2/13/19

For project:

* Free string first, then free the node
* Make the test.txt
* Make the readme.txt file
* Tar the file and submit that bitch

Header files:

* Preprocessing comes before compilation
  + Allow user to signal to the compiler what you want to have done for preprocessing (‘#’ is the symbol for this, like when we see #include)
    - Appends all of these .h files into the corresponding c file
* Useful for modularity and clean code
  + Good practice to include the following in headers:
    - Import structs, typedefs, etc…
    - Function prototypes <return type> function\_name(<args>);
      * Defines the functions that the corresponding c file provides
      * Any time the header is included, so are the functions
      * With these, you can call any function, anywhere
* Once only headers
  + Compiler might see 2 defs of the same struct and throw an error
  + #ifndef HEADER\_H (header in all caps is for the header name)

#def HEADER\_H

…

#endif

* If I only have prototypes in my header, how does compiler know where to find implementation?
  + 4 stages of compilation
    - Preprocessing
    - Compile
    - Linking
    - Assembler
* Macros
  + #define 🡪 you can define a substitution to take place
    - #define SQUARE(X) x\*x
      * Will make any SQUARE(X) == x\*x

Version Control:

* Group project next week
* Structured way to manage a code base
* Git (GitHub)
  + Repository: project (code base) that is managed by version control
  + Commit: A unit of changes to the repository
  + Branch: a pointer to the head of a series of commits
    - Enable us work on features independent of master branch
    - Ex: Asst0
      * Master branch with working code
      * Parsing and sorting branches (for our project)
* Git: the actual software responsible for version control
* GitHub: is a company that uses git, but allows me to host repositories in the cloud