Copy Constructor / Assignment Operator

Automatically Generated Functions

- We have learned of two member functions sometimes automatically generated by the compiler
  - Constructor – An empty default (ie. no params) constructor is created if no constructor is defined.
  - Destructor – An empty destructor is created if no destructor is defined.
- Today we will discuss two other sometimes automatically generated member functions
  - Copy Constructor
  - Assignment Operator

Copy Constructor

- A copy constructor IS a constructor and therefore:
  - has the same name as the class
  - has no return type (although, it seems to return a class object when called explicitly)
- Like the conversion constructor, there are situations when the copy constructor is called implicitly. They are:
  - when an object is declared to have the same value as another object

    Example: Fraction f1(1,2);

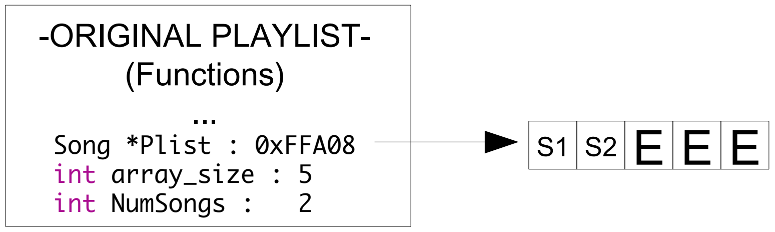
    Fraction f2 = f1; //new object f2 is initialized as a COPY of f1
  - when an object is passed *by value* into a function
  - when an object is returned *by value* from a function

Copy Constructor Declaration

- Since the purpose of a copy constructor is to initialize a new object to be a copy of another object, it accepts a single object as a parameter
- Format: classname(const classname&)
- The argument is const because the copy constructor should not alter the original (not required)
- The argument MUST be passed by reference, why?
- Examples
  - Fraction(const Fraction& f)
  - Mixed(const Mixed& m)

Shallow Copy vs Deep Copy

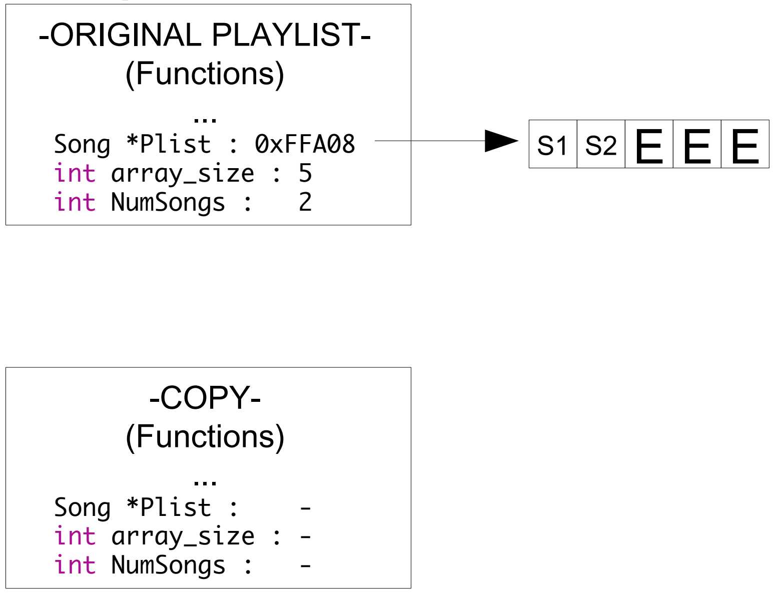
- Suppose we want to copy a playlist object:



- There are two ways we could make a copy:
  - Shallow (default) – All member data is copied EXACTLY from the old object into the new one.
  - Deep (overloaded) – New dynamic memory is created for pointers

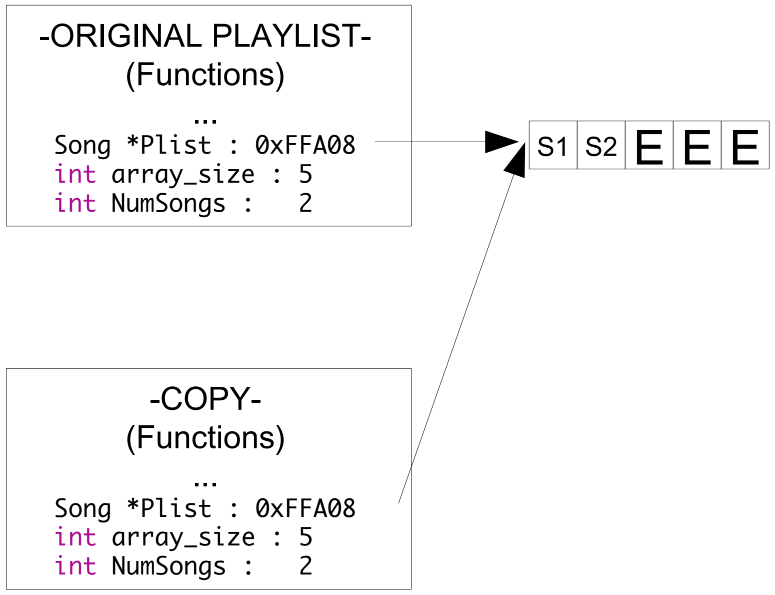
Shallow Copy

- We start in the copy constructor of COPY with the original as a parameter



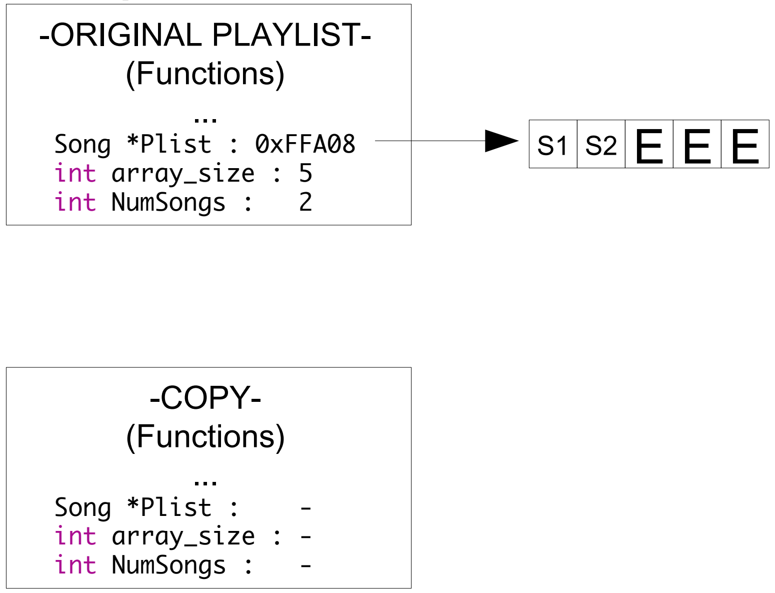
Shallow Copy

- Set data in copy equal to that of the original... DONE.



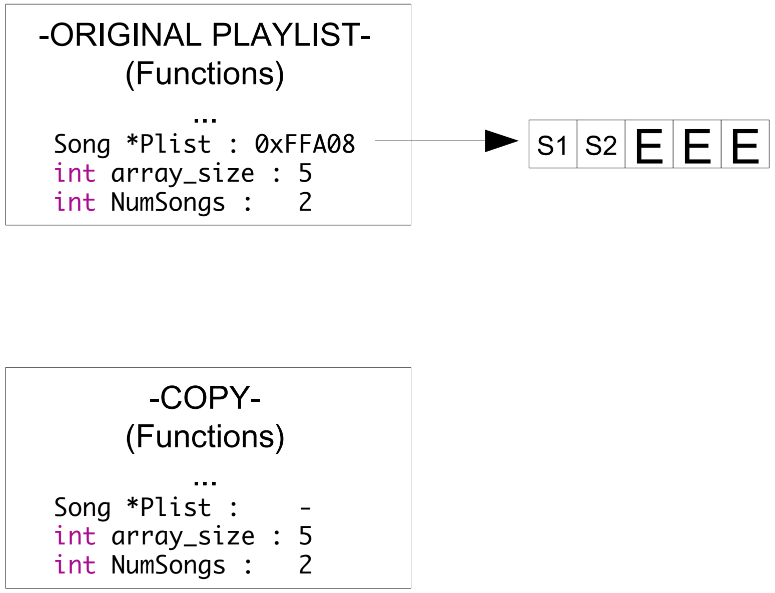
Deep Copy

- We start in the copy constructor of COPY with the original as a parameter



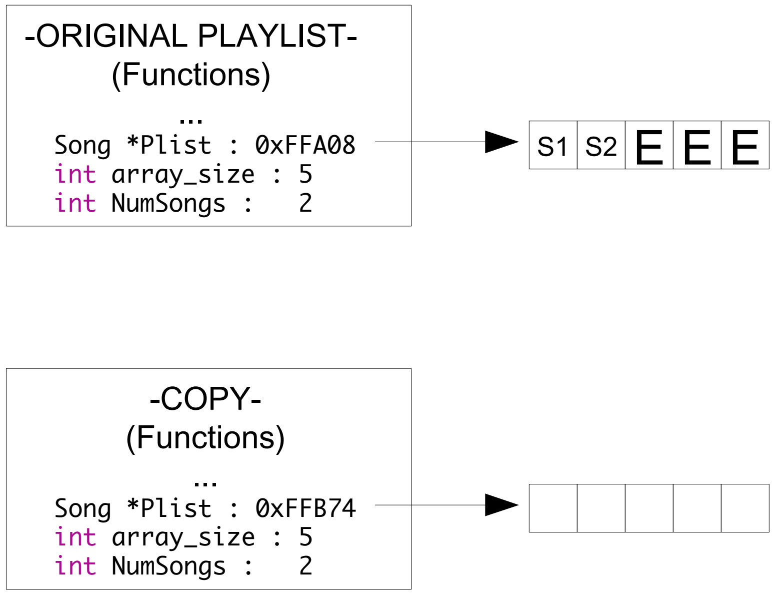
Deep Copy

- Set NON-POINTER data in the copy equal to the original



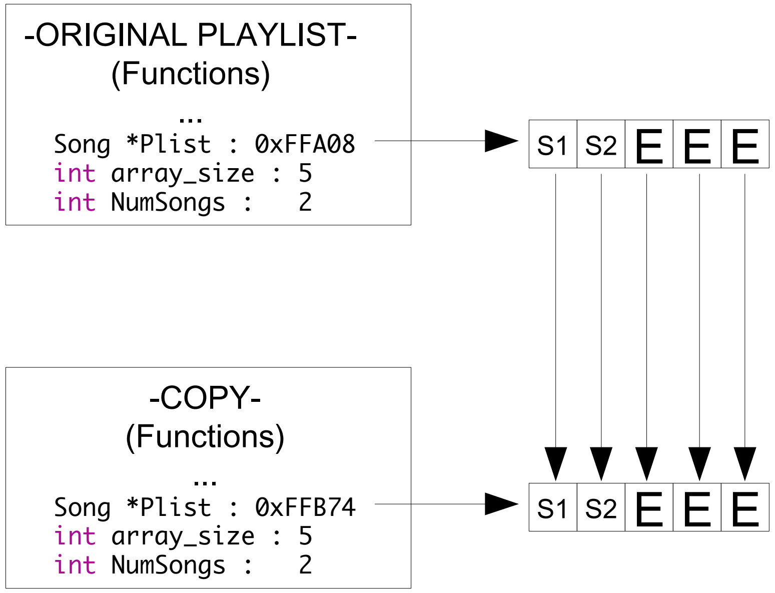
Deep Copy

- Allocate new memory for data pointer points to.



Deep Copy

- Copy data from old dynamic memory to new... DONE.



Assignment operator

- The assignment operator (=) is called when one object is assigned to another
- The assignment operator is similar to the copy constructor, but there are some key differences
  - The assignment operator is a normal member function not a constructor, this means 2 objects already exist and have been initialized
  - The assignment operator returns the value it was assigned (allows cascading calls)
    - Fraction f1(1,2),f2,f3,f4;
    - f4 = f3 = (f2 = f1);
    - f4 = (f3 = (f2))
    - (f4 = (f3))
    - (f4)

Assignment operator

- Format: classname& operator=(const classname&);
- Ex. Fraction lhs(1,2), rhs(2,5);
-       lhs = rhs;
- lhs is the calling object, rhs is the parameter, the assignment function alters lhs to be a copy of rhs and returns a reference to lhs.
- If lhs is the calling object, how can we return a reference to it?

The *this* pointer

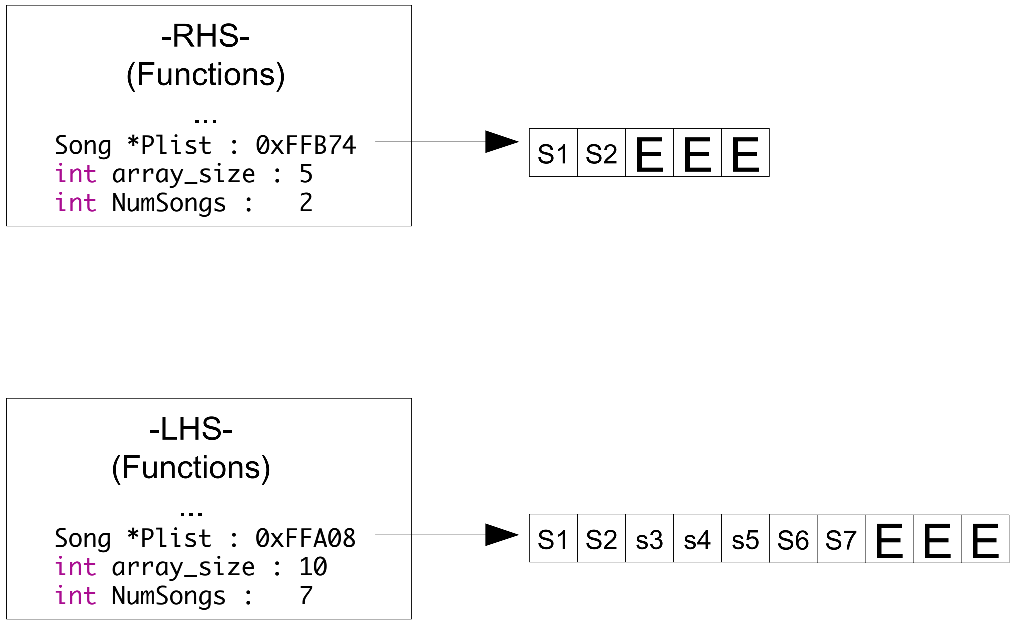
- Inside every object is a pointer named 'this'
- It's like having 'classname *this;' in the member data of an object
- The 'this' pointer is set to point to the object itself
- You can actually call another member function with the statement this->memberFunction()
- We can use the this pointer to return a reference to the object itself in the assignment operator
  - Should we return this or *this ? (this pointer or whats at this pointer?)

Deep Copy (Assignment)

- Suppose we are assigning playlist LHS to RHS (LHS=RHS;)
- The automatically generated copy constructor performs a shallow copy
- Lets see what we would have to do in order to do an overload of the assignment operator that performs a deep copy

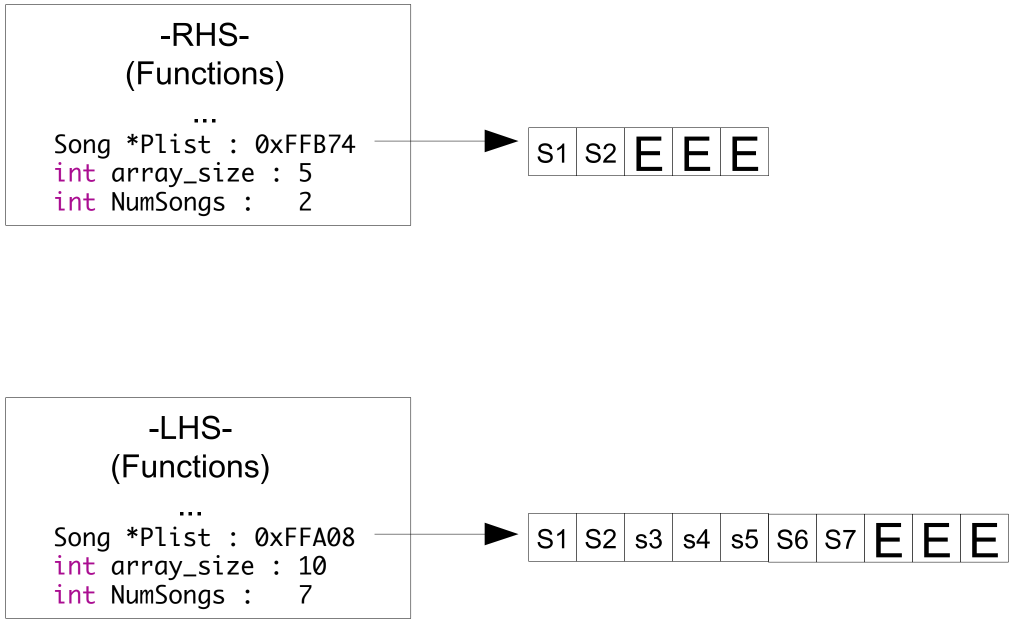
Deep Copy (Assignment)

- LHS is the calling object and already has its own member data that we want to match RHS



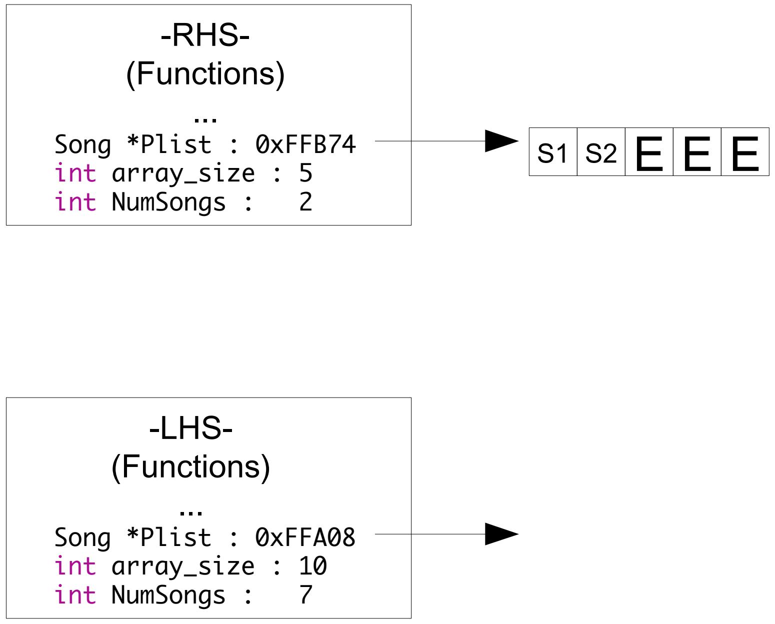
Deep Copy (Assignment)

- Since LHS's array is the wrong size, we must deallocate it and reallocate the correct size



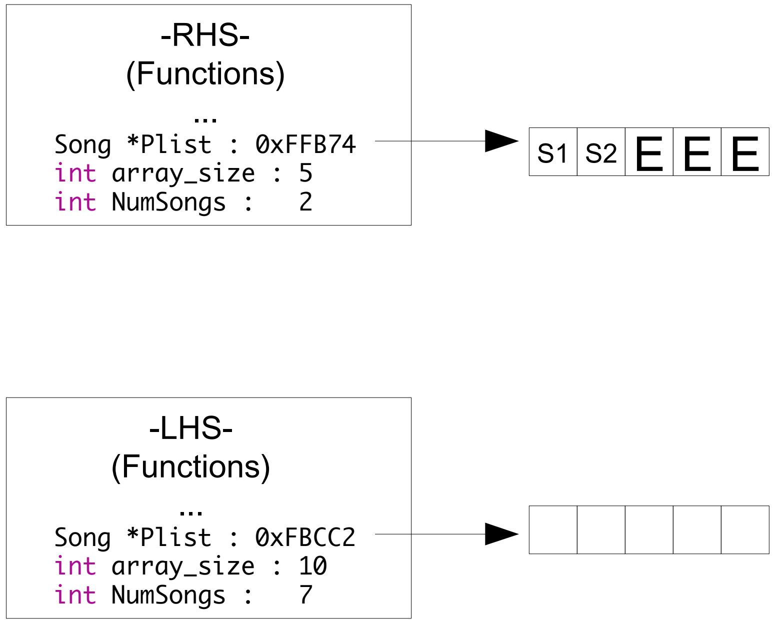
Deep Copy (Assignment)

- Since LHS's array is the wrong size, we must deallocate it and reallocate the correct size



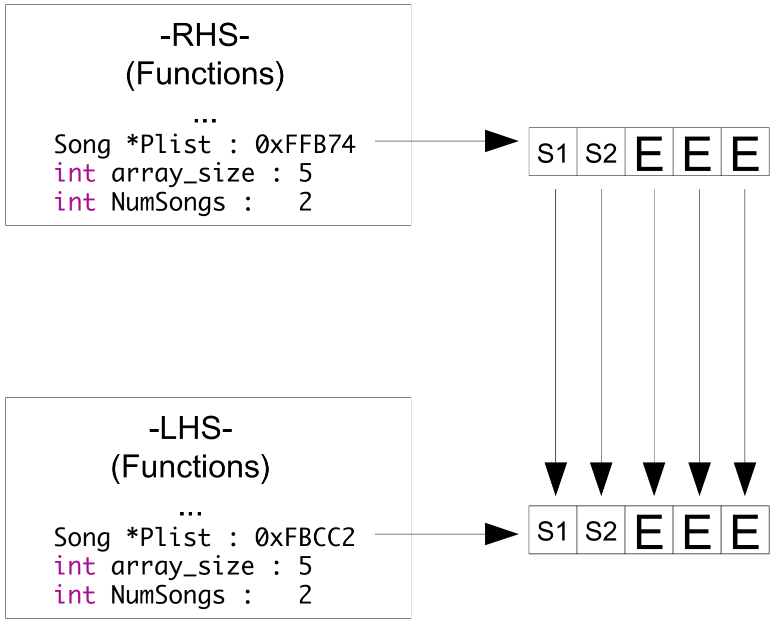
Deep Copy (Assignment)

- Since LHS's array is the wrong size, we must deallocate it and reallocate the correct size



Deep Copy (Assignment)

- We can now copy the elements of RHS to LHS and copy the other member data...DONE.



Everything else

- Assignment operator must always be a member function (can't be friend)
- Assignment operator implementation always ends with: return *this;
- If you define a copy constructor, but no other constructor, an empty default constructor WILL NOT be generated by the compiler