## Shreyas Belkune

Sb2660@rit.edu

## 1)

a) *F (n) = Fn+1 − 1.*

*TF (0) = 0*

*TF (1) = 0*

*Assume the below is true.*

*TF(k) = F(k+1) − 1.*

Inductive step

*TF(k+1) = 1 + TF (k) + TF (k-1)*

*TF(k+1) =1+ Fk+1 -1+ TF (k-1)*

*TF(k+1) = Fk+1 + TF (k-1)*

*TF(k+1) = Fk+1 + Fk - 1*

*=Fk+2 -1*

b)

2) *f(0; a, b) = a*

*f(1; a, b) = b*

*f(n; a, b) = f(n − 1; b, a + b)*

*TF (0) = 0*

*TF (1) = 0*

*TF (n) = 1 + TF (n − 1)*

*= 1 + 1 + TF (n − 2)*

*= 1 + 1+ 1 + TF (n − 3)*

*Put k = n,*

*TF(n) = k + TF(n-k)*

*TF(n) = k + TF(0)*

*TF (n) = n*

The time complexity of fibItHelper function is O(n).

3) For n =1,

*L(a,b) = (b,a+b)*

*i.e., (f(1;a,b),f(2;a,b)) = (b,f(1;b,a+b))*

*= (b,a+b)*

*Assume true for n =k,*

*Lk(a,b) = (f(k;a,b) , (f(k+1;a,b))*

*So for n = k+1,*

*Substituting k+1 we get,*

*L(k+1) = L(Lk(a,b))*

*= L(f(k;a,b) , f(k+1;a,b))*

*= (f(k+1;a,b) , f(k;a,b) + f(k+1;a+b))*

*= (f(k+1;a,b) , f(k+2;a,b))*

For any n we can prove ∈ N, Ln(a, b) = (f(n; a, b), f(n + 1; a, b)).

## 5)

a. False, Fib is not a pseudo-polynomial. This is because time complexity grows exponentially with increasing input value. For instance, the amount of time needed to calculate the recursive functions fib(n-1) and fib(n-2) will be same.

b. True, Yes, fibIt is pseudo-polynomial. This is due to the fact that, in the worst situation, computing the nth number in the Fibonacci series will require O(n).

c. True, FibPow is indeed a pseudo-polynomial. In the worst case, it will take O(nlog(n)) because each time fibPow calls itself, the value of n is half.

## 6)

a) *T(n+1) = T(n) + 5*

*T(0) = 5*

*T(1) = T(0) + 5*

*= 5*

*T(n) = T(n-1) + 5*

*= T(n-2) + 5 + 5*

*= T(n-3) + 5 + 5 + 5*

*= T(n-4) + (5\*4)*

*Using k*

*= T(n-k) + (5\*k)*

*Putting k in n*

*n = k*

*= T(k – k) + (5\*k)*

*= T(0) + 5\*k*

*= 5k*

T(n) = 5n

b) *T(n + 1) = n + T(n)*

*T(0) = 0*

*T(n) = (n-1) + T(n-1) + (n-1)*

*T(n) = (n-2) + T(n-2) + (n-1)*

*T(n) = (n-3) + T(n-3) + (n-1)*

*T(n) = (n-k) + T(n-k)+ (n-1)*

*n = k*

*T(k) = 0 + T(0) + (k-1)*

*T(n) = T(0) + 0 + 0 + 1 + 2 + 3….*

*i = 0 Σ n = i – n*

*= n(n+1)/2 – n*

*= n2-n/2*

## 7)

a) *T(n + 1) = 2T(n)*

*T(0) = 1*

*T(1) = 2T(n-1)*

*T(1) = 2T(0)*

*T(1) = 0*

*T(n) = 2T(n-1)*

*T(n) = 2(2T(n-2))*

*T(n) = 2(2(2T(n-3)))*

*Put k*

*T(n) = 2kT(n-k)))*

*n = k*

*T(n) = 2nT(0)*

*T(n) = 2n.1*

*T(n) = 2n*

Hence Proved

b) *T(n+1) = 2n+1 + T(n)*

*T(n) = 2n + T(n-1)*

*T(n) = 2n + 2n-1 T(n-2)*

*T(n) = 2n + 2n-1 + 2n-2 + T(n-3)*

*T(n) = 2n + 2n-1 + 2n-2 +...+ 2n-(k-1) + T(n-k)*

*Let k = n*

*T(n) = 20 + 21 + 22… 2n-1 +2n*

*= (2n+1-1)/(2-1)*

*= 2n+1 -1*

## 8)

a) Expanding the recurrence relation:

*T(n) = n + T(n/2)*

*T(n) = n + n/2 + T(n/4)*

*T(n) = n + n/2 + n/4 + T(n/8)*

*T(n) = n + n/2 + n/4 + ... + T(1)*

where n = 2^m

*= 1+ 2 + 22 + … + 2m-1+2m … Using the assumption*

*= 2m+1 -1*

*= 2n – 1*

b) Expanding the recurrence relation:

*T(n) = 1 + T(n/3)*

*T(n) = 1 + 1 + T(n/9)*

*T(n) = 1 + 1 + 1 + T(n/27)*

*T(n) = 1 + 1 + 1 + ... + T(1)*

*T(n) = k + T(n/3k)*

*n = 3m*

*T(3m) = k + T(3m-k)*

*T(3m) = k + T(1)*

*T(n) = log3n+1*