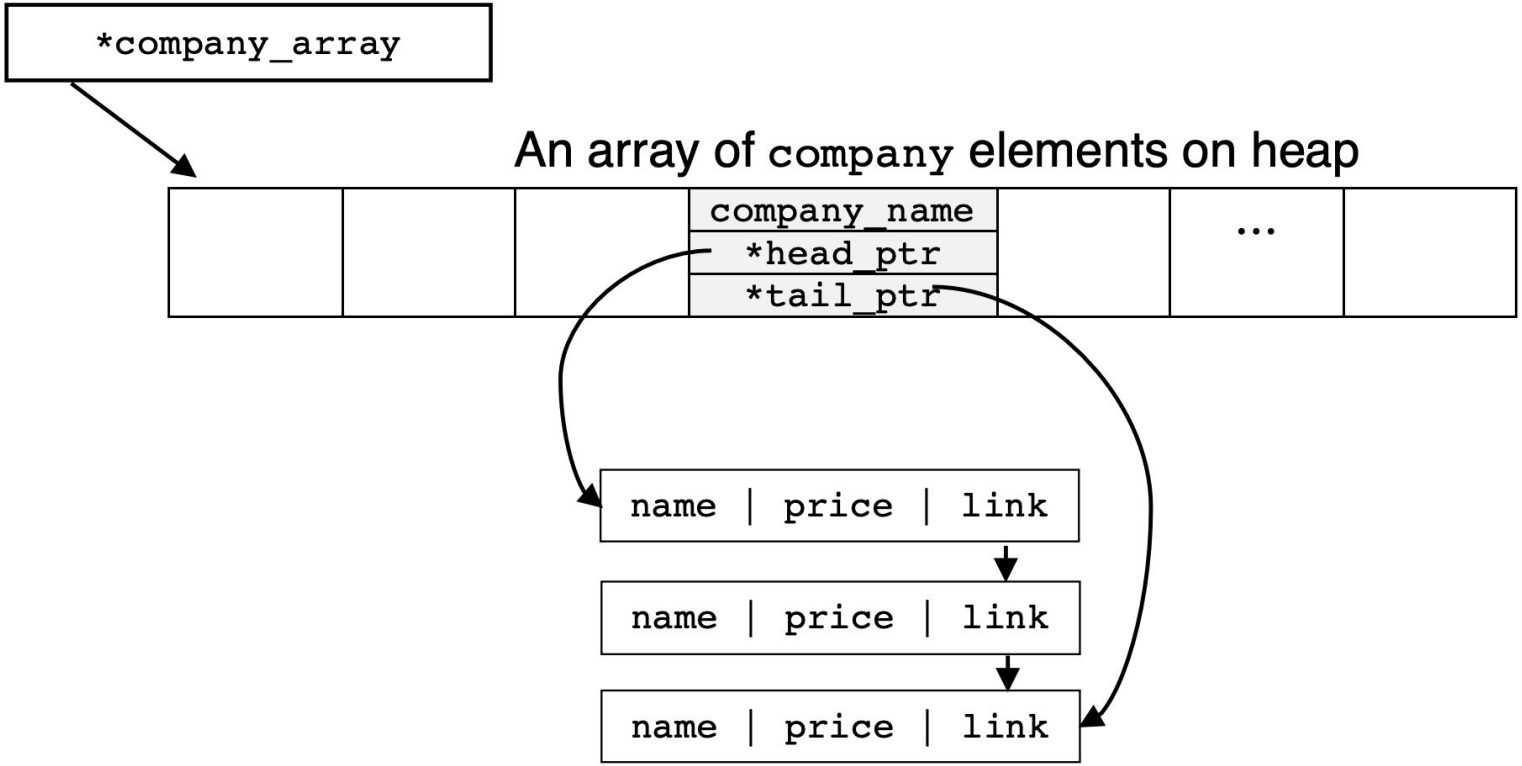
Lab 7

Week of 2/15

Partnering Up

- Each partner does the **same** provided lab for the week
  - **Not** working together, both partners need to **individually** complete and demo the lab
- Still need to write test cases and report for partner like normal


*company_array
An array of company elements on heap
company_name
*head_ptr
*tail_ptr
...
name | price | link
name | price | link
name | price | link

Database

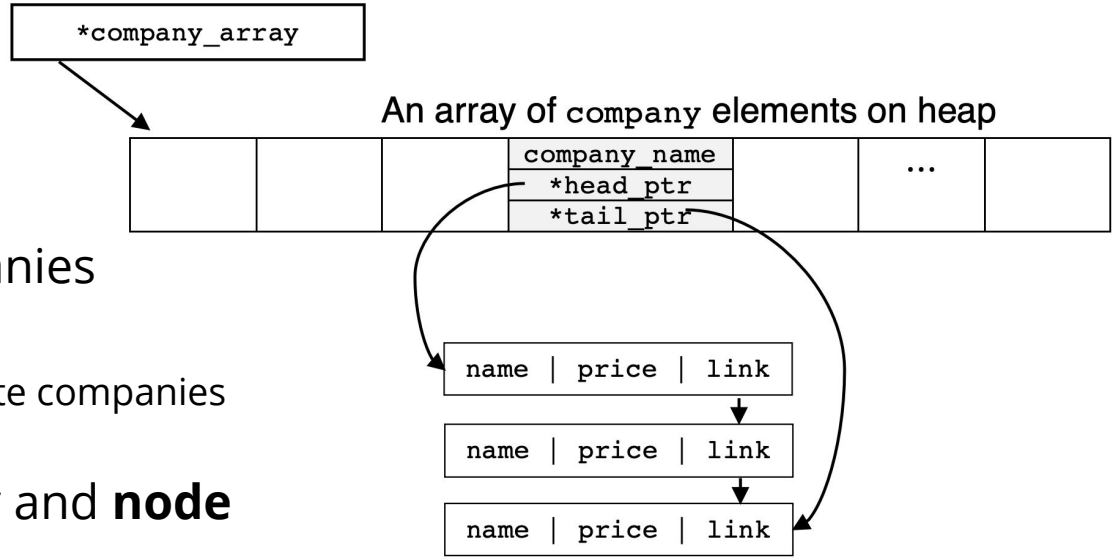
Node

- One node of a linked list
  - One product and link to next node
- Once again their are toolkit functions built into the implementation of this class that can make your **company** easier
- Private variables
  - std::string name; // name of product
  - float price; // price of product
  - node *link; // link to next product
- Functionality similar to **node** class from last week

Company

- Maintains a linked list of products (**node** class)
  - Not allowed to have duplicate products in a company
- Forward linked list
  - Order does NOT matter this week
- Private variables
  - std::string company_name; // name of company
  - node *head_ptr; // head of linked list of products
  - node *tail_ptr; // tail of linked list of products
- Functionality similar to **sequence** class from last week

Database



- Stores the name of companies as well as their products
  - Not allowed to have duplicate companies in a database
- Builds off of the **company** and **node** classes
- Private variables
  - company *company_array;
  - size_t aloc_slots;
  - size_t used_slots;
- Functionality is similar to **myString/poly** classes from a couple weeks ago

Help/Notes

- Order
  - Start with node
    - Refer back to **node** of last week (similar class)
  - Company next
    - Refer back to **sequence** from last week (similar class)
  - Database last
    - Refer back to **myString/poly** from two weeks ago (similar class)
- Add this line of code in database = operator to match output file
  - std::cout << "copying elements of database..." ;
- Remember that order of products does not matter
  - Can be helpful with *erase* functionality

Provided Files

- Node
  - node.h
  - node.cpp
- Company
  - company.h
  - company.cpp
- Database
  - database.h
  - database.cpp
- main.cpp and expected_output.txt
  - For testing
- Makefile
- All .h files implemented for you and many functions within .cpp files as well

Compile/Demo

- All files together this week (option 1)
  - g++ main.cpp database.cpp company.cpp node.cpp
  - ./a.out > output.txt
  - diff output.txt expected_output.txt
- Using Makefile (option 2)
  - make
  - ./database_test > output.txt
  - diff output.txt expected_output.txt

Don’t forget

- Demo code to me
  - Either today or next week
  - **Must compile and run on linux servers**
- Submit code to camino by the end of next lab
- Comment code
  - Loops and conditionals
- File with description of lab is on Camino
- Check google sheet to make sure that I didn’t forget to check you off for a demo