null pointers - New & delete - allocate memory
- pointer arithmetic
- data members, member functions
- constructors, destructors
- structs
- classes

if, for, functions

for the Fung

Bring pi-day

Close book
1. Language Fundamentals  ** Assume there is a class called: Person

   a. What does it mean to create an object of a class? Show an example

   

   data type Person me;

   b. What is the purpose of the constructor? Write the prototype:

   Initialize data members in a class

   Person();  // default constructor

   c. What is the purpose of a destructor? Write the prototype:

   close down - do anything at the end of an object lifetime
   ~Person();  // deallocate all dynamic memory
d. Show the prototype for a function where you want to pass an object of type Person by Reference and return nothing:

```c
void function (Person &);
```

e. Show the prototype for a function where you want to pass an array of objects of type Person and return an integer:

```c
int function (Person[ ], int);
```

f. Why would we want to pass a pointer (such as a head pointer) by reference? Show the prototype of a function that does this:

```c
void function (node * & head);
```
g. Create a struct called "node" that has in it a "person" object and a pointer to another node:

2. **Fundamental Linked List Questions**
   Assume there is a node struct **

a. Create a temporary pointer to a node
   \[
   \text{node} \ + \  \text{temp} \]

b. Show how to allocate memory for 1 node (you do not need to store the data yet)
   \[
   \text{temp} = \text{new node} \]
d. Show how to deallocate this one node

```cpp
delete temp;
```

e. Now, how can you tell if the list is empty?

```cpp
if (!head)
    or if (head == NULL)
```

f. Show how to insert this newly allocated node into the beginning of a non-empty linear linked list:

```cpp
temp->next = head;
head = temp;
```
3. Pointers

a. Create a pointer variable that can point to a character

```
char * ptr;
```

b. Show how to allocate memory for an array large enough to hold your favorite movie title

```
ptr = new char[strlen("MATTY")] + 1;
```

c. Show how to store your favorite movie title into this memory, keeping it simple:

```
strcpy(ptr, "MATTY");
```
d. Alternatively, what if you wanted to read in a movie title from the user – assume your title is something like: “The Blind Side”

```cpp
    char temp[100];
    cout << " Please enter your movie title: 
    cin.get(temp, 100); cin.ignore(100, vi);
    ptr = new char [strlen(temp) + 1];
    strcpy(ptr, temp);
```

e. Show how to capitalize the very first letter using pointer arithmetic

```cpp
    *ptr = toupper(*ptr);
```

f. Show how to count the number of alphabetic letters in the name (a-z or A-Z) using pointer arithmetic

```cpp
    if((variable >= 'a' && variable <= 'z') || (variable >= 'A' && variable <= 'Z'))
```
bool isitalpha (char letter)
{
    if ((letter >= 'a' && letter <= 'z') ||
        (letter >= 'A' && letter <= 'Z'))
        return true;
    return false;
}

len = strlen(away); counter = 0;
// count # alphabetic in an array
for (int i = 0; i < len; ++i)
    if (isitalpha(away[i]))
        ++counter;

for (int i = 0; away[i] != '0'; ++i)
    if (isitalpha(away[i]))
        ++counter;

// Pointer arithmatic
for (int i = 0; *(away+i)++; ++i)
    if (isitalpha(*(away+i)))
        ++counter;
```c
char * ptr;
ptr = array;

for (counta = 0; *ptr; ++ptr)
    if (is it alpha(*ptr))
        ++counta;
```
4. C++ Coding Questions. **MAKE NO ASSUMPTIONS**

Assume that you have a linear linked list of just integers

a. Write the code to remove the last node from an existing linear linked list.
b. Write the code to display the last two integers in a linear linked list.

c. Write the code to delete all nodes in a LLL.
```
class list { // maintains a linked list

public:
    list();
    ~list();
    void read();
    void display();
    void add(char name[20], char phone[13]);

private:
    node *head;
};

2b. Write the code for the constructor:
    head = NULL;

2c. Write the code for the destructor:
    list();
    {
        node *temp;
        while (head)
            {
                temp = head->next;
                delete [] head->name;
                delete head;
            }
        head = temp;
    }
```