Department of Computer Science and Engineering

PES UNIVERSITY

UE19CS251: Design and Analysis of Algorithms(4-0-0-4-4)

Q.1 Design an algorithm for swapping two 3 digit non-zero integers *n*, *m*. Besides using arithmetic operations, your algorithm should not use any temporary variable

Q.2 Design an algorithm for computing gcd(*m*, *n*) using Euclid's algorithm.

Q.3 Write a pseudocode for an algorithm for finding real roots of equation $ax^2 + bx + c = 0$ for arbitrary real coefficients *a*, *b*, and *c*.

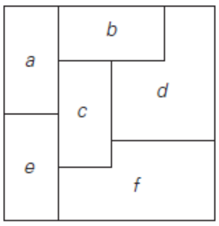
Q.4 Design an algorithm to convert a binary number to a decimal integer.

Q.5 Consider the following algorithm for the searching problem:
ALGORITHM Linearsearch *(A*[0, ..*n* − 1], key)
//Searches an array for a key value by Linear search
//Input: Array *A*[0..*n* − 1] of values and a key value to search
//Output: Returns index if search is successful
for *i* ← 0 **to** *n* − 1 **do**
if (key == *A*[*i*])
return *i*

a. Apply this algorithm to search the list 10, 92, 38, 74, 56, 19, 82, 37 for a key value 74.
b. Is this algorithm efficient?
c. When can this algorithm be used?

Q.6 Design a simple algorithm for string matching problem

Q.7 Consider the following map:



a. Explain how we can use the graph-colouring problem to colour the map so that no two neighbouring regions are coloured the same.
b. Use your answer to part (a) to colour the map with the smallest number of colours.

Q.8 For each of the following algorithms, indicate
(i) a natural size metric for its inputs;

(ii) its basic operation;
(iii) whether the basic operation count can be different for inputs of the same size:
a. computing the sum of n numbers
b. computing n!
c. finding the largest element in a list of n numbers
d. Euclid's algorithm

Q.9 Define time complexity and space complexity. Write an algorithm for adding 'n' natural numbers and find the time and space required by that algorithm

Q.10 For each of the following functions, indicate how much the function's value will change if its argument is increased fourfold.
a. $\log_2 n$ **b.** $\sqrt{n}$ **c.** n **d.** n^2 **e.** n^3 **f.** 2^n

Q.11 Compare the two functions 2^n and n^2 for various values of n. Determine when will the second function become the same, smaller, and larger than the first function.

Q.12 Use the most appropriate notation among O, Theta and omega to indicate the time efficiency class of binary search
a. in the worst case.
b. in the best case.
c. in the average case.

Q.13 From the following equalities, indicate the ones that are incorrect?

a. $6n^2 - 8n = \Theta(n^2)$ **b.** $12n^2 + 8 = O(n)$
c. $3n^2 3^n + n \log n = \Theta(n^2 3^n)$ **d.** $3n^2 \log n = \Theta(n^2)$

Q.14 For each of the following functions, indicate the class Theta*(g(n))* the function belongs to. (Use the simplest *g(n)* possible in your answers.)

a. $(n^2 + 1)^{10}$ **b.** $\sqrt{10n^2 + 7n + 3}$

Q.15 Arrange the following functions according to their order of decay (from the highest to the lowest)

$$(n+1)!2^{3n},\ 2n^4 + 2n^3 + 4,\ n \log n,\ \log n, 6n, 8n^2.$$

Q.16 algo what(a[l ..r] , l, r)

```
if l = r then
    return a[l]
else if a[l] > a[r] then
    return what(a, l + 1, r)
else
    return what(a, l, r - 1)
```

i) what does the given function do?
ii) What is the basic operation?
iii) What is the basic size?
iv) Express and solve the recurrence relation for number of operations?

Q.17 Consider the following algorithm:
ALGORITHM *Sum (n)*
//Input: A nonnegative integer n
$S \leftarrow 0$
for $i \leftarrow 1$ **to** n **do**
$S \leftarrow S + i$
return S
a. What does this algorithm compute?
b. What is its basic operation?
c. How many times is the basic operation executed?
d. What is the efficiency class of this algorithm?
e. Suggest an improved algorithm and indicate its efficiency class. If you cannot do it, try to prove that it cannot be done.

Q.18 Consider the following algorithm
ALGORITHM *GE*($A[0..n-1, 0..n]$)
//Input: An n-by-$n+1$ matrix $A[0..n-1, 0..n]$ of real numbers
for $i \leftarrow 0$ **to** $n-2$ **do**
for $j \leftarrow i+1$ **to** $n-1$ **do**
for $k \leftarrow i$ **to** n **do**
$A[j, k] \leftarrow A[j, k] - A[i, k] * A[j, i] / A[i, i]$
Find the time efficiency class of this algorithm.

Q.19 . Solve the following recurrence relations.
a. $x(n) = x(n-1) + 5$ for $n > 1$, $x(1) = 0$
b. $x(n) = 3x(n-1)$ for $n > 1$, $x(1) = 4$
c. $x(n) = x(n-1) + n$ for $n > 0$, $x(0) = 0$

Q.20. Consider the following recursive algorithm.
ALGORITHM *Q(n)*
//Input: A positive integer n
if $n = 1$ **return** 1
else return $Q(n-1) + 2 * n - 1$
a. Set up a recurrence relation for this function's values and solve it to determine what this algorithm computes.
b. Set up a recurrence relation for the number of multiplications made by this algorithm and solve it.
c. Set up a recurrence relation for the number of additions/subtractions made by this algorithm and solve it.

Q.21. Consider the following recursive algorithm.
ALGORITHM *Min1*($A[0..n-1]$)
//Input: An array $A[0..n-1]$ of real numbers
if $n = 1$ **return** $A[0]$
else $temp \leftarrow Min1(A[0..n-2])$
if $temp \le A[n-1]$ **return** $temp$
else return $A[n-1]$
a. What does this algorithm compute?
b. Set up a recurrence relation for the algorithm's basic operation count and solve it.