**Lab 02**

**Name:** Will Townsend

**Class:** COSC 320 – Section 751

**Date:** 09/16/2020

**Lab Report:**

This lab was another relatively straightforward one. The implementation of search algorithms have always come easier to me than sorting algorithms. This lab took me about 30 minutes to code, 30 minutes to debug, and 45 minutes to put together this lab report. I can confirm and assure you that every part of this lab was done wholly, correctly, and done 100% independently. I was a bit puzzled by the empirical result equation because it always returns 1.00 which is constant. This matches the best case, but not the worst case.

**Pre-lab:**

The linear search algorithm does exactly what the name suggests. It searches through an array linearly. This makes the best case O(1) and the worst and average cases O(n).

The binary search algorithm works by using divide and conquer. It works by looking at the middle value of an array and splitting it up depending on whether that value is greater than the middle value or less. Since it is already sorted, the system already rules out all the values above or below that middle value. It then makes the lower index or upper index the middle index plus one (lower) or minus one (upper). The complexity is best case O(1); worst and average cases O(log n).

**Lab:**

**binSearch.h:**

#include<stdio.h>

template <typename any>

bool binSearch(any\* A,int begin, int end,any val){

if(end>-1&&begin>-1&&end>=begin){

int count=0;

while(begin<=end){

int mid=begin+(end-begin)/2;

if(A[mid]==val){

count++;

printf("Found value after %d iterations.\n",count);

return true;

}

if(A[mid]>val)

end=mid-1;

else

begin=mid+1;

count++;

}

printf("Value not found after %d iterations.\n",count);

return false;

}

}

**Updated deSelsort.h:**

#include<stdio.h>

#include<iostream>

template <typename any>

void swap(any\* A, int val1, int val2){

int tmp=A[val1];

A[val1]=A[val2];

A[val2]=tmp;

}

template <typename any>

void printArr(any\* A,int len){

for(int i=0;i<len;i++)

std::cout<<A[i]<<" ";

puts("");

}

template <typename any>

bool isSorted(any\* A, int len){

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

if(A[i]>A[i+1])

return false;

}

return true;

}

template <typename any>

void deSelsort(any\* A,int len){

int min=0;

int max=0;

for(int i=0;i<len/2;i++){

int end=len-i-1;

min=i;

max=end;

for(int j=i;j<end+1;j++){

if(A[j]<A[min])

min=j;

}

swap(A,i,min);

for(int j=i;j<end+1;j++){

if(A[j]>A[max])

max=j;

}

swap(A,end,max);

if(len<30){

printf("Rotation %d:\n",i+1);

printArr(A,len);

}

}

if(isSorted(A,len))

puts("Sorted Sucessfully!");

else

puts("Failed to Sort!");

}

**lab02.cpp:**

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

#include<math.h>

#include"binSearch.h"

#include"deSelsort.h"

int main(){

srand(time(0));

int sumFailCom=0;

int successTotal=0;

const int ARRSIZE=10000;

const int RANDOMLIMIT=99999;

const int RANDOMVALUES=10000;

int arr[ARRSIZE];

for (int i=0;i<ARRSIZE;i++)

arr[i]=rand()%RANDOMLIMIT+1;

printf("Lab 02: Search Algorithms\nArray was ");

deSelsort(arr,ARRSIZE);

for (int i=0;i<RANDOMVALUES;i++){

int val=rand()%RANDOMLIMIT+1;

if(binSearch(arr,0,ARRSIZE-1,val))

successTotal++;

else

sumFailCom++;

}

double result=sumFailCom/(RANDOMVALUES-successTotal);

printf("\n%d of %d failed searches.\n",sumFailCom,RANDOMVALUES);

printf("%d of %d successful searches.\n",successTotal,RANDOMVALUES);

printf("Empirical result for worst case: %0.2f\n",result);

puts("End of program...");

}

**Sample Outputs:**

**Output #1:**

Lab 02: Search Algorithms

Array was Sorted Successfully!

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 14 iterations.

Found value after 13 iterations.

Value not found after 14 iterations.

Found Value after 13 iterations.

Value not found after 14 iterations.

Value not found after 13 iterations.

Value not found after 14 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

[…]

Found value after 14 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Found value after 7 iterations.

9035 of 10000 failed searches.

965 of 10000 successful searches.

Empirical result for worst case: 1.00

End of program…

**Output #2:**

Lab 02: Search Algorithms

Array was Sorted Successfully!

Value not found after 14 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 14 iterations.

Value not found after 14 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

[…]

Found value after 14 iterations.

Value not found after 13 iterations.

Value not found after 14 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

Value not found after 13 iterations.

9109 of 10000 failed searches.

891 of 10000 successful searches.

Empirical result for worst case: 1.00

End of program…

**Lab Question:**

See Lab report…