rec06 FAQ - Copy Control

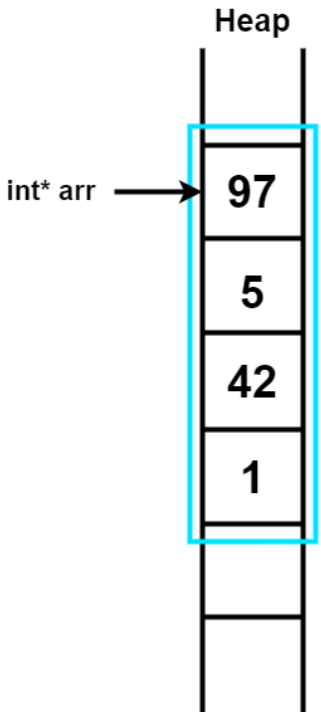
These are just some questions that I often find students ask.

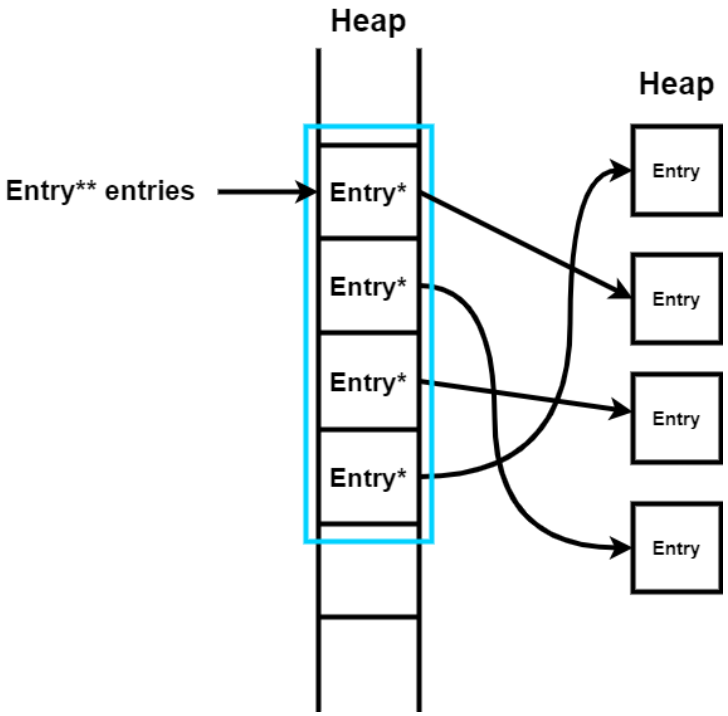
- Q: Do I need “getters” for fields in the `Directory` to be used in the `Directory` copy constructor and assignment operator?
  A: No. A class’s methods can access the private members of *any* instance of that class.

- Q: To make “copies” of the `Entry` objects, should I be using the `Directory`’s add method in the `Directory` copy constructor?
  A: No! That would be **wrong**. You just want to *initialize* a copy of the `Entry`. How do you initialize? Use a constructor, here the `Entry`’s **copy constructor**.
  - Q: Does that mean I have to write a copy constructor for the `Entry` class?
  - A: Again, no. All classes are provided with a copy constructor by the system. You only write your own if the one the system provides doesn’t do what you need.

- Q: Why does the field `entries` in the `Directory` have the type `Entry**` ??
  A: `entries` is a pointer to an array.
  - The type for a pointer to an array is “pointer to the type of an element in the array”.
  - What type of things does the `Directory`’s array hold? **Pointers** to `Entry` objects.
  - So the type of a pointer to the array is “pointer to … pointer to an `Entry` object”, or in C++ syntax: `Entry**`.

- Whats the difference between the array that we allocate for the Vector class and what we are doing here in rec06?
  - In the Vector, just the array is on the heap.
  - Here, not only is the array on the heap, but also the Entries!!!
  - When you are making a copy, think about what you need to allocate on the heap.
  - [Picture on next page.]


Heap
int* arr
97
5
42
1


Heap
Heap
Entry** entries
Entry*
Entry*
Entry*
Entry*
Entry
Entry
Entry
Entry