First Name Last Name

First Name Last Name

First Name Last Name

Review: INHERITANCE

1.

A. Circle the name of the base class
B. Circle the name of the derived class
C. What is the output?

```
class House
{
    public:
    House(){ cout << "In the house.\n";}
    ~House() { cout << "Out of the house.\n";}
};

class Room : public House
{
    public:
    Room(){ cout << "In the room.\n";}
    ~Room(){ cout << "Out of the room.\n";}
};

// In main()
    Room myRoom;
```

OUTPUT:

2. What is the output?

```
class House
{
    public:
    House(){ cout << "In the house.\n";}
    House(double fahr)
        { cout << fahr << " degrees in the house" << endl;}
    ~House() { cout << "Out of house house.\n";}
};

class Room : public House
{
    public:
    Room(){ cout << "In the room.\n";}
    Room(double t1, double t2) : House (t1)
          { cout << t2 << " degrees in the room" << endl;}
    ~Room(){ cout << "Out of the room.\n";}
};

// In main()
    Room myRoom(65, 70);
```

OUTPUT:

Review: INHERITANCE

3. Given the following:

```
class Alpha
{
    private:
        int one;
        int two;
    protected:
        int six;
        void setOne(int a) { one = a; }
        void setTwo(int b) { two = b; }
    public:
        void setSix(int c) { six = c; }
};
```

```
//A.
class Beta : public Alpha
```

Indicate whether each member of the Alpha class is
`private`, `protected`, `public`, or `inaccessible`:

`one` `two` `six` `setOne` `setTwo` `setSix`

```
//B.
class Beta : protected Alpha
```

Indicate whether each member of the Alpha class is
`private`, `protected`, `public`, or `inaccessible`:

`one` `two` `six` `setOne` `setTwo` `setSix`

```
//C.
class Beta : private Alpha
```

Indicate whether each member of the Alpha class is
`private`, `protected`, `public`, or `inaccessible`:

`one` `two` `six` `setOne` `setTwo` `setSix`