SRN

PES University, Bangalore

(Established under Karnataka Act No. 16 of 2013)

UE16CS311

END SEMESTER ASSESSMENT (ESA) B.Tech. V SEMESTER - Dec. 2018

UE16CS311 - Advanced Algorithms

Time: 3 Hrs	Answer All Questions	Max Marks: 100

1a (06)

```
Algorithm Run(n)
    count ← 0
    i ← n
    while(i > 0)
        j ← 1
        while(j <= i)
            for k ← 1 to n
                count ← count + 1
            j ← j + 2
        i ← floor(i / 2)
    return count
```

Find the worst-case asymptotic time complexity of the above algorithm in Θ-notation.

1b (06)

```
INCREMENT( A[0…k-1] )
    i ← 0
    while i < k and A[ i ] = 1
        A[ i ] ← 0 (toggles the bit)
        i ← i + 1
    if  i < k
    then A[ i ] ← 1 (toggles the bit)
```

Using "Aggregate Method", find the amortized cost of the above operation which increments a binary number A[0..k-1].

1c (08)

```
TABLE-INSERT(T, x)
 if size[T] = 0
    then allocate table[T] with 1 slot
         size[T] ← 1
 if num[T] = size[T]
    then allocate new-table with 2 · size[T] slots
         insert all items in table[T] into new-table
         free table[T]
         table[T] ← new-table
         size[T] ← 2 · size[T]
 insert x into table[T]
 num[T] ← num[T] + 1
```

Consider the given TABLE-INSERT procedure for inserting a new element x in the dynamic table T. Find the amortized cost of the given TABLE-INSERT procedure using the potential method.

SRN

2a Find the pattern **"31415"** in the text **"2359023141526739921"** using the Rabin-Karp method. Use radix-**10** with **"mod 13"** for finding the hash values. **06**

2b **06**

```
KMP-MATCHER(T, P)
n = T.length
m = P.length
π = COMPUTE-PREFIX-FUNCTION(P)
q = 0                                // number of characters matched
for i = 1 to n                       // scan the text from left to right
    while q > 0 and P[q + 1] ≠ T[i]
        q = π[q]                     // next character does not match
    if P[q + 1] == T[i]
        q = q + 1                    // next character matches
    if q == m                        // is all of P matched?
        print "Pattern occurs with shift" i − m
        q = π[q]                     // look for the next match
```

For the **KMP-MATCHER** algorithm given above, write the **COMPUTE-PREFIX-FUNCTION** procedure it requires.

2c Explain an **O(n)** method of finding the longest common substring of two strings **T1** and **T2** of length **O(n)** using generalized suffix trees. Use the method to find the longest common substring of **T1** = "nonsense" and **T2** = "offense". **08**

3a Find the integers **x** and **y** in the equation 840**x** + 462**y** = gcd(840, 462) using the extended Euclid's algorithm. **06**

3b Explain the procedure of generating a pair of public-private keys for a participant in the RSA public-key cryptosystem. **06**

3c Let P_A, S_A, P_B, S_B are public-key and secret-key of Alice, and public-key and secret-key of Bob, respectively. Explain with a block diagram, a procedure for:
(i) Bob to send message **M** to Alice confidentially,
(ii) Alice to send a short message **M`** to Bob authenticating it is her who sent the message. **08**

4a What is memoization technique? Write an algorithm using memoization technique to find the binomial coefficient **C(n, k)** in **O(nk)** time. Use Pascal's identity given below to find the binomial coefficient. **06**

C(n, k) = C(n-1, k-1) + C(n-1, k) for n > k > 0
C(n, 0) = 1, C(n, n) = 1 for n ≥ 0

SRN

4b	Using dynamic programming, find the longest common subsequence of two strings T1 = "**GCCCTAGCG**" and T2 = "**GCGCAATG**".	06
4c	Apply dynamic programming to solve the problem of fully parenthesizing for optimally multiplying a chain of **n** matrices. Find the optimal substructure, recursive solution and write the procedure to find the optimal value (finding an optimal solution is not necessary, just the optional value is needed).	08
5a	Convert the polynomial $\mathbf{A(x) = a_0 + a_1x + a_2x^2 + a_3x^3}$ from the coefficient representation to the point-value representation.	06
5b	Explain the addition of two polynomials in point-value representation.	06

5c	RECURSIVE-FFT(a)
	$n = a.length$
	if $n == 1$
	    return a
	$\omega_n = e^{2\pi i/n}$
	$\omega = 1$
	$a^{[0]} = (a_0, a_2, \ldots, a_{n-2})$
	$a^{[1]} = (a_1, a_3, \ldots, a_{n-1})$
	$y^{[0]} =$ RECURSIVE-FFT$(a^{[0]})$
	$y^{[1]} =$ RECURSIVE-FFT$(a^{[1]})$
	for $k = 0$ **to** $n/2 - 1$
	    $y_k = y_k^{[0]} + \omega\, y_k^{[1]}$
	    $y_{k+(n/2)} = y_k^{[0]} - \omega\, y_k^{[1]}$
	    $\omega = \omega\, \omega_n$
	return y
	
	Explain the working of the above FFT algorithm to evaluate coefficient vector **a** into its value vector **y**.
		08