Chapter 7: Recurrence Relations
Correction of Exercises

I. Introduction:

Exercise 1: Assume that a person invests \$2000 at 14 percent interest compounded annually. Let An represent the amount at the end of n years.

a) Find a recurrence relation for the sequence A0, A1, …

$$A_n = (1.14)A_{n-1}$$

b) Find an initial condition for the sequence A0, A1, …

$$A_0 = 2000$$

c) Find A1, A2, and A3.

$$A_1 = 2280, \quad A_2 = 2599.20, \quad A_3 = 2963.088$$

d) Find an explicit formula for A_n.

$$A_n = (1.14)^n \; 2000$$

e) How long will it take for a person to double the initial investment?

We must have $A_n = 4000$ or $(1.14)^n$ 2000 = 4000 or $(1.14)^n = 2.$

Taking the algorithm of both sides, we must have n log 1.14 = log 2.

Thus n = $\frac{\log 2}{\log 1.14} = 5.29$

Exercise 2: Let Sn denote the number of n-bit strings that do not contain the pattern 000. Find the recurrence relation and initial conditions for the sequence {Sn}.

We count the number of n-bit strings not containing the pattern 000.

- Begin with 1. In this case, if the remaining (n-1)-bit string does not contain 000, neither will the n-bit string. There are S_{n-1} such (n-1)-bit string.
- Begin with 0, there are two cases to consider

1. Begin with 01. In this case, if the remaining (n-2)-bit string does not contain 000, neither will the n-bit string. There are S_{n-2} such (n-2)-bit string.
2. Begin with 00. Then the third bit must be a 1 and if the remaining (n-3)-bot string does not contain 000, neither will the n-bit string. There are S_{n-3} such (n-3)-bit string.

Since the cases are mutually exclusive and cover all n-bit strings (n > 3) not containing 000, we have $S_n = S_{n-1} + S_{n-2}\ S_{n-3}$ for n > 3. S_1 = 2 (there are two 1-bit strings), S_2 = 4 (there are four 2-bit strings), and S_1 = 7 (there are seven 3-bit strings but one of them is 000).

II. Solving Recurrence Relations:

Exercise 1: Tell whether or not each relation is linear homogeneous recurrence relation with constant coefficients. Give the order of each linear homogeneous recurrence relations with constant coefficients.

a) An = -3 An-1

   Yes, order 1

b) An = An-1 + n

   No

c) An = (lg2n) An-1 – [lg(n-1)] An-2

   No

d) An = - An-1 + 5 An-2 – 3 An-3

   Yes, order 3.

Exercise 2: Solve the recurrence relation for the initial condition given.

a) An = -3 An-1; A0 = 2

$$a_n = 2\,(-3)^n$$

b) An = 6 An-1 – 8 An-2; A0 = 1 and A1 = 0

$$a_n = 2^{n+1} - 4^n$$

c) 2An = 7 An-1 – 3 An-2; A0 = A1 = 1

$$a_n = (2^{2-n} + 3^n)/5$$

d) An = -8 An-1 – 16 An-2; A0 =2 and A1 = -20

$$a_n = 2\,(-4)^n + 3n\,(-4)^n$$

Exercise 3: Show that $f_{n+1} \geq (\frac{1+\sqrt{5}}{2})^{n-1}$ n ≥ 1

Where f denotes the Fibonacci sequence.

We estimate the inequality by using induction n.

The base cases n = 1 and n = 2 are left to the reader.

Now assume that the inequality is true for values less than n+1, then

$f_{n+2} = f_{n+1} + f_n \geq$

$$(\frac{1+\sqrt{5}}{2})^{n-1} + (\frac{1+\sqrt{5}}{2})^{n-2} = \left(\frac{1+\sqrt{5}}{2}\right)^{n-2}\left(\frac{1+\sqrt{5}}{2} + 1\right) = \left(\frac{1+\sqrt{5}}{2}\right)^{n-2}\left(\frac{1+\sqrt{5}}{2}\right)^{2} = \left(\frac{1+\sqrt{5}}{2}\right)^{n}$$

and the induction step is complete.

$$\frac{1+\sqrt{5}}{2} + 1 = \frac{2(1+\sqrt{5})}{4} + \frac{4}{4} = \frac{2+(2\sqrt{5})+4}{4} = \frac{1+(2\sqrt{5})+5}{4} = \frac{1+(2\sqrt{5})+\sqrt{5}^2}{4} = (\frac{1}{2})^2\,(1+\sqrt{5})^2 = (\frac{1+\sqrt{5}}{2})^2$$

III. Applications to the analysis of Algorithms:

Exercise 1: Refer to the sequence

S1 = C, S2= G, S3 = J, S4 = M, S5 = X

1. Show how the algorithm of the binary search (slide 27) executes in case key = G

At line 2, since $i > j$, $(1 > 5)$ is false, we proceed to line 4, where we set k to 3. At line 5, since “key” (G) is not equal to S3 (J), we proceed to line 7. At line 7, key < Sk (G < J) is true, so at line 8 we set j to 2. We then invoke this algorithm with $i = 1$, $j = 2$ to search for key in

$$S1 = C, \qquad S2 = G$$

At line 2, since $i > j$ $(1 > 2)$ is false, we proceed to line 4, where we set k to 1. At line 5, since key (G) is not equal to S1 (C), we proceed to line 7. At line 7, key < Sk (G< C) is false, so at line 10 we set i to 2. We then invoke this algorithm with $i = j = 2$ to search for key in $S2 = G$

At line 2, since $i > j$ $(2 > 2)$ is false, we proceed to line 4, where we set k to 2. At line 5, since key (G) is equal to S2 (G), we return 2, the index of key in the sequence S.

2. Show how the algorithm of the Binary search (slide 27) executes in case key = Z.

At line 2, since $i > j$, $(1 > 5)$ is false, we proceed to line 4, where we set k to 3. At line 5, since “key” (Z) is not equal to S3 (J), we proceed to line 7. At line 7, key < Sk (Z < J) is true, so at line 10 we set j to 4. We then invoke this algorithm with $i = 4$, $j = 5$ to search for key in

$$S4 = M, \qquad S5 = X$$

At line 2, since $i > j$ $(4 > 5)$ is false, we proceed to line 4, where we set k to 4. At line 5, since key (Z) is not equal to S4 (M), we proceed to line 7. At line 7, key < Sk (Z< M) is false, so at line 10 we set i to 5. We then invoke this algorithm with $i = j = 5$ to search for key in $S5 = X$

At line 2, since $i > j$ $(5 > 5)$ is false, we proceed to line 4, where we set k to 5. At line 5, since key (Z<X) is false, so at line 10 we set i to 6. We then invoke this algorithm with $i = 6$ and $j = 5$.

At line 2, since $i > j$ $(6> 5)$ is true, we return 0 to indicate that we failed to find key.

Exercise 2: Professor Larry proposes the following version of binary search:

```
binary_search3(s, i, j, key){
  while ( i ≤ j) {
     k = |_(i+j)/2_|
     if (key==Sk)
        return k
     if (key < Sk)
        j = k
    else
        i = k
   }
   return 0
 }
```

Is professor's version correct (does it find key if it is present and return 0 if it is not present? If the professor's version is correct, what is the worst-case time?

The algorithm is not correct. If s is a sequence of length 1, S1 = 9 and key = 8, the algorithm does not terminate.