Suggested Solutions to HW #7

2. (6.16) Compute the *next* table as in the KMP algorithm for the string *abbababbaa*. Show the details of your calculation.

Solution.

$i =$	1	2	3	4	5	6	7	8	9	10
$B =$	a	b	b	a	b	a	b	b	a	a
$next =$	-1	0	0	0	1	2	1	2	3	4

Initial : $next(1) = -1; next(2) = 0$

$i = 3$:

$$j_1 = next(i-1) + 1 = next(2) + 1 = 1; b_{i-1} \neq b_{j_1}$$
$$\Rightarrow \quad j_2 = next(j_1) + 1 = next(1) + 1 = 0; next(i) = j_2 = 0$$

$i = 4$:

$$j_1 = next(i-1) + 1 = next(3) + 1 = 1; b_{i-1} \neq b_{j_1}$$
$$\Rightarrow \quad j_2 = next(j_1) + 1 = next(1) + 1 = 0; next(i) = j_2 = 0$$

$i = 5$:

$$j_1 = next(i-1) + 1 = next(4) + 1 = 1; b_{i-1} = b_{j_1}; next(i) = j_1 = 1$$

$i = 6$:

$$j_1 = next(i-1) + 1 = next(5) + 1 = 2; b_{i-1} = b_{j_1}; next(i) = j_1 = 2$$

$i = 7$:

$$j_1 = next(i-1) + 1 = next(6) + 1 = 3; b_{i-1} \neq b_{j_1}$$
$$\Rightarrow \quad j_2 = next(j_1) + 1 = next(3) + 1 = 1; b_{i-1} = b_{j_2}; next(i) = j_2 = 1$$

$i = 8$:

$$j_1 = next(i-1) + 1 = next(7) + 1 = 2; b_{i-1} = b_{j_1}; next(i) = j_1 = 2$$

$i = 9$:

$$j_1 = next(i-1) + 1 = next(8) + 1 = 3; b_{i-1} = b_{j_1}; next(i) = j_1 = 3$$

$i = 10$:

$$j_1 = next(i-1) + 1 = next(9) + 1 = 4; b_{i-1} = b_{j_1}; next(i) = j_1 = 4$$

□

3. (6.17) Given two strings *babcbb* and *baacbab*, compute the minimal cost matrix $C[0..6, 0..7]$ for changing the first string character by character to the second one. Aside from giving the cost matrix, please show the details of how the entry $C[6, 7]$ is computed.

Solution.

In the following table, r, i, and b denote replacement, insertion, and deletion respectively.

		b	a	a	c	b	a	b
	0	1	2	3	4	5	6	7
b	1	$0(r)$	$1(i)$	$2(i)$	$3(i)$	$4(i,r)$	$5(i)$	$6(i,r)$
a	2	$1(d)$	$0(r)$	$1(i,r)$	$2(i)$	$3(i)$	$4(i,r)$	$5(i)$
b	3	$2(d,r)$	$1(d)$	$1(r)$	$2(i,r)$	$2(r)$	$3(i)$	$4(i)$
c	4	$3(d)$	$2(d)$	$2(d,r)$	$1(r)$	$2(i)$	$3(i,r)$	$4(i,r)$
b	5	$4(d,r)$	$3(d)$	$3(d,r)$	$2(d)$	$1(r)$	$2(i)$	$3(i,r)$
b	6	$5(d,r)$	$4(d)$	$4(d,r)$	$3(d)$	$2(d,r)$	$2(r)$	$2(r)$

$$C[6,7] = \min \begin{cases} C[5,7] + 1 (= 4) \\ C[6,6] + 1 (= 3) \\ C[5,6] + c[6,7] (= C[5,6] = 2) \end{cases} = 2$$

□

4. (6.39) Let A and B be two sets, both with n elements, such that A resides in computer P and B in computer Q. P and Q can communicate by sending messages, and they can perform any kind of local computation. Design an algorithm to find the nth smallest element (i.e., the median) of the union of A and B. You can assume, for simplicity, that all the elements are distinct. Your goal is to minimize the number of messages, where a message can contain one element or one integer. Please present your algorithm in an adequate pseudo code and make assumptions wherever necessary. What is the number of messages in the worst case?

Solution. (Jinn-Shu Chang, modified by Ming-Hsien Tsai)
To facilitate the collaborative search process, the two computers store the two sets of numbers in sorted arrays (indexed from 1 to n), also referred to as A and B. They both run a copy of the algorithm **FindNthElement** as shown below. Computer P invokes the algorithm with **FindNthElement**$(A, 1, n)$ and Q with **FindNthElement**$(B, 1, n)$. The algorithm proceeds in rounds by utilizing the following proposition.

Proposition 0.1 *The nth smallest element in $2n$ elements must be greater than or equal to $n-1$ elements and smaller than or equal to n elements.*

In each round, the two computers try to eliminate half of the elements in the remaining arrays. This is done by exchanging and comparing the median elements in the remaining arrays. The pair of functions *Send* and *Receive* are invoked to send the median element to and receive the corresponding element from the other computer. After the exchange, a quarter of elements that are not greater than the smaller median can be removed since they are not greater than or equal to $n-1$ elements. Similarly, a quarter of elements that are not smaller than the larger median can be removed. Since half of elements are removed in each round (a quarter before the nth smallest element and a quarter after the nth smallest element), the nth smallest element is still the median in the remaining

arrays. The algorithm continues until both remaining arrays contain exactly only one element when the smaller median is selected as the nth smallest element (because of Proposition 0.1). As the number of elements is halved in each round, there will be a total of $\log n$ rounds and the number of messages exchanged will be $2 \times \log n$. (Note: this algorithm always needs $\log n$ rounds. Can you do better?)

Algorithm FindNthElement$(X, \mathit{Left}, \mathit{Right})$
Input: X (the sorted array that stores the set, either A or B, residing in this computer), *Left* (the left boundary of the remaining array, set to 1 initially), and *Right* (the right boundary of the remaining array, set to n initially).
Output: the nth smallest element of the union of A and B.

```
begin
    Middle := ⌊(Left+Right)/2⌋;
    EvenFlag := (Right − Left) mod 2;
    my_mid := X[Middle];
    Send(my_mid);
    Receive(other_mid);

    /* Allow duplicate elements. */
    if my_mid = other_mid then
        return my_mid;

    if Left = Right then
        return min(my_mid, other_mid);
    if my_mid > other_mid then
        FindNthElement(X, Left, Middle);
    else
        FindNthElement(X, Middle + EvenFlag, Right);
end
```

□