## CSCI 140 PA 4 Submission

## Due Date: 09/21/2021 Late (date and time):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Name(s): Nero Li

Exercise 1 -- need to submit source code and I/O  
 -- check if completely done ✔️ ; otherwise, discuss issues below  
Source code below:

/\* Program: PA\_4\_exercise\_1

Author: Nero Li

Class: CSCI 220

Date: 09/21/2021

Description:

Implement the linear Fibonacci version based on pseudocode on page 146 of

the C++ textbook.

I certify that the code below is my own work.

Exception(s): N/A

\*/

#include <iostream>

using namespace std;

struct fibPair

{

int a;

int b;

};

fibPair LinearFibonacci(int k)

{

fibPair temp;

int i, j;

if (k <= 1)

{

temp.a = k;

temp.b = 0;

}

else

{

temp = LinearFibonacci(k - 1);

i = temp.a;

j = temp.b;

temp.a = i + j;

temp.b = i;

}

return temp;

}

int main()

{

fibPair result;

for (int i = 10; i > 0; --i)

{

result = LinearFibonacci(i);

cout << "k = " << i << ": (" << result.a << ", " << result.b << ")\n";

}

cout << "Author: Nero Li\n";

return 0;

}

Input/output below:

k = 10: (55, 34)

k = 9: (34, 21)

k = 8: (21, 13)

k = 7: (13, 8)

k = 6: (8, 5)

k = 5: (5, 3)

k = 4: (3, 2)

k = 3: (2, 1)

k = 2: (1, 1)

k = 1: (1, 0)

Author: Nero Li

Exercise 2 -- need to submit source code and I/O  
 -- check if completely done ✔️ ; otherwise, discuss issues below  
Source code below:

/\* Program: PA\_4\_exercise\_2

Author: Nero Li

Class: CSCI 220

Date: 09/21/2021

Description:

Perform max sum problem.

I certify that the code below is my own work.

Exception(s): N/A

\*/

#include <iostream>

#include <ctime>

#include <chrono>

using namespace std;

long long maxSum(long long x[], long long n)

{

long long max{0};

for (long long i = 0; i < n; ++i)

{

long long cur{x[i]};

for (long long j = i + 1; j < n; ++j)

{

cur += x[j];

if (cur > max)

{

max = cur;

}

}

}

return max;

}

void runningTimeCheck(int n)

{

long long x[100000];

srand(time(NULL));

for (size\_t i = 0; i < n; ++i)

{

x[i] = rand() % 20001 - 10000;

}

auto start = chrono::high\_resolution\_clock::now();

maxSum(x, n);

auto end = chrono::high\_resolution\_clock::now();

cout << (chrono::duration\_cast<chrono::nanoseconds>(end - start).count() \* (double)1e-6) << " ms" << endl;

}

int main()

{

/\* function check \*/

long long x[10] = {31, -41, 59, 26, -53, 58, 97, -93, -23, 84};

cout << maxSum(x, 10) << endl;

/\* running time check \*/

runningTimeCheck(100);

runningTimeCheck(1000);

runningTimeCheck(10000);

runningTimeCheck(100000);

cout << "Author: Nero Li\n";

return 0;

}

Input/output below:

187

0 ms

0.9778 ms

123.049 ms

11905.5 ms

Author: Nero Li

Answer for Exercise 2 question:

The calculated runtime equation is , so my solution for Big-Oh notation will be O(n^2), and the collected time is a little bit lower than Big-Oh notation for the first two test cases, and then become higher than Big-Oh notation for the second two test cases. However, they are pretty close to the expected running time. Furthermore, if we take a look at the gap between each test case, they have almost the same increase degree as the f(n) = n^2 function. The reason for the deviation is because, for the current group of numbers, the sum process will take more time than expected since their number is too big. If we just collect the running time for loop, the output will look like this:

0 ms

0.9765 ms

99.0006 ms

9757.14 ms

Answer for Question 1:

If we are using a random number to experiment, the final runtime may not be the highest runtime due to a better random queue such as for the search algorithm, so it may not meet the Big-Oh notation.

Answer for Question 2:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 Second | 1 Hour | 1 Month | 1 Century |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Extra Credit

Source code below:

/\* Program: PA\_4\_extra\_credit

Author: Nero Li

Class: CSCI 220

Date: 09/21/2021

Description:

Perform P-4.1 from C++ textbook.

I certify that the code below is my own work.

Exception(s): N/A

\*/

#include <iostream>

#include <ctime>

#include <chrono>

using namespace std;

long long \*prefixAverages1(int x[], int n)

{

long long A[100000];

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

{

long long a = 0;

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

{

a += x[j];

}

A[i] = a / (i + 1);

}

long long \*p = A;

return p;

}

long long \*prefixAverages2(int x[], int n)

{

long long A[100000];

long long s{0};

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

{

s += x[i];

A[i] = s / (i + 1);

}

long long \*p = A;

return p;

}

void runningTimeCheck(int n)

{

int x[100000];

srand(time(NULL));

for (size\_t i = 0; i < n; ++i)

{

x[i] = rand() % 20001 - 10000;

}

auto start1 = chrono::high\_resolution\_clock::now();

prefixAverages1(x, n);

auto end1 = chrono::high\_resolution\_clock::now();

cout << (chrono::duration\_cast<chrono::nanoseconds>(end1 - start1).count() \* (double)1e-6) << " ms" << endl;

auto start2 = chrono::high\_resolution\_clock::now();

prefixAverages2(x, n);

auto end2 = chrono::high\_resolution\_clock::now();

cout << (chrono::duration\_cast<chrono::nanoseconds>(end2 - start2).count() \* (double)1e-6) << " ms" << endl;

}

int main()

{

long long \*p;

int test[] = {31, -41, 59, 26, -53, 58, 97, -93, -23, 84};

int n{10};

/\* function check \*/

p = prefixAverages1(test, n);

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

{

cout << \*(p + i) << ' ';

}

cout << endl;

p = prefixAverages2(test, n);

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

{

cout << \*(p + i) << ' ';

}

cout << endl;

/\* running time check \*/

runningTimeCheck(100);

runningTimeCheck(1000);

runningTimeCheck(10000);

runningTimeCheck(100000);

cout << "Author: Nero Li\n";

return 0;

}

Input/output below:  
31 -5 16 18 4 13 25 10 6 14

31 -5 16 18 4 13 25 10 6 14

0 ms

0 ms

0.977 ms

0 ms

110.861 ms

0 ms

11030.1 ms

0.9773 ms

Author: Nero Li