

Start of Report

Files in directory followed by a normal run of the program followed by a valgrind run of the program. Random.txt only has like 100 words.

[illegible]

```
hermod.ics.uci.edu - PuTTY
UnorderedArrayList& operator = (const UnorderedArrayList& a) = delete;

bool isFull() //T(N) = 3; O(1)
{
    if(size == capacity)//1
    {
        return 1;//1
    }
    return 0;//1
}

bool isEmpty() //T(N) = 3; O(1)
{
    if(size == 0)//1
    {
        return 1;//1
    }
    return 0;//1
}

void insert(string word)//T(N) = 1002; O(1)
{
    buf[size] = word;//1 + 1000(because of new?)
    ++size;//1
}

bool find(string word)//T(N) = 5N + 2; O(N)
{
    for(int i = 0; i < size; ++i)//1 + N + N
    {
        if(buf[i] == word)//N + N
        {
            return 1;//N
        }
    }
    return 0;//1
}

void remove(string word)//T(N) = 6N + 1; O(N)
{
    for(int i = 0; i < size; ++i)//1 + N + N
    {
        if(buf[i] == word)//N + N
        {
            buf[i] = "";//N
            --size;//N
        }
    }
}

void print(ostream& out)//T(N) = 3N + 1; O(N)
{
    for(int i = 0; i < size; ++i)//1 + N + N
    {
        out << buf[i] << endl;//N
    }
}

~UnorderedArrayList()//T(N) = 1000; O(1)
{
    delete[] buf;//1000
};
```

84,1 Bot

Here is $O(N)$ for the ArrayList methods. I wrote the $T(N)$ to help me but $O(N)$ is followed by that on the same line.

```
hermod.ics.uci.edu - PuTTY

static void print(ostream& out, ListNode* L)//T(N) = 1000; O(1)
{
    if(L)
    {
        out << L->info << endl;//1000
    }
}

static ListNode* insert(string s, ListNode* L)//T(N) = 1001; O(1)
{
    return new ListNode(s, L);//1 + 1000
}

static ListNode* find(string s, ListNode* L)//T(N) = 4N + 2; O(N)
{
    for(ListNode* p = L; p!=nullptr; p = p->next)//1 + N + N
    {
        //ListNode::print(cout, p);
        if(p->info == s)//N
        {
            return p;//N
        }
    }
    return nullptr;//1
}

static ListNode* remove(string s, ListNode* L)//T(N) = 6N + 10; O(N)
{
    ListNode* p = L;//1
    if(p == nullptr)//1
    {
        return nullptr;//throw error 1
    }
    if(p->next == nullptr)//1
    {
        if(p->info == s)//1
        {
            p = p->next;//1
            return nullptr;//1
        }
        return nullptr;//1
    }
    ListNode* prev = p;//1
    while(p != nullptr)//N
    {
        p = p->next;//N
        if(p->info == s)//N
        {
            prev->next = p->next;//N
            return p;//N
        }
        prev = prev->next;//N
    }
    return nullptr;//1
}

};
ListNode* head;

public:
    UnorderedLinkedList()
```

76, 2-9

17%

Here is the $O(N)$ for the static functions in the Linked List class. I wrote the $T(N)$ to help me but $O(N)$ is followed by that on the same line.

```
hermod.ics.uci.edu - PuTTY

bool isFull()//T(N) = 1; O(1)
{
    return 0;//1
}

bool isEmpty()//T(N) = 3; O(1)
{
    if(head == nullptr)//1
    {
        return 1;//1
    }
    return 0;//1
}

void insert(string word)//T(N) = 1002; O(1)
{
    head = ListNode::insert(word, head);//1 + 1001
}

bool find(string word)//T(N) = 4N + 5; O(N)
{
    ListNode* temp;
    temp = ListNode::find(word, head);//1 + 4N + 2
    if(temp)
    {
        return 1;//1
    }
    return 0;//1
}

void remove(string word)//T(N) = 6N + 1012; O(N)
{
    ListNode* temp;
    temp = ListNode::remove(word, head);//1

    if(temp != nullptr)//1
    {
        delete temp;//1000
    }
}

void print(ostream& out)//T(N) = 3N + 1001; O(N)
{
    for(ListNode* p = head; p != nullptr; p = p->next)//1 + N + N
    {
        out << p->info << endl;//1000 + N
    }
}

~UnorderedLinkedList()//T(N) = 1003N; O(N)
{
    while(head != nullptr)//N
    {
        ListNode* next = head->next;//N
        delete head;//1000N
        head = next;//N
    }
}

};
```

88,0-1 Bot

Here are the methods for the Linked List class. I wrote the T(N) to help me but O(N) is followed by that on the same line.

```
hermod.ics.uci.edu - PuTTY
void insert_all_words(string file_name, UnorderedArrayList& L)//O(N)
{
    /*
        declare time object
        open file
        start timer
        for each word, w, in file
            L.insert(w);
        stop time
        close file
        report time
    */
    Timer t;
    double eTime;
    ifstream file;
    file.open(file_name);
    t.start();

    string word;
    while(file >> word)
    {
        L.insert(word);//1002N
        cout << word << endl;
    }
    t.elapsedUserTime(eTime);
    cout << eTime << endl;
    file.close();
}

void find_all_words(string file_name, UnorderedArrayList& L)//O(N^2)
{
    Timer t;
    double eTime;
    ifstream file;
    file.open(file_name);
    t.start();

    string word;
    while(file >> word)
    {
        //cout << word << endl;
        L.find(word);//5N^2 + 2N
    }
    t.elapsedUserTime(eTime);
    cout << eTime << endl;
    file.close();
}

void remove_all_words(string file_name, UnorderedArrayList& L)//O(N^2)
{
    Timer t;
    double eTime;
    ifstream file;
    file.open(file_name);
    t.start();

    string word;
    while(file >> word)
    {
        L.remove(word);//6N^2 + N
    }
    t.elapsedUserTime(eTime);
    cout << eTime << endl;
}
```

8,30

7%

Here is the $O(N)$ for the test functions of Array List.

```

}

void insert_all_words(string file_name, UnorderedLinkedList& L)//O(N)
{
    Timer t;
    double eTime;
    ifstream file;
    file.open(file_name);
    t.start();

    string word;
    while(file >> word)
    {
        L.insert(word);//1002N
    }

    t.elapsedUserTime(eTime);
    cout << eTime << endl;
    file.close();
}

void find_all_words(string file_name, UnorderedLinkedList& L)//O(N^2)
{
    /**
    Timer t;
    double eTime;
    ifstream file;
    file.open(file_name);
    t.start();

    string word;
    while(file >> word)
    {
        //cout << L.find(word) << endl;
        L.find(word);//4N^2 + 5N
    }

    t.elapsedUserTime(eTime);
    cout << eTime << endl;
    file.close();
    **/
}

void remove_all_words(string file_name, UnorderedLinkedList& L)//O(N^2)
{
    Timer t;
    double eTime;
    ifstream file;
    file.open(file_name);
    t.start();

    string word;
    while(file >> word)
    {
        L.remove(word);//6N^2 + 1012N
    }

    t.elapsedUserTime(eTime);
    cout << eTime << endl;
    file.close();
}

```

Here is the $O(N)$ for the test functions of Array List.

End of Report