

REGENT UNIVERSITY
COLLEGE OF SCIENCE AND TECHNOLOGY



**SCHOOL OF INFORMATICS AND
ENGINEERING**
END OF SEMESTER EXAMINATION

EXAMINATION PAPER

**SICS 152: OBJECT ORIENTED
PROGRAMMING USING C++**

DATE: 16TH NOVEMBER 2007

TIME ALLOWED: 2 HOURS

EXAM MATERIAL PROVIDED: NONE

INSTRUCTIONS: *ATTEMPT ALL QUESTIONS*

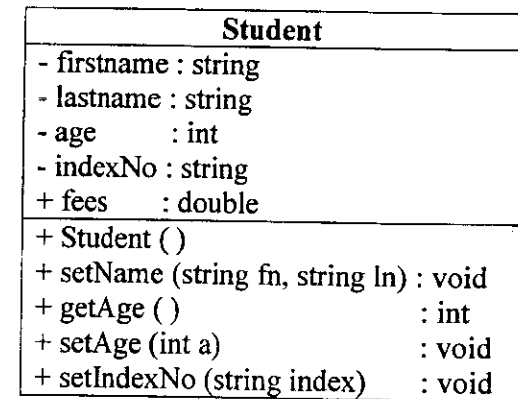
LECTURER: MR KENNETH AZUMAH

Attempt all questions.

1. Answer **True** or **False** for (a) to (j) [10 marks]
 - a. The value of the expression `13%4` is 3
 - b. A **do**-loop is executed at least once
 - c. `int hour; hour = "12";` generates a compiler error
 - d. `int boys[5]; int b = boys[5];` is valid code
 - e. In using a object oriented languages like C++ we can define our own data types.
 - f. In a structure members are **public** by default
 - g. Data members of a class must be declared **private**
 - h. In a class members are **private** by default
 - i. Members declared as **private** in a class are accessible to all the member functions of that class.
 - j. A class constructor has the **int** return type by default

2. Fill in the blanks (write answer in your answering booklet) [10 marks]
 - a. The wrapping up of data and functions into a single unit is called
 - b. The process by which objects of one class acquire the attributes of objects of another class is known as
 - c. Every C++ statement ends with a
 - d. Every C++ program begins execution at the function
 - e. A function that has return type does not return anything
 - f. Member functions of a class are normally declared as
 - g. A constructor's name is the same as

- h. In designing with inheritance, attributes common to classes are normally moved higher up the
 - i. The elements of an array of size 10 are numbered from to
 - j. A function prototype tells the compiler the return type, name and
3. Implement the following class diagram using the C++ language. [20 marks]
(Properly indent and adequately comment your code. **main** function not required)



4. Below are two classes with common attributes. [30 marks]
Using the diagram below modify the classes taking following into consideration.
 - a. Fully implement a base class Account
 - b. Fully re-implement the Savings and Checking class.

Fully implement all the member functions.
(Fill in with the appropriate code to offer
functionality as name implies)

```
class Savings{
private:
    int accNo;
    double accBalance;
    double interestRate;
public:
    Savings();
    void Deposit(double deposit)
    void Withdraw(double amt);
    double getBalance();
};
```

```
class Checking{
private:
    int accNo;
    double accBalance;
public:
    Checking();
    void Deposit(double deposit)
    void CashCheque(double amt);
    double getBalance();
};
```

