Algorithmn HW2

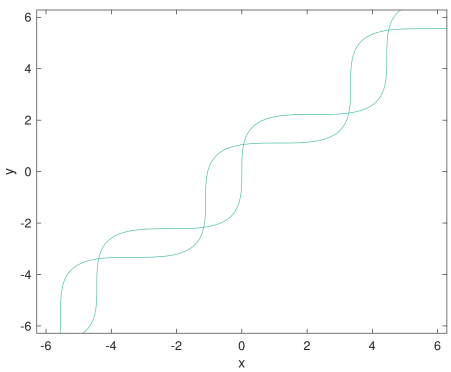
5140379032 JIN YI FAN

Problem 1.17

Construct $f(n)$ and $g(n)$ such that:

$$\frac{\sqrt{2}}{2}(f(n)-x)=\sin(\frac{\sqrt{2}}{2}(f(n)+x))$$
$$\frac{\sqrt{2}}{2}(g(n)-x)=\cos(\frac{\sqrt{2}}{2}(g(n)+x))$$

which are shown in axis below, obviously they are neither the other's upper bound.



Problem 1.34

(a) $O(n)$

$MAX \leftarrow A[0]$
$MIN \leftarrow A[0]$
for $i = 0$ to n **do**
  $MAX \leftarrow max\{MAX, A[i]\}$
  $MAX \leftarrow min\{MIX, A[i]\}$
  $i \leftarrow i + 1$
end for

(b) $\Omega(nlogn)$

QuickSort($A[]$)
$MAX \leftarrow A[n]$
$MIN \leftarrow A[0]$

Problem 1.35

Ensure: no duplicated elements in $A[]$
$a \leftarrow A[0], b \leftarrow A[1], c \leftarrow A[2]$
$B[]$=Sort(a, b, c)
return $B[1]$

Problem 1.37

Construct an array $P[]$, s.t.$P[i] = a_i$

(a) $\Omega(n^2)$

$sum \leftarrow P[0]$
for $i = 1$ to n **do**
  $exp \leftarrow 1$
  for $j = 1$ to i **do**
    $exp \leftarrow exp * x$
    $j \leftarrow j + 1$
  end for
  $sum \leftarrow sum + exp * P[i]$
  $i \leftarrow i + 1$
end for
return sum

(b) $O(n)$

$sum \leftarrow P[n]$
for $i = n - 1$ to 0 **do**
  $sum \leftarrow sum * x + P[i]$
  $i \leftarrow i - 1$
end for
return sum

Problem 2: Egg drop

Version 0:

Require: 1 egg, $\leq T$ tosses

$floor \leftarrow 1$
for $floor = 1$ to N **do**
  if $egg.drop(floor) == break$ **then**
    return $floor$
  end if
  $floor \leftarrow floor + 1$
end for
return $floor$

Version 1:

Require: $logN$ eggs, $logN$ tosses

1: $low \leftarrow 1$
2: $high \leftarrow N$
3: $floor \leftarrow 1$
4: **for** $i = 1$ to $logN$ **do**
5:   $floor \leftarrow low + \frac{high-low}{2}$
6:   **if** $egg.drop(floor) == break$ **then**
7:     **if** $high == low$ **then**
8:       **return** $high$
9:     **end if**
10:     $high \leftarrow floor$
11:   **else**
12:     **if** $high - low \leq 1$ **then**
13:       **return** $high$
14:     **end if**
15:     $low \leftarrow floor$
16:   **end if**
17: **end for**

Version 2:

Require: $logT$ eggs, $2logT$ tosses

$floor \leftarrow 1$
repeat
  $floor \leftarrow (floor * 2)$
until $egg.drop(floor) == break$
$low \leftarrow (floor/2)$
$high \leftarrow floor$
BinaryEggDrop(low, $high$) {The algorithmn starting at Version 1, line 3}

Version 3:

Require: 2 eggs, $2\sqrt{T}$ tosses

$floor \leftarrow 1$
repeat
  $floor \leftarrow (floor + \sqrt{T})$
until $eggOne.drop(floor) == break$ {At most $\sqrt{T} tosses$}
for $i \leftarrow (floor - \sqrt{T})$ to $floor$ **do**
  if $eggTwo.drop(i) == break$ **then**
    return $i - 1$
  else
    $i \leftarrow i + 1$
  end if{At most $\sqrt{T}$ tosses}
end for

Version 4:

Require: 2 eggs, $\leq c\sqrt{T}$ tosses

Don't know how to do it...
Maybe DP?