Final Exam Notes:

linked lists:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <iostream>

struct P

{

char data;

struct P\* next;

};

typedef struct P \_p;

typedef struct P\* ptr;

//initialize random seed

srand((unsigned)time(NULL));

ptr CreateNodes(const int amountOfNodes)

{

ptr head;

ptr tail;

tail = head = (ptr)malloc(sizeof(\_p));

head->data = (char)(rand() % 26 + 97);//generates a random lower case letter from using the ascii table

head->next = NULL;

for(int i = 0; i < amountOfNodes - 1; ++i)

{

tail->next = (ptr)malloc(sizeof(\_p));

tail = tail->next;

tail->next = NULL;

tail->data = (char)(rand() % 26 + 97);//generates a random lower case letter from using the ascii table

}

return head;

}

Files:

* **Creating a File**
  + **FILE \*cfPtr;**
    - **Creates a FILE pointer called cfPtr**
  + **cfPtr = fopen(“clients.dat", “w”);**
    - **Function fopen returns a FILE pointer to file specified**
    - **Takes two arguments – file to open and file open mode**
    - **If open fails, NULL returned**
  + **fprintf** 
    - **Used to print to a file**
    - **Like printf, except first argument is a FILE pointer (pointer to the file you want to print in)**
  + **feof( *FILE pointer* )**
    - **Returns true if end-of-file indicator (no more data to process) is set for the specified file**
  + **fclose( *FILE pointer* )**
    - **Closes specified file**
    - **Performed automatically when program ends**
    - **Good practice to close files explicitly**
* FILE \*cfPtr;//cfPtr = clients.dat file pointer
* /fopen opens file. Exit program if unable to create file
* if((cfPtr = fopen("clients.dat", "w")) == NULL)
* {
* printf("File could not be opened\n");
* }//end if
* else
* {
* //do stuff
* }

Recursion:

void doll ( int size )

{

if ( size == 0 ) // No doll can be smaller than 1 atom (10^0==1) so doesn't call itself

return; // Return does not have to return something, it can be used

// to exit a function

doll ( size - 1 ); // Decrements the size variable so the next doll will be smaller.

}

int main()

{

doll ( 10 ); //Starts off with a large doll (it's a logarithmic scale)

}