

## CS2260, Fall 2003, Test 2

Time 70 min. All questions are weighted the same. Use extra paper as needed, but make sure to identify each answer.

YOU MUST RETURN THIS PAGE. NAME \_\_\_\_\_

- 1 Implement class **A** with private integer **a**, **get\_a()**/**set\_a()** methods as needed (use pass by value for **set\_a()** and pass by reference for **get\_a()**), and two constructors: w/o argument sets **a=0**, and w/argument sets **a** to the value of the argument. Show header and implementation files separately.
- 2 Derive class **B1** from **A**. **B1** has additional private integer **b**, **get\_b()**/**set\_b()** methods as needed, and three constructors:  
**B1()** // set a=b=0  
**B1(x)** // set a=x, b=x  
**B1(x,y)** // set a=x, b=y, do not use **set\_a()** if possible  
 Show both files.
- 3 Derive class **B2** from **A**. **B2** has nothing new except that when asked **get\_a()** it should say "Not your business". Show both files.
- 4 Using #2, application program writes:  
**B1 b1;**  
 Write subsequent code to set **a=5, b=10;**
- 5 Using #3, application program writes:  
**B2 b2;**  
 Write subsequent code to display the values of **a** to the screen if possible.;
- 6 What is **abstract class** and what is **virtual base class**. Show example of each.
- 7 What to do in #1 and #2 to have **get\_a()** polymorphic? Show details.
- 8 Using #7 (polymorphic case) suppose an application program has  
**A \*ap = new A;**  
**B1 \*bp = new B1;**  
**A \*ap = new B1;**  
**B1 \*bp = new A;**  
 Which ones are errors, and for those that are ok what will be printed if you invoke the **get\_a()** method through the pointer?
- 9 For #1, overload the ++operator so that  
**A a;**  
**cout << a++;**  
**cout << ++a;**  
 will both print "hello" and will increment the value of **a** by 1.  
 Show only modifications in #1.

10 For #1, overload the binary + so that

```
A a(2);
```

```
A a1;
```

```
a1=a+5;
```

will create **a1** object with its private member equals the private data from **a** plus 5 (7 in this case).

11 Now allow

```
a1=5+a;
```

to do exactly the same. Do it i) using friend and ii) using access methods and iii) reversing the arguments (using the solution in #10).

12 Overload << output for **A** in #1 to display the value of the member **a**.

13 Overload << output for **B1** in #2 to display both **a** and **b**. Do it i) using the access method and ii) not using the access method if possible.