Kayley Seow

Chapter 3

Chapter 3 Written Homework

Section 3.2 (p. 127): 19, 21

19. An algorithm that outputs the index of the first occurrence of the value *key* in the sequence and outputs 0 if *key* is not in the algorithm:

```
procedure keymatch (k, s, n)
for i := 0 to n do
    if k := s[i] then
            return (i)
    return (0)
end keymatch
```

21. An algorithm that outputs the index of the first item that is less than its predecessor in the sequence, and outputs 0 if items are in increasing order:

```
procedure increasing (s, n)
for i := 1 to n-1 do
    if s[i] > s[i+1] then
            return (i)
    return (0)
end increasing
```

Section 3.3 (p. 131): 1-5 (all), 13, 22-24 (all)

1. GCD of 60, 90 = 30

- 90 = 60*1 + 30
- 60 = 30*2 + 0

2. GCD of 110, 273 = 1

- 273 = 110*2 + 53
- 110 = 53*2 + 4
- 53 = 4*13 +1
- 4 = 1*4 + 0

3. GCD of 220, 1400 = 20

- 1400 = 220*6 + 80

- 220 = 80*2 +60
- 80 = 60*1 + 20
- 60 = 20*3 + 0

4. GCD of 315, 825 = 15

- 825 = 315*2 + 195
- 315 = 195*1 + 120
- 195 = 120*1 + 75
- 120 = 75*1 + 45
- 75 = 45*1 + 30
- 45 = 30*1 + 15
- 30 = 15*2 + 0

5. GCD of 20, 40 = 20

- 40 = 20*2 + 0

13. Suppose that a, b, and c are positive integers. Show that if a|b and b|c, then a|c.

- From the property, ak = b and bj = c.
- Substituting ak into b, we get c = (ak)j.
- Rearrange to get c = a(kj).
- Because kj are two integers, their product will be an integer and we can treat them as one.
- Using division algorithm property, we can conclude that a|c.

22. An algorithm to compute the greatest common divisor of two nonnegative integers a and b, not both zero, that uses subtraction but not the modulus operation.

```
procedure gcd (a, b)
    if a<b then
            swap a, b
    while b :!= 0 do
            if a<b then
                    swap a, b
            a := a-b
    return (a)
end gcd
```

23. Show that for some n, postage of n cents or more can be achieved by using only p-cent and q-cent stamps provided that gcd(p,q) = 1.

- If either p or q is 1, we can make n cents postage for all n ≥ 1 by using n 1-cent stamps, so assume that both p and q are greater than 1.

- There exists integers s and t such that $sp + tq = 1$, and because of the first step, s and t are both not equal to 0, and neither are negative.
- Let $n = -t(p-1)q$.
- $n+j = -t(p-1)q + j(sp+tq) = (js)p + (-t(p-1)+jt)q$.
- If $0 \leq j \leq p-1$, then $-t(p-1) + jt \geq -t(p-1) + t(p-1) = 0$.
- Using induction, we can prove this.

24. Show that if $\gcd(p, q) > 1$, the above statement is false.

- The proof does not work because you could just make n cents postage by using 1 cent stamps.

Section 3.4 (p. 137): 8, 9, 12, 13, 27, 28

8. Use the formulas $S1 = 1$, $Sn = Sn-1 + n$, $n >= 2$ to write a recursive algorithm that computes $Sn = 1 + 2 + 3 + n$ and give a proof.

- Input: n, Output: $1 + 2 + \ldots + n$
- procedure sum(n)

```
procedure sum(n)
        if n := 1 then
                return (1)
        return (sum(n-1)+n)
end sum
```

- Proof: If n is 1, then we return 1. n is greater than 1 so we add the inductive statement of n-1 which means $sum(n-1) = 1 + 2 + \ldots + n$, which is the correct value. (The algorithm returns $1 + 2 + \ldots + n$ when the input is n is true when $n = 1$ and it is also true for $n = k+1$ when the statement is true for $n = k$.)

9. Use the formulas $S1 = 2$, $Sn = Sn-1 + 2n$, $n >= 2$ to write a recursive algorithm that computes $Sn = 2 + 4 + 6 + 2n$.

- Input: n, Output: $2 + 4 + \ldots + 2n$
- procedure sum2(n)

```
procedure sum2(n)
        if n := 1 then
                return (2)
        return (sum(n-1)+2n)
end sum
```

- Proof: If n is 1, then we return 2. n is greater than 1 so we add the inductive statement of n-1 which means $sum(n-1) = 2 + 4 + \ldots + 2n$, which is the correct value. (The algorithm returns $2 + 4 + \ldots + 2n$ when the input is n is true when $n = 1$ and it is also true for $n = k+1$ when the statement is true for $n = k$.)

12. Write a recursive algorithm to find the minimum of a finite sequence of numbers. Give a proof using mathematical induction that your algorithm is correct.

- Input: s, n, Output: s_{min} (minimum of the sequence)
- procedure minimum(s, n)

```
procedure minimum(s, n)
        if n := 1 then
```

```
                return s[1]
        if s[n] < s[n-1] then
                s[n-1] = s[n]
                return (minimum(s, n-1))
        return (minimum(s, n-1))
end minimum
```

- Proof: Let P(n) mean that the algorithm minimum returns the minimum from the sequence s when s has n terms. The base step, n = 1 is true, and P(1) is true. Inductive step, we need to prove that P(k+1) is true; since k >= 1, k+1 >= 2. If the smaller value comes after the larger value, then we need to switch the values to make sure that we are returning the minimum value. If the larger value comes after the smaller value, it is ignored, and we can continue on with the function. These both prove that P(k) is true, and thus P(k+1) is true as well; as long as n is a positive integer, this algorithm works.

13. Write a recursive algorithm to find the maximum of a finite sequence of numbers. Give a proof using mathematical induction that your algorithm is correct.

- Input: s, n, Output: s_{max} (maximum of the sequence)
- procedure maximum(s, n)

```
        if n := 1 then
                return s[1]
        if s[n] > s[n-1] then
                s[n-1] = s[n]
                return (maximum(s, n-1))
        return (maximum(s, n-1))
end maximum
```

- Proof: Let P(n) mean that the algorithm minimum returns the minimum from the sequence s when s has n terms. The base step, n = 1 is true, and P(1) is true. Inductive step, we need to prove that P(k+1) is true; since k >= 1, k+1 >= 2. If the larger value comes after the smaller value, then we need to switch the values to make sure that we are returning the minimum value. If the smaller value comes after the larger value, it is ignored, and we can continue on with the function. These both prove that P(k) is true, and thus P(k+1) is true as well; as long as n is a positive integer, this algorithm works.

27. Use mathematical induction to show that for n >= 5, $f_n > (3/2)^n$.

Base case: $f_5 = 8$, $8 > 1.5^5$, which is approximately 7.59.

Inductive step: The Fibonacci sequence gets bigger at a faster rate than 3/2 does after 5, which is why this is true.

28. Use mathematical induction to show that for n >=1, $f_n < (2)^n$.

Base case: $f_1 = 2$, $2 = 2^1$, which is 2.

Inductive step: The Fibonacci sequence increases at a slower rate compared to the function, as 2^n is an exponential function. Therefore, this is why this is true.

Section 3.5 (p. 149): 1-12 (all), 17, 19

1. $6n + 1$, constant.

2. $2n^2 + 1$, quadratic.

3. $6n^3 + 12n^2 + 1$, cubic.

4. $3n^2 + 2nlgn$, quadratic.

5. $2lgn + 4n + 3nlgn$, logarithmic.

6. $6n^6 + n + 4$, exponential.

7. $2 + 4 + 6 + \ldots + 2n$, quadratic.

8. $(6n +1)^2$, quadratic.

9. $(6n+4)(1+lgn)$, logarithmic.

10. $[(n+1)(n+3)]/(n+2)$, constant.

11. $[(n^2 + lgn)(n+1)]/n+n^2$, linear.

12. $2 + 4 + 8 + 16 + \ldots + 2^n$, exponential.

17. Constant.

19. Quadratic.

Section 3.7 (p. 160): 1-9 (all)

1. The message “COOL BEAVIS” using the key would be encrypted to “FKKGEJAIMWQ”.

2. The message “UTWR ENKDTEKMIGYWRA” using the key would be decrypted to “DRINK YOUR OVALTINE”.

3. 333 encrypted using the public key 713, 29 would be 293.

4. 411 decrypted using $s = 569$ would be 500.

We use primes $p = 17$, $q = 23$, and $n = 31$.

5. z is equal to 391.

6. ϕ is equal to 352.

7. s is equal to 159 because it is between 0 and ϕ and satisfies $31*159 \pmod{352} = 1$.

8. 101 encrypted using the public key $z = 391$, $n = 31$ is 186. You get this by recognizing that $c = a^n(\text{mod } z)$, which equals $101^{31}(\text{mod } 391)$, which equals 186.

9. 250 decrypted is equal to 10. You get this by recognizing that a is equal to $c^s(\text{mod } z)$, which equals $250^{159}(\text{mod } 391)$, which equals 10.