EECE 7205: Introduction of Computer Engineering

Assignment 3

Jiayun Xin

NUID: 001563582

College of Engineering

Northeastern University Boston, Massachusetts

Fall, 2021

**Q1**

**Codes:**

#include <iostream>

#include <algorithm>

#include <time.h>

void swap (int \*a, int \*b) {

int temp = \*a;

\*a = \*b;

\*b = temp;

}

void printArray(int arr[], int n) {

for (int i = 0; i < n; ++i) std::cout << arr[i] << " ";

std::cout << "\n";

}

int partition\_e(int a[], int p, int r) {

int x = a[r];

int i = p-1;

for(int j = p; j < r; j++){

if(a[j] <= x){

i++;

swap(&a[i], &a[j]);

}

}

swap(&a[i+1], &a[r]);

return i+1;

}

int partition\_w (int arr[], int l, int r, int x) {

int i;

for (i=l; i<r; i++)

if (arr[i] == x)

break;

swap(&arr[i], &arr[r]);

i = l;

for (int j = l; j <= r - 1; j++)

{

if (arr[j] <= x)

{

swap(&arr[i], &arr[j]);

i++;

}

}

swap(&arr[i], &arr[r]);

return i;

}

int findMedian(int arr[], int n) {

std::sort(arr, arr+n);

return arr[n/2];

}

int Randomized\_Partition(int a[], int p, int r) {

int i = rand() % r + 1;

swap(&a[i], &a[r]);

return partition\_e(a, p, r);

}

int Randomized\_Select\_expected(int a[], int p, int r, int i) {

if(p==r)

return a[p];

int q = Randomized\_Partition(a,p,r);

int k = q-p+1;

if(i == k)

return a[q];

else if(i < k) {

return Randomized\_Select\_expected(a, p, q-1, i);

}

else {

return Randomized\_Select\_expected(a, q+1, r, i-k);

}

}

int Randomized\_Select\_worst(int arr[], int l, int r, int k) {

if (k > 0 && k <= r - l + 1) {

int n = r-l+1;

int i, median[(n+4)/5];

for (i=0; i<n/5; i++) median[i] = findMedian(arr+l+i\*5, 5);

if (i\*5 < n) {

median[i] = findMedian(arr+l+i\*5, n%5);

i++;

}

int medOfMed = (i == 1)? median[i-1]:

Randomized\_Select\_worst(median, 0, i-1, i/2);

int pos = partition\_w(arr, l, r, medOfMed);

if (pos-l == k-1)

return arr[pos];

if (pos-l > k-1)

return Randomized\_Select\_worst(arr, l, pos-1, k);

return Randomized\_Select\_worst(arr, pos+1, r, k-pos+l-1);

}

return INT\_MAX;

}

void shuffleRandom (int arr[], int n ) {

srand ( time(NULL) );

for (int i = n-1; i > 0; i--) {

int j = rand() % (i+1);

swap(&arr[i], &arr[j]);

}

}

int main() {

int arr\_size = 100;

int arr[arr\_size];

clock\_t start, finish;

double duration;

for (int i = 0; i < arr\_size; i++) {

arr[i] = i + 1;

}

shuffleRandom (arr, arr\_size);

start = clock();

std::cout << "K'th smallest element (with linear expected running time) is "

<< Randomized\_Select\_expected(arr, 0, 99, 10) << std::endl;

finish = clock();

duration = (double)(finish - start) / CLOCKS\_PER\_SEC;

std::cout << "Running time is: " << duration << "seconds" << std::endl << std::endl;

start = clock();

std::cout << "K'th smallest element (with linear worst-case running time) is "

<< Randomized\_Select\_worst(arr, 0, 99, 10) << std::endl;

finish = clock();

duration = (double)(finish - start) / CLOCKS\_PER\_SEC;

std::cout << "Running time is: " << duration << "seconds";

}

**Results:Text

Description automatically generated**

**Q2**

**Codes:**

#include <stdio.h>

#include <string.h>

#include <iostream>

using namespace std;

int max(int a, int b) {

return (a > b)? a : b;

}

int lcs(char \*str1, char \*str2, int l1, int l2) {

if (l1 == 0 || l2 == 0)

return 0;

if (str1[l1-1] == str2[l2-1])

return 1 + lcs(str1, str2, l1-1, l2-1);

else

return max(lcs(str1, str2, l1, l2-1), lcs(str1, str2, l1-1, l2));

}

int main() {

clock\_t start, finish;

double duration;

char str1[] = "ADFHAGHTSDFGFDSGFADFADF";

char str2[] = "SAGHGFSSDFADATDGSDFGADF";

int l1 = strlen(str1);

int l2 = strlen(str2);

start = clock();

cout<<"Length of LCS is " << lcs(str1, str2, l1, l2) << endl;

finish = clock();

duration = (double)(finish - start) / CLOCKS\_PER\_SEC;

cout << "Running time is: " << duration << " seconds";

return 0;

}

**Results:**

