Assignment 2

Jiwon Park

October 21, 2021

1. Write code to find the nth to last element of a singly linked list.

```
class LinkedListNode {
    LinkedListNode* next = NULL;
    int data;

    LinkedListNode(int d) { data = d; }

    void appendToTail(int d) {
        LinkedListNode* end = new LinkedListNode(d);
        LinkedListNode* n = this;
        while (n->next != NULL) { n = n->next; }
        n->next = end;
    }
};

LinkedListNode* nthToLast(LinkedListNode* head, int n) {









}
```

2. Let $p(n)$ and $q(n)$ be two nonnegative functions.
$p(n)$ is asymptotically bigger ($p(n)$ asymptotically dominates $q(n)$) than the function $q(n)$

$$\iff \lim_{n\to\infty} \frac{q(n)}{p(n)} = 0 \tag{1}$$

$q(n)$ is asymptotically smaller than $p(n)$ iff $p(n)$ is asymptotically bigger than $q(n)$. $p(n)$ and $q(n)$ are asymptotically equal iff neither is asymptotically bigger than the other. Using the above, show the following $p(n)$ is asymptotically bigger than $q(n)$:

$$p(n) = 6n^{1.5} + 12;\ q(n) = 100n \tag{2}$$

proof.

3. (Chapter 5) Write the method `arrayList<T>::pop_back`, which erases the element at the right end of the list. Do not use the erase method. What is the time complexity of your method.

```
template<class T>
class arrayList: public linearList<T> {
public:
    // constructor, copy constructor and destructor
    arrayList(int initialCapacity = 10);
    arrayList(const arrayList<T>&);
    *arrayList() {delete [] element;}

    // ADT methods
    bool empty() const {return listSize == 0;}
    int size() const {return listSize;}
    T& get(int theIndex) const;
    int indexOf(const T& theElement) const;
    void erase(int theIndex);
    void insert(int theIndex, const T& theElement);
    void output(ostream& out) const;

    // additional method
    int capacity() const {return arrayLength;
protected:
    void checkIndex(int theIndex) const;
        // throw illegalIndex if the Index invalid
    T* element;         // 1D array to hold list elements
    int arrayLength;    // capacity of the 1D array
    int listSize;       // number of elements in list
};

template<class T>
const arrayList<T>::pop_back()
{









};
```