CSci 243 Homework 7

Due: Wednesday, November 2, end of day

My name

1. (10 points) What is the largest problem size n that we can solve in no more than **one hour** using an algorithm that requires $f(n)$ operations, where each operation takes 10^{-9} seconds (this is close to a today's computer), with the following $f(n)$?

   (a) $\log_2 n$
   (b) $\log_2^4 n$
   (c) $3n$
   (d) $n\log_2 n$
   (e) $n\log_2^2 n$
   (f) n^2
   (g) $(3n)^3$
   (h) 2^n
   (i) $n!$
   (j) n^n

   10^{-9} seconds

   $10^{-9} \times 3600 =$ ops

2. (10 points) Use pseudocode to describe an algorithm that determines whether a given function from a finite set to another finite set is one-to-one. Assume that the function $f : A \to B$ is given as a set of pairs $\{(a_i, f(a_i)), \forall a_i \in A = \{a_1, \ldots, a_m\}$. You may also assume that you are given $B = \{b_1, \ldots, b_n\}$.

   check one-to-one compare the sets

3. (5 points) Describe an algorithm that produces the maximum, the minimum, and the mean, of a set of n real numbers, passing through the numbers once. $O(n)$ I guess ez?

4. (10 points) Consider the following pseudocode that finds x in a list of sorted numbers by using ternary search. The algorithm is simlar to binary search, only it splits the current list into three parts (instead of two) and checks which part x may be in. Thus at each step, the algorithm removes 2/3 of the items in the current list. Find the complexity of the algorithm. Is it faster or slower asymptotically than binary search? = faster than $O(\log n)$? It would have to be like constant time or some ugly nasty log log thing

```
Algorithm TernarySearch(x int, a(1)...a(n) int, output: loc int)
i=1;
j=n;
remaining = j-i+1;
while (remaining > 1)
   interval = floor(remaining/3);
   m1 = i+interval;
   m2 = i+2*interval;
   if (x <= a(m1))
      j=m1;
   else if (x<=a(m2))
      i=m1+1;
      j=m2;
   else
      i=m2+1;
   endelsif
   remaining = j-i+1;
endwhile

if (remaining == 1 and x not equal a(i)) return loc=0;
else return loc = j;
```

n=36 → 18 18 → 9 9 → 5 4 → 3 2 → 2 1 → 1

n=36 number is like → 12 12 12 → 4 4 4 → 1 2 1 → 1 1 – → done

I mean idk

$\frac{2}{3}\log_3(n+1)$ $(\log_3(n+1)$ according to BIG or $\log_3 n$ $2\log_3 n$ " $\frac{2}{\log_3 3}\log_3 n$ $\frac{2}{\log_3 3} > 1$ so binary is better

$\frac{4}{3}\log_3(n+1)$ $\log_2(n+1)$ " ternary is 79% slower" according to Google

where is there another common base anyways?

CSci 243 Homework 7

Due: Wednesday, November 2, end of day
My name

1. (10 points) What is the largest problem size n that we can solve in no more than **one hour** using an algorithm that requires $f(n)$ operations, where each operation takes 10^{-9} seconds (this is close to a today's computer), with the following $f(n)$?

   (a) $\log_2 n$
   (b) $\log_2^4 n$
   (c) $3n$
   (d) $n\log_2 n$
   (e) $n\log_2^2 n$
   (f) n^2
   (g) $(3n)^3$
   (h) 2^n
   (i) $n!$
   (j) n^n

2. (10 points) Use pseudocode to describe an algorithm that determines whether a given function from a finite set to another finite set is one-to-one. Assume that the function $f : A \to B$ is given as a set of pairs $\{(a_i, f(a_i)), \forall a_i \in A = \{a_1, \ldots, a_m\}$. You may also assume that you are given $B = \{b_1, \ldots, b_n\}$.

3. (5 points) Describe an algorithm that produces the maximum, the minimum, and the mean, of a set of n real numbers, passing through the numbers once.

4. (10 points) Consider the following pseudocode that finds x in a list of sorted numbers by using ternary search. The algorithm is simlar to binary search, only it splits the current list into three parts (instead of two) and checks which part x may be in. Thus at each step, the algorithm removes 2/3 of the items in the current list. Find the complexity of the algorithm. Is it faster or slower asymptotically than binary search?

```
Algorithm TernarySearch(x int, a(1)...a(n) int, output: loc int)
i=1;
j=n;
remaining = j-i+1;
while (remaining > 1)
   interval = floor(remaining/3);
   m1 = i+interval;
   m2 = i+2*interval;
   if (x <= a(m1))
      j=m1;
   else if (x<=a(m2))
      i=m1+1;
      j=m2;
   else
      i=m2+1;
   endelsif
   remaining = j-i+1;
endwhile

if (remaining == 1 and x not equal a(i)) return loc=0;
else return loc = j;
```

```
max_min_mean (array)
max = lowest_value
min = most_value
sum = 0
for i in array.length {
    k = array[i]
    if k < min
        min = k
    if k > max
        max = k
    sum = sum + k
}
mean = sum / array.length
return max, min, mean
```

CSci 243 Homework 7

Due: Wednesday, November 2, end of day

My name

1. (10 points) What is the largest problem size n that we can solve in no more than **one hour** using an algorithm that requires $f(n)$ operations, where each operation takes 10^{-9} seconds (this is close to a today's computer), with the following $f(n)$?

   (a) $\log_2 n$
   (b) $\log_2^4 n$
   (c) $3n$
   (d) $n\log_2 n$
   (e) $n\log_2^2 n$
   (f) n^2
   (g) $(3n)^3$
   (h) 2^n
   (i) $n!$
   (j) n^n

2. (10 points) Use pseudocode to describe an algorithm that determines whether a given function from a finite set to another finite set is one-to-one. Assume that the function $f : A \to B$ is given as a set of pairs $\{(a_i, f(a_i)), \forall a_i \in A = \{a_1, \ldots, a_m\}$. You may also assume that you are given $B = \{b_1, \ldots, b_n\}$.

3. (5 points) Describe an algorithm that produces the maximum, the minimum, and the mean, of a set of n real numbers, passing through the numbers once.

4. (10 points) Consider the following pseudocode that finds x in a list of sorted numbers by using ternary search. The algorithm is simlar to binary search, only it splits the current list into three parts (instead of two) and checks which part x may be in. Thus at each step, the algorithm removes 2/3 of the items in the current list. Find the complexity of the algorithm. Is it faster or slower asymptotically than binary search?

```
Algorithm TernarySearch(x int, a(1)...a(n) int, output: loc int)
i=1;
j=n;
remaining = j-i+1;
while (remaining > 1)
   interval = floor(remaining/3);
   m1 = i+interval;
   m2 = i+2*interval;
   if (x <= a(m1))
      j=m1;
   else if (x<=a(m2))
      i=m1+1;
      j=m2;
   else
      i=m2+1;
   endelsif
   remaining = j-i+1;
endwhile

if (remaining == 1 and x not equal a(i)) return loc=0;
else return loc = j;
```

CSci 243 Homework 7

Due: Wednesday, November 2, end of day

My name

1. (10 points) What is the largest problem size n that we can solve in no more than **one hour** using an algorithm that requires $f(n)$ operations, where each operation takes 10^{-9} seconds (this is close to a today's computer), with the following $f(n)$?

   0.000 000 001 a billionth of a second

   a billion per second

   one hour = 3600 seconds

   => 3.6 billion operations

   3,600,000,000,000

   (a) $\log_2 n$ $\log_2 n = 3.6T \Rightarrow 2^{3.6T} = n$

   (b) $\log_2^4 n$ $(\log_2 n)^4 = 3.6T \Rightarrow 2^{\sqrt[4]{3.6T}} = n$

   (c) $3n$ $3n = 3.6T \Rightarrow n = 1.2T$

   (d) $n\log_2 n$ $n\log_2 n = 3.6T \Rightarrow$

   (e) $n\log_2^2 n$ $n(\log_2 n)^2 = 3.6T \Rightarrow$

   (f) n^2 $n^2 = 3.6T \Rightarrow n = \sqrt{3.6T}$

   (g) $(3n)^3$ $(3n)^3 = 7.6T \Rightarrow n = \sqrt[3]{3.6T}/3$

   (h) 2^n $2^n = 3.6T \Rightarrow \log_2 3.6T = n$

   (i) $n!$ $n! = 3.6T \Rightarrow$

   (j) n^n $n^n = 3.6T \Rightarrow$

2. (10 points) Use pseudocode to describe an algorithm that determines whether a given function from a finite set to another finite set is one-to-one. Assume that the function $f : A \to B$ is given as a set of pairs $\{(a_i, f(a_i)), \forall a_i \in A = \{a_1, \ldots, a_m\}$. You may also assume that you are given $B = \{b_1, \ldots, b_n\}$.

3. (5 points) Describe an algorithm that produces the maximum, the minimum, and the mean, of a set of n real numbers, passing through the numbers once.

4. (10 points) Consider the following pseudocode that finds x in a list of sorted numbers by using ternary search. The algorithm is simlar to binary search, only it splits the current list into three parts (instead of two) and checks which part x may be in. Thus at each step, the algorithm removes 2/3 of the items in the current list. Find the complexity of the algorithm. Is it faster or slower asymptotically than binary search?

```
Algorithm TernarySearch(x int, a(1)...a(n) int, output: loc int)
i=1;
j=n;
remaining = j-i+1;
while (remaining > 1)
   interval = floor(remaining/3);
   m1 = i+interval;
   m2 = i+2*interval;
   if (x <= a(m1))
      j=m1;
   else if (x<=a(m2))
      i=m1+1;
      j=m2;
   else
      i=m2+1;
   endelsif
   remaining = j-i+1;
endwhile

if (remaining == 1 and x not equal a(i)) return loc=0;
else return loc = j;
```

function is Injective (f , B)

previous_values = [ ]

for each pair in f :

if f(a_i) is in B

and f(a_i) is not in previous_values :

previous_values.append (f(a_i))

elif f(a_i) is in previous_values

and f(a_i) is in B :

return false

return true

CSci 243 Homework 7

Due: Wednesday, November 2, end of day

My name

1. (10 points) What is the largest problem size n that we can solve in no more than **one hour** using an algorithm that requires $f(n)$ operations, where each operation takes 10^{-9} seconds (this is close to a today's computer), with the following $f(n)$?

   (a) $\log_2 n$
   (b) $\log_2^4 n$
   (c) $3n$
   (d) $n\log_2 n$
   (e) $n\log_2^2 n$
   (f) n^2
   (g) $(3n)^3$
   (h) 2^n
   (i) $n!$
   (j) n^n

2. (10 points) Use pseudocode to describe an algorithm that determines whether a given function from a finite set to another finite set is one-to-one. Assume that the function $f : A \to B$ is given as a set of pairs $\{(a_i, f(a_i)), \forall a_i \in A = \{a_1, \ldots, a_m\}$. You may also assume that you are given $B = \{b_1, \ldots, b_n\}$.

3. (5 points) Describe an algorithm that produces the maximum, the minimum, and the mean, of a set of n real numbers, passing through the numbers once.

4. (10 points) Consider the following pseudocode that finds x in a list of sorted numbers by using ternary search. The algorithm is simlar to binary search, only it splits the current list into three parts (instead of two) and checks which part x may be in. Thus at each step, the algorithm removes 2/3 of the items in the current list. Find the complexity of the algorithm. Is it faster or slower asymptotically than binary search?

```
Algorithm TernarySearch(x int, a(1)...a(n) int, output: loc int)
i=1;
j=n;
remaining = j-i+1;
while (remaining > 1)
   interval = floor(remaining/3);
   m1 = i+interval;
   m2 = i+2*interval;
   if (x <= a(m1))
      j=m1;
   else if (x<=a(m2))
      i=m1+1;
      j=m2;
   else
      i=m2+1;
   endelsif
   remaining = j-i+1;
endwhile

if (remaining == 1 and x not equal a(i)) return loc=0;
else return loc = j;
```

binary search worst case

$T(N) = c + T(\frac{N}{2})$

$T(\frac{N}{2}) = c + T(\frac{N}{4})$

$T(N) = c + c + T(\frac{N}{4}) = 2c + T(\frac{N}{4})$

...

$T(N) = T(\frac{N}{2^i}) + i \cdot c$

↓ stops when $\frac{N}{2^i} = 1$

At some point on $N/2^i$ branching, we reach only 1 element

$T(N/2^i) = T(1)$

$\frac{N}{2^i} = 1$

$N = 2^i$

$\log_2 N = \log_2 2^i$

$\log_2 N = i \log_2 2$

$\log_2 N = i$

$T(N) = T(\frac{N}{2^{\log_2 N}}) + c \cdot \log_2 N$

$= T(\frac{N}{N}) + c \cdot \log_2 N$

$= T(1) + c \log_2 N$

$T(N) = T(1) + c \log_2 N$

$T(1) =$ some constant time k

$T(N) = k + c \log_2 N$

$O(N) = \log_2 N$

ok can I do this for ternary search?

in binary search
by every 1 comparison reduces elements by 50%

in ternary search
by every 2 comparisons reduces elements by 66.6 %

so is the extra comparison still more efficient when you account for reducing elements by another 16.6 %?
no
how do I show that?
idk

binary $2 \times \log_2 n + \cancel{O(1)}$ ← constant time extra work

ternary $4 \times \log_3 n + \cancel{O(1)}$

so which is smaller
$2 \log_2 n$ or $4 \log_3 n$?

simplify further ...
$\log_2 n$ or $2 \log_3 n$?

$\log_2 n \quad <=> \quad 2 \log_3 n$

b/c log properties idk ↳ $2 \cdot \frac{\log(2)}{\log(3)} \cdot \log_2 n$

$\log_2 n \quad <=> \quad 2 \frac{\log 2}{\log 3} \log_2 n$

$1 \quad <=> \quad 2 \frac{\log 2}{\log 3}$

$1 \quad < \quad 2 \frac{\log 2}{\log 3}$

so binary more efficient