Announcements

MP2 available, due 9/16, 11:59p.

Where were we?

Where were we?

Write the copy constructor function signature as it appears in the sphere class definition:

List two instances in which a class’s copy constructor is called:

1.

2.

Write the destructor function signature as it appears in the sphere class definition:
The destructor, a summary:

1. Destructor is never “called.” Rather, we provide it for the system to use in two situations:
   a) __________________
   b) __________________

2. If your constructor, __________________, allocates dynamic memory, then you need a destructor.

3. Destructor typically consists of a sequence of delete statements.

```cpp
class sphere{
public:
   // tons of other stuff
   ~sphere();
private:
   double theRadius;
   int numAtts;
   string * atts;
};
```
One more problem:

class sphere{

public:
    sphere();
    sphere(double r);
    sphere(const sphere & orig);
    ~sphere();
    ...

private:
    double theRadius;
    int numAtts;
    string *atts;
};
Overloaded operators:

```cpp
int main(){
    // declare a,b,c
    // initialize a,b,c
    c = a + b;
    return 0;
}

// overloaded operator
sphere & sphere::operator+(const sphere & s) {
}
```
Overloaded operators: what can be overloaded?

arithmetic operators, logical operators, I/O stream operators

+   -   *   /   =   <   >   +=   -=   *=   /=
<<   >>   <<=  >>=   ==   !=   <=   >=   ++   --   %   &
^   !   |   ~   &=   ^=   |=   &&   ||   %=
  []   ()   ,   ->*   ->
  new  delete  new[]  delete[]
One more problem: default assignment is memberwise, so we redefine =.

```cpp
class sphere{
public:
sphere();
sphere(double r);
sphere(const sphere & orig);
~sphere();
_________ operator=(____________________);
…
private:
double theRadius;
int numAtts;
string * atts;
};
```

```cpp
int main(){
    sphere a, b;
    // initialize a
    b = a;
    return 0;
}
```
Some things to think about...

\[ b = a \]

c = b = a

- 250690176
- 4
- wet
- rocky
- rotating
- inhabited

- 2.5
- 3
- Red
- Juicy
- crunchy

- 0.84
- 1
- dimpoled
class sphere {
public:
  sphere();
  sphere(double r);
  sphere(const sphere & orig);
  ~sphere();
...
private:
  double theRadius;
  int numAtts;
  string * attributes;
};

int main() {
  sphere a, b;
  // initialize a
  b = a;
  return 0;
}

// overloaded =
sphere & sphere::operator=(const sphere & rhs){
...
int numAtts;
string * attributes;
}