Class Test 8 (18 Marks)

Given the code below:

```cpp
class Employee {
protected:
    string name;
    int grossSalaryAmount;

public:
    Employee(string n, int grossSalary) : name(n),
grossSalaryAmount(grossSalary) {}

    int grossSalary() {
        return grossSalaryAmount;
    }

    int netSalary() {
        int tax = 0;
        if (grossSalaryAmount <= 237100)
            tax = grossSalaryAmount * 0.18;
        return grossSalaryAmount - tax;
    }
};
```

1) Which of the following is true about grossSalary():

a) The method grossSalary() is a setter method that modifies the gross salary.
b) The method grossSalary() calculates the gross salary after taxes.
c) The method grossSalary() returns the value of grossSalaryAmount.
d) The method grossSalary() should be declared as private. (1)

2) What is the purpose of the protected access modifier/specifier in the Employee class?

a) It allows access only to members from within the Employee and child classes.
b) It allows only the Employee class to access the members.
c) It makes the member variables available to any class or function.
d) It provides the highest level of protection and prevents access from other classes. (1)

3) What does the netSalary() method do in the Employee class?

a) It calculates the gross salary after deductions.
b) It calculates the employee’s net salary by subtracting the tax from the gross salary.
c) It returns the gross salary without considering any tax brackets.
d) It overrides the grossSalary() method to provide a more detailed salary calculation. (1)

Given the code below:

```cpp
class Employee {
public:
    virtual void showDetails() {
        cout << "Employee Details" << endl;
    }
};

class Manager : public Employee {
public:
    void showDetails() override {
        cout << "Manager Details" << endl;
    }
};
```

4) What is Method overriding?

a) Defining multiple methods with the same name but different parameters in the same class.
b) Reusing the method of a base class method in the child class and providing a new implementation.
c) Changing the return type of a method in the base class while keeping the same parameters in the derived class.
d) Using the same method name in two unrelated classes without connection. (1)

5) What is the significance of the virtual keyword in the Employee class?

a) It allows the class to define multiple constructors.
b) It prevents any modifications to the class members.
c) It forces all child classes to implement the method.
d) It enables polymorphism by allowing the method to be overridden in derived classes (1)

Given the code below:

```
class Manager : public Employee {
private:
    string department;

public:
    Manager(string n, int grossSalary, string dept) : Employee(n,
grossSalary), department(dept) {}

    void showDetails() override {
        cout << "Name: " << name << endl;
        cout << "Department: " << department << endl;
        cout << "Gross Salary: R" << grossSalary() << endl;
        cout << "Net Salary: R" << netSalary() << endl;
    }
};
```

6) Which of the following statements is true?

a) The department attribute is publicly accessible from outside the class.
b) The department attribute can only be accessed by methods within the Manager class.
c) The department attribute is inherited from the Employee class.
d) The grossSalary() method is overridden in the Manager class. (1)

```
class Employee {
protected:
    string name;
    int grossSalaryAmount;

public:
    Employee(string n, int grossSalary) : name(n),
grossSalaryAmount(grossSalary) {}

    int grossSalary() {
        return grossSalaryAmount;
    }

    int netSalary() {
        int tax = 0;
        if (grossSalaryAmount <= 237100)
            tax = grossSalaryAmount * 0.18;
        return grossSalaryAmount - tax;
    }
};
```

```
class Manager : public Employee {
private:
    string department;

public:
    Manager(string n, int grossSalary, string dept) : Employee(n,
grossSalary), department(dept) {}

    void showDetails() override {
        cout << "Name: " << name << endl;
        cout << "Department: " << department << endl;
        cout << "Gross Salary: R" << grossSalary() << endl;
        cout << "Net Salary: R" << netSalary() << endl;
    }
};
```

7) Using the above Employee and Manger example, explain inheritance and its benefits. (4)

8) Using the above Employee and Manger example, explain polymorphism. (4)

9) Using the above Employee and Manger example, explain why protected access specifier is used instead of Private (4)