

**PROVIDED BY  
SHAHZAD AHMAD  
bc020400591@VU.EDU.PK  
SHAHZAD\_591@YAHOO.COM  
MODRATOR BHOLA GROUP  
PH: 0923-612958**

## **CS502 Fundamentals of Algorithms**

**Mid Term Examination - November 2004**

**Time Allowed: 90 Minutes**

### **Instructions**

**Please read the following instructions carefully before attempting any of the questions:**

1. The duration of this examination is 90 Mins.
2. This examination is closed book, closed notes, closed neighbors.
3. Do not ask any questions about the contents of this examination from anyone.
  - a. If you think that there is something wrong with any of the questions, attempt it to the best of your understanding.
  - b. If you believe that some essential piece of information is missing, make an appropriate assumption and use it to solve the problem.
4. Some of the examination consists of multiple-choice questions. Choose only one choice as your answer.
  - a. If you believe that two (or more) of the choices are correct for a particular question, choose the best one.
  - b. On the other hand, if you believe that all of the choices provided for a particular question are wrong then select the one that appears to you as being the least wrong.

**\*\*WARNING: Please note that Virtual University takes serious note of unfair means. Anyone found involved in cheating will get an `F` grade in this course.**

**Total Marks: 70  
Questions: 7**

**Total**

**Question No. 1**

**Marks : 15**

A PC like computer can perform  $10^9$  operations per second. Consider that we have five different algorithms for a specific problem. For each algorithm  $i$ , we know the number of operations  $T_i(n)$  it will perform on a problem of size  $n$ :

$$T_1(n) = 6000000 \cdot n$$

$$T_2(n) = 60000 \cdot n \log n$$

$$T_3(n) = 0.003 \cdot n^2$$

$$T_4(n) = 10^{-6} \cdot n^3$$

$$T_5(n) = 10^{-18} \cdot 2^n$$

For each algorithm compute the size  $n_{\max}$  of the largest problem the respective algorithm can solve within 1 second, 1 minute and 1 hour. Enter the maximal problem sizes into the following

	1 second	1 minute	1 hour
$n_{\max}(T_1)$			
$n_{\max}(T_2)$			
$n_{\max}(T_3)$			
$n_{\max}(T_4)$			
$n_{\max}(T_5)$			

**Question No. 2**

**Marks : 5**

Describe an efficient algorithm to find the **median** of a set of  $10^6$  integers; it is known that there are fewer than 100 distinct integers in the set.

**Question No. 3**

**Marks : 20**

Recall the counting sort algorithm, where

- 1 n is the number of elements to be sorted
- 2 each element is an integer in the range 1 to k
- 3 A is the input array, of size n
- 4 B is the final sorted output array
- 5 C is the intermediate array of size k

```

1: for i = 1 to k
2:   C[i] = 0
3: for i = 1 to n
4:   C[A[i]] = C[A[i]] + 1
5: for i = 2 to k
6:   C[i] = C[i] + C[i-1]
7: for i = n downto 1
8:   B[C[A[i]]] = A[i]
9:   C[A[i]] = C[A[i]] - 1

```

(a) For the input array [6,3,7,5,4,3,3,2,8,9,1,6], and k = 9, specify the contents of the arrays B and C

- (4 points) after the second for-loop (line 3); C =
- (2 points) after the third for-loop (line 5); C =
- After **each** of the *first three* iterations of the fourth loop (line 7);

1. (3 points)

B =

C =

2. (3 points)

B =  
C =  
3. (3 points)  
B =  
C =

- (b) (5 points) Suppose that the for-loop on line 7 above, is modified as follows:  
7: for i = 1 to n

Will the algorithm still sort the input? If no, explain why not. If yes, what is the effect of this Modification to the algorithm and the output?

**Question No. 4**

**Marks : 5**

When using merge sort to sort an array, do the recursive calls to merge sort depend on the number of elements in the array, the values of the elements or both? Briefly explain.

**Question No. 5**

**Marks : 5**

Insertion sort can be expressed as a recursive procedure as follows: In order to sort array  $A[1..n]$ , we recursively sort array  $A[1..n-1]$  and then insert  $A[n]$  into the sorted array  $A[1..n-1]$ . Give an equation that describes the overall running time of this algorithm on an input array of size  $n$ , in terms of the running time on smaller input.

$T(n) =$

**Question No. 6**

**Marks : 10**

Compute the edit distance and edit scripts for the strings "HEAP" and "CHEAT". Recall that the edit distance recurrence is

$$E(i, j) = \min \begin{pmatrix} E(i-1, j) + 1 \\ E(i, j-1) + 1 \\ E(i-1, j-1) + 1 \quad \text{if } A[i] \neq B[j] \\ E(i-1, j-1) \quad \text{if } A[i] = B[j] \end{pmatrix}$$

**Question No. 7**

**Marks : 10**

Using big-oh notation, give the running time of the following piece of code. Briefly justify your answer.

```
for (i = 1; i <= n; i++)  
    for (j = 1; j <= n/2; j++)  
        for (k = 1; k <= j; k++)  
            cout << "hello world\n";
```

