CS131 Homework #6 (13 pts)

1. (3 pts) The reversal of a string *w* is the string *wR* consisting of the characters of *w* in the reverse order.
   1. (2 pts) Give a recursive algorithm for the reversal of a string
      1. Base step: empty string, one character string
      2. Recursive step: a string of length n: w=xy, where x is a string of length n-1, y is a character

**Answer:**

**if *w* is empty or length(*w* )==1:**

***wR* =*w***

**else if the string of length n>1:**

**represent w=xy, where x is a string of length n-1, y is a character**

***wR* =y*xR***

**return *wR***

**Note: the algorithm may be written in a different form, but check that the base and recursive steps are explained correctly.**

* 1. (1 pt) Prove that the recursive algorithm is correct. (Hint: use math induction).

**Answer:**

**Base step: If the string is empty or consists of one char – its reversal is itself.**

**Ind. step: assume that we know how to obtain *xR* for a string x of length n-1, n-1≥0.**

**Then for any string of length n: represent *w=xy*, where x is a string of length n-1, y is a character, *wR* =y*xR* will be a reverse string for *w*: *y -* the first char of *wR* is the last char of *w*, 2nd to the last char of *wR* are 1st to one before last char-s of *w* in the reversed order.**

1. (2 pts) If you are not familiar with Quicksort algorithm – read about it online, e.g. https://en.wikipedia.org/wiki/Quicksort)
   1. (1 pt) Describe Quicksort algorithm using pseudocode, using calls to Partition and recursive calls to Quicksort. (Do not provide a pseudocode for Partition, but describe its effect on the array).

**Answer:**

**algorithm** Quicksort(A, lo, hi) **is**

**if** lo < hi **then**

p := Partition(A, lo, hi)//partitions the array into the elements a[x]≤a[p] – pivot for lo≤x≤p-1, and a[x]>a[p] for p+1≤x≤hi.

Quicksort(A, lo, p – 1)

Quicksort(A, p + 1, hi)

* 1. (1 pt) Prove that Quicksort is correct (assuming that Partition is correct). (Hint: use strong induction.)

**Answer:**

**Base step: lo≥hi, A is empty or one element array – already sorted.**

**Inductive step:**

**Inductive hypothesis: assume that Quicksort works correctly on all the arrays of size k<n: it sorts them in increasing order.**

**Partition divides the array into the left part (consisting of elements smaller or equal to a[p] – pivot element) and right part (consisting of elements bigger than a[p]). Then we sort left part and right part via Quicksort, which will work correctly, by inductive hypothesis. If left part is sorted in increasing order, right part is sorted in increasing order, and any element of the left part is less than any element of the right part, then the whole array is sorted in increasing order.**

1. (2 pts) A model for the number of lobsters caught per year is based on the assumption that the number of lobsters caught in a year is the average of the number caught in two previous years.
   1. (1 pt) Find a recurrence relation for Ln – the number of lobsters caught in year n.

**Answer: *Ln*= 0.5*Ln*−1 + 0.5*Ln*−2 for *n*≥2.**

* 1. (1 pt) Solve this recurrence relation (find Ln), if 100,000 lobsters were caught in year 1 and 300,000 lobsters were caught in year 2.

**Solution:**

***Ln*= 0.5*Ln*−1 + 0.5*Ln*−2 for *n*≥2, *L1*=100,000, *L2*=300,000.**

***Characteristic equation: r2=*0.5*r*+ 0.5.**

***Roots: r1 = -0.5, r2=1.***

***Ln*=*α1 + α2 =* *α1 + α2* = *α1 + α2***

***Use initial conditions:***

***-0.5α1+ α2 =*100,000,**

**0.25*α1+ α2*=300,000.**

**0.75 *α1=*200,000,**

***α2 =*100,000+*0.5α1***

***α1=*266,667, *α2=*233,333.**

***Answer: Ln=* 266,667*+* 233,333.**

1. (3 pts) Solve these recurrence relations together with the initial conditions:
   1. *(1 pt) an* = 2*an*−1 − *an*−2 for *n*≥2, *a*0 = 4, *a*1 = 1

**Solution:**

***Characteristic equation: r2=*2*r*- 1.**

***Roots: r0 = 1.***

***an*=*α1 + α2 =* *α1 + α2***

***Use initial conditions:***

***α1 =*4,**

***α1+ α2*=1,**

***α2*=-3.**

**Answer: *an*=*4-3.***

* 1. *(1 pt) an* = 2*an*−1 + *an*−2 − 2*an*−3 for *n*≥3, *a*0 = 3, *a*1 = 6, *a*2 = 0

**Solution:**

***Characteristic equation: r3=*2*r2*+ r-2**

***Roots: r1 = 1, r2 = -1, r3 = 2***

***an*=*α1 + α2 α3***

***Use initial conditions:***

***α1 + α2 α3* = 3,**

***α1 - α2 α3*= 6,**

***α1 + α2 α3*= 0.**

**Solving this linear system we get:**

***α1 =6, α2 α3* = -1,**

**Answer: *an*= *62 .***

* 1. *(1 pt) an* = 2*an*−2 - *an*−4 for *n*≥4, *a*0 = 1, *a*1 = 0, *a*2 = 1, *a*3 = 0.

**Solution:**

***Characteristic equation: r4=*2*r2*-1**

***Roots: r1 = 1, r2 = -1, both with multiplicity 2.***

***an*=*α11 + α12 +α21 α22 =***

***=α11 + α12  +α21 α22***

***Use initial conditions:***

***α11 +α21* =1**

***α11 + α12 -α21 α22* =0**

***α11 + 2α12 +α21 2α22* =1**

***α11 + 3α12 -α21 α22* =0**

**Solving this linear system we get:**

***α11* =0.5, *α12* =0, *α21*=0.5, *α22* =0**

**Answer: *an*= *0.5+ 0.5***

1. (3 pts) Prove Theorem 2 (on solving recurrence relation of 2nd degree with 2 initial conditions when characteristic equation has a repeated root) from Lecture 11, slide 26.

(Hint#1: follow the proof strategy for Theorem 1 from Lecture 11, slides 19-22.

Hint #2: if is a repeated root of char. equation: *r2 =* c1*r+ c*2,

then it satisfies the char. equation: =c1+*c*2, and because it is a repeated root, we get: D=+4*c*2=0, *=* c1 /2, so D/2= c1 *+ 2c*2=0.)

**We need to prove:**

1. ***an =* α1+ α2 *→ an =* c1*an*−1 *+ c*2*an−*2 for *≥* 2 and*a0 =C0, a1 =C1*.**
2. ***a0 =C0, a1 =C1*, *an =* c1*an*−1 *+ c*2*an−*2 *→ an =* α1+ α2**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**a)**

**(1 pt) Proof of *an =* α1+ α2 *→ an =* c1*an*−1 *+ c*2*an−*2**

**Take *an =* α1+ α2 *, where* is a repeated root of char. equation: *r2 =* c1*r+ c*2**

**c1*an*−1 *+ c*2*an−*2=**

**=c1α1+ c1α2 + *c*2α1+ *c*2α2=**

**=α1 (c1 *+ c*2)+ α2 (c1 *+ c*2)=**

**= α1 + α2 (c1+*c*2) - α2 (c1 *+ 2c*2=**

**= α1 + α2  + 0 = *an***

**(1pt) Proof that *an =* α1+ α2 *→ a0 =C0, a1 =C1*.**

**Taking *C0 =* α1+ α2 *=* α1**

**and *C1 =* α1+ α2**

**we get α1*= C0,***

**α2*=C1 /- C0***

***r2 =* c1*r+ c*2 =( *r-*) *2 so if both* c1 and *c*2≠0 then≠0.**

**Thus, we can pick α1andα2, so that*an =* α1+ α2satisfies *a0 =C0, a1 =C1.***

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**(1 pt) Proof of b) follows from the fact that recurrence relation *an =* c1*an*−1 *+ c*2*an−*2 with the initial conditions *a0 =C0, a1 =C1* has a unique solution, so if *an* is a solution of the rec. relation satisfying the initial conditions, then *an =* α1+ α2 *,* with α1*= C0,* α2*=C1 /- C0,* since we proved earlier that the above expression solves the rec. relation with the initial conditions.**