Khang Nguyen Student ID: 300007277 CSI2110 Assignment #1

1.

a)

By definition of Big-O, a real constant and a constant integer (becaus) does not exist when since the input for log would be negative) is needed such that for

Because it is safe to say that for . It is also safe to say for .

Thus

One solution is and for which is **true**. Thus is

b)

By definition of Big-Omega, a real constant and a constant integer (because ) does not exist when since the input for log would be negative) is needed such that for

Because it is safe to say that for .

Thus

in this case and would make it false

Because it is safe to say that the inequality is **false** for all where and . Thus is not

c)

By definition of Big-O, a real constant and a constant integer is needed such that for

Keeping in mind that

Because it is safe to say that the inequality is **false** for all where and . Thus is not

d)

By definition of Big-Theta, two real constant , and a constant integer is needed such that and for

Keeping in mind that

Because it is safe to say that the inequality is **false** for all where and . Thus is not and by consequence not

e)

By definition of Big-O, a real constant and a constant integer is needed such that for

Keeping in mind that and that then it is understandable that

Because it is safe to say that the inequality is **false** for all where and . Thus is not

f)

By definition of Big-Omega, a real constant and a constant integer is needed such that for

Keeping in mind that and that then it is understandable that

Because it is safe to say that the inequality is true

One solution is and for which is **true**. Thus is

g)

By definition of Big-O, a real constant and a constant integer is needed such that for

Because it is safe to say that

Because it is safe to say that the inequality is true

One solution is and for which is **true**. Thus is and

h)

By definition of Big-Omega, a real constant and a constant integer is needed such that for

Because it is safe to say that

Because it is safe to say that the inequality is **false** for all where and . Thus is not

2.

a)

To start off, bit shifts work in binary, thus y must be changed into binary. Every time a bit is shifted to the right it is divided by 2, and every time it is shifted to the left, it is multiplied by 2. Thus the number of y should be presented in thus . Since running times would leave b at 1, we must run it an extra time, to add there is 2 bit shift per loop(8 and 9). Thus the number of bit shifts would be

b)

By definition of Big-O, a real constant and a constant integer is needed such that for

Keeping in mind that it is safe to say that

Thus

One solution is and for which is **true**. Thus is

3.

What we know: old machine would take 1 second for size N dataset. Thus for N. The new machine is 10 times faster. Thus for N

a)

where p is a coefficient of n. We know it would take 1 second. Thus and . To continue for N. Thus . By consequence for we can see that

b)

where p is a coefficient of n. We know it would take 1 second. Thus and . To continue for N. Thus . By consequence for we can see that

c)

where p is a coefficient of n. We know it would take 1 second. Thus and . To continue for N. Thus . By consequence for we can see that and thus

4.

a)

i) In this case the array only stores 0 or 1 thus it is running the loop 2N times (N for 0 and N for 1) because it is running it 2N times which can be written as where c is 2. Thus the function is

ii)In this case the array stores N or near N elements thus it is running the loop N\*N times (once for every element) because it is running it N\*N times which can be written as where c is 1. Thus the function is

b)

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

H[i] = 0;

}

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

H[A[i]] ++;

}

The first loop will create an array that contains all of the elements from 0 to n. The second loop will then run through all of array A’s elements (in this case N as said in the question). Using the value of A[i] as index for H will make it so for example all integers 1 will be in the same array of H

For the first case the array only stores 0 or 1 thus it is running the loop 2 times for 0 and 1 and N times for the element list. Thus 2+N times, because it is running it N+2 times which can be written as where c is 1. Thus the function is

For the second case the array stores N different values and N elements, thus it is running the loop 2N times (N for values and N for elements) because it is running it 2N times which can be written as where c is 2. Thus the function is