before calculating the shipping cost.

(i) Write the C++ interface of each class [12]

(ii) Write the C++ implementation of each class [8]

(iii) Write a C++ program that creates objects of each type of package and test their member function `calculateCost`. [5]

END OF PAPER