

Homework 11: Inheritance and Derived Classes

Instructor: Mehmet Emre

CS 32 Spring '22

Due: 5/18 2pm

Name & Perm #:

Homework buddy (leave blank if you worked alone):

Reading: PS 15.1, 15.2, DS 14.1

1

For each statement, indicate if it is True or False by circling T or F. If you need to cross out an answer, be sure that your final answer is clear and unambiguous—otherwise it will receive no credit.

- | | | | |
|---------|---|---|---|
| (3 pts) | An object of a derived class has access to the public methods of its base class | T | F |
| (3 pts) | An object of a base class has access to the private helper methods of its derived class | T | F |
| (3 pts) | Destructors are not inherited by derived classes | T | F |
| (3 pts) | An object of a derived class inherits the copy constructor of its base class | T | F |
| (3 pts) | Operators are passed down inheritance hierarchies | T | F |
| (3 pts) | Destructors in derived classes are called after their base class calls its destructor | T | F |
| (3 pts) | Constructors of base classes are accessible by derived classes | T | F |

2

Assume there is a class called **Student** that has private member variables `string name`, `int perm`. Assume that **Student** has getters and setters for each of these data members `getName`, `setName`, `getPerm` and `setPerm`, and a constructor that takes `name` and `perm` as parameters. Assume that the class is declare in file `student.h`. In other words, you are given the following class definition:

```
class Student {  
    std::string name;  
    int perm;  
public:  
    Student(std::string name, int perm) :  
        name(name), perm(perm) {}  
  
    const std::string & getName() const {  
        return name;  
    }  
  
    void setName(std::string name) {  
        this->name = name;  
    }  
}
```

```
int getPerm() const {
    return perm;
}

void setPerm(int perm) {
    this->perm = perm;
}
}
```

1. (8 pts) Write the contents of a .h file for a derived a class called **CmpscStudent** that inherits from **Student** and has additional data members including `*string* ugradDegreeType`, `*bool* graduateStudent`. Include prototypes for a public constructor that initializes all of the data members of **CmpscStudent** as well as getters (but not setters) for the additional data members of CmpscStudent. Do not implement these constructors and methods as inline. Just give the prototypes in the class definition.

2. (8 pts) Write the function definition for the constructor **CmpscStudent** that takes as its parameters all of the data members of CmpscStudent, and fully initializes the object being constructed, as it would appear in the .cpp file.