Xi'an Jiaotong-Liverpool University 西交利物浦大学

PAPER CODE	EXAMINER	DEPARTMENT	TEL
INT102	Wenjin Lu	Intelligent Science	1505

2nd SEMESTER 2021/22 EXAMINATIONS (Final Open Book)

BACHELOR DEGREE – Year 2

ALGORITHMIC FOUNDATIONS AND PROBLEM SOLVING

TIME ALLOWED: 2 Hours

INSTRUCTIONS TO CANDIDATES

READ THE FOLLOWING CAREFULLY:

1. The paper consists of Part A and Part B. Answer all questions in both parts. Total marks available are 100. Marks for this examination account for 80% of the total credit for INT102.

2. In Part A, each of the questions comprises 5 statements, for which you should select the one most appropriate answer.

3. Answers to questions in Part B should be written in the answer script.

4. This is an OPEN BOOK examination. You can reference textbooks and notes but discuss with other students in any way is not allowed.

5. The time of the exam is strictly limited to 2 hours.

6. At the end of the online examination, be absolutely sure to submit your answer via Learning Mall. The time for submission of your answer via Learning Mall is strictly limited to 15 minutes. Once the time is over, the submission link will be closed.

7. All answers must be in English.

PART A (60 marks)

Questions 1 to 4 refer to the following algorithm.

Algorithm: P($a[l \dots r]$)

Input: an array $a[l \dots r]$ of real numbers

begin

if $l == r$ **return** l

else

$ll = \mathrm{P}(a[l \dots \lfloor (r+l)/2 \rfloor])$

$rr = \mathrm{P}(a[\lfloor (r+l)/2 \rfloor + 1 \dots r])$

if $a[ll] > a[rr]$ **return** ll

else **return** rr

end

C

1. Which algorithm design technique is employed in the above algorithm? **2.5**

[A] Brute Force technique
[B] Greedy technique
[C] Divide- and-Conquer ✓
[D] Dynamic Programming
[E] Time and Space trade-off

$k = \log_2 n$

C

2. For $n = 2^k$ and $k \geq 1$, the time complexity of the algorithm can be best expressed by

[A] $T(n) = T(n/2) + 1$
[B] $T(n) = T(n/2)$
[C] $T(n) = 2T(n/2) + 1$
[D] $T(n) = 2T(n/2)$
[E] None of the above

$T_n = 2T(\frac{n}{2}) + 1 = 2 \cdot (2T(\frac{n}{2^2}) + 1) + 1$

$\dots$

$= 2^k + 1 + 2 + 2^3 + \dots + 2^k$

$= n + 1 \cdot (2^k - 1)$

$= 2n$

$= O(n)$

A

3. The time complexity of the algorithm is **2.5**

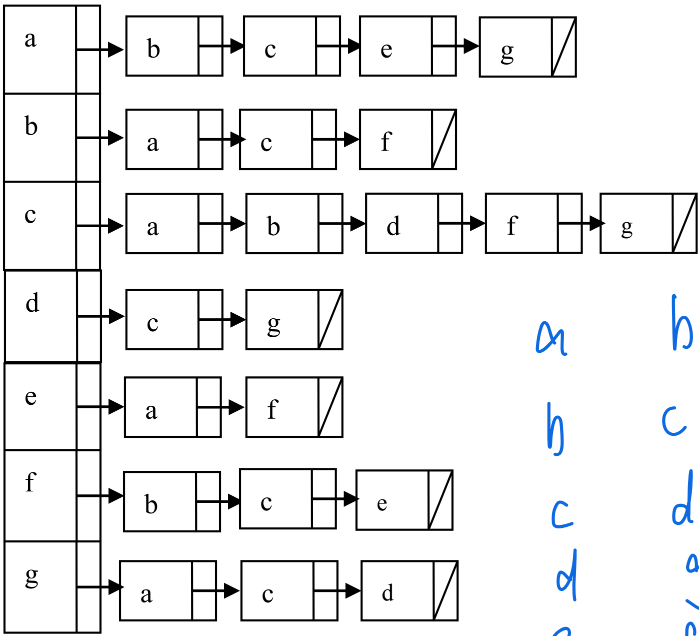
[A] $O(n)$
[B] $O(\log n)$
[C] $O(n^2)$
[D] $O(n \log n)$
[E] None of the above

D

4. What is the output of the algorithm for the input a[0..7] = [12, 12, 12, 12, 12, 12, 12, 12]? **2.5**

[A] 1
[B] 3
[C] 5
[D] 7
[E] 12

Questions 5 to 8 refer to the graph G represented by the following adjacency list



a bceg
b cf
c dfg
d g
e f
g

E.

5. Which of the following statement is true: **2.5**

[A] The total degree of the graph G is 20 and it has a Euler circuit
[B] The total degree of the graph G is 18 and it has a no Euler circuit
[C] The total degree of the graph G is 11 and it has a Euler circuit
[D] The total degree of the graph G is 29 and it has a Euler circuit
[E] The total degree of the graph G is 22 and it has no Euler circuit

C

6. Starting at the vertex *a* and resolving ties by the vertex alphabetical order, traverse the graph by breadth-first-search (BFS). Then, the order of vertices visited is **2.5**

[A] *a, b, d, e, f, c, g*
[B] *a, b, c, d, e, f, g*
[C] *a, b, c, e, g, ~~d, f~~* f, d
[D] *a, b, g, e, f, c, d*
[E] *a, b, f, g, c, d, e*

a b c eg d f

E

7. Starting at the vertex *a* and resolving ties by the vertex alphabetical order, traverse the graph by depth-first-search (DFS). Then, the last vertex being visited is **2.5**

[A] *g*
[B] *f*
[C] *c*
[D] *d*
[E] *e*

a b c d g f e

B

8. Starting at the vertex *a* and resolving ties by the vertex alphabetical order, traverse the graph by depth-first-search (DFS). Then, the 5th vertex being visited is 2.5

[A] *f*
[B] *g*
[C] *d*
[D] *c*
[E] *b*

A.

9. Let *T* be a tree constructed by Dijkstra's algorithm in the process of solving the single-source shortest path problem for a weighted connected graph G. 2.5

I. *T* is a spanning tree of G √
II. *T* is a minimum spanning tree of G X
III. T is a binary tree X

Which one of the following is correct for every weighted connected graph?

[A] I is true, II and III are false. √
[B] I and II and III are true.
[C] I and II and III are false.
[D] II is true but I and III are false.
[E] I and III are true but II is false.

Questions 10 to 11 refer to the following Bubble sort algorithm.

```
ALGORITHM Bubble Sort (A[0..n − 1])
//Sorts a given array by bubble sort
//Input: An array A[0..n − 1] of orderable elements
//Output: Array A[0..n − 1] sorted in ascending order
for i=0 to n − 2 do
    for j = n-1 down to i+1 do
        if A[j ]<A[j-1 ] swap A[j ] and A[j - 1]
```

C

10. The number of swapping operations needed to sort the numbers A[0..5]=[6，1，2，3，4，5] in ascending order using the Bubble sort algorithm is **2.5**

6 1 2 3 4 5

[A] 10
[B] 4
[C] 5
[D] 15
[E] 20

D

11. The number of key comparisons needed to sort the numbers A[0..5]= [6，1，2，3，4，5] in ascending order using the Bubble sort algorithm is **2.5**

5+4+3+2+1

[A] 10
[B] 4
[C] 5
[D] 15
[E] 20

Questions 12 to 13 refer to the following Selection sort algorithm.

```
ALGORITHM Selection Sort(A[0..n − 1])
//Sorts a given array by selection sort
//Input: An array A[0..n − 1] of orderable elements
//Output: Array A[0..n − 1] sorted in ascending order
for i =0 to n − 2 do
    min = i
    for j =i + 1 to n − 1 do
        if A[j ]<A[min]   min = j
    swap A[i] and A[min]
```

D

12. The number of comparisons needed to sort the numbers A[0..5]= [6，1，2，3，4，5] in ascending order using the selection sort algorithm is **2.5**

5+4+3+2+1

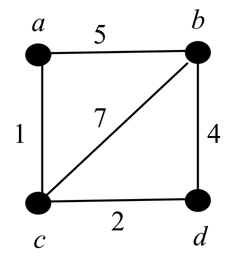
[A] 10
[B] 4
[C] 5
[D] 15
[E] 20

C

13. The number of swapping operations needed to sort the numbers A[0..5]= [6，1，2，3，4，5] in ascending order using the selection sort algorithm is **2.5**

[A] 10
[B] 4
[C] 5
[D] 15
[E] 20

Note: If a weighted graph G is represented by its adjacency matrix A, then its element $A[i, j]$ will simply contain the weight of the edge from the ith to the jth vertex if there is such an edge and a special symbol ∞, if there is no such edge. Such a matrix is called the weight matrix of G. For example, in the following, the left side is a weighted graph and the right side is its weight matrix



$$
\begin{array}{c}
\\ a \\ b \\ c \\ d
\end{array}
\begin{array}{c}
\begin{array}{cccc} a & b & c & d \end{array} \\
\begin{bmatrix}
\infty & 5 & 1 & \infty \\
5 & \infty & 7 & 4 \\
1 & 7 & \infty & 2 \\
\infty & 4 & 2 & \infty
\end{bmatrix}
\end{array}
$$

Questions 14 to 17 refer to the weighted connected graph represented by the following weight matrix:

$$
\begin{array}{c}
\\ a \\ b \\ c \\ d \\ e
\end{array}
\begin{array}{c}
\begin{array}{ccccc} a & b & c & d & e \end{array} \\
\begin{bmatrix}
\infty & 4 & 5 & 9 & 18 \\
4 & \infty & 4 & 3 & 2 \\
5 & 4 & \infty & 1 & 4 \\
9 & 3 & 1 & \infty & 4 \\
18 & 2 & 4 & 4 & \infty
\end{bmatrix}
\end{array}
$$

E

14. Let T be a minimum spanning tree of the graph computed using Kruskal's algorithm. The order of edges selected by Kruskal's algorithm is 2.5

[A] *(c,d) (a,b) (b,d) (a,c)*
[B] *(c,d) (a,b) (b,d) (d,e)*
[C] *(c,d) (a,b) (b,d) (a,d)*
[D] *(c,d) (a,b) (b,d) (b,c)*
[E] *(c,d) (b,e) (b,d) (a,b)*

cd ab
be
bd

B.

15. Let T be a minimum spanning tree of the graph computed using Prim's algorithm. Assume vertex a is selected first, then the order of vertices selected by Prim's algorithm is 2.5

a, b, e, d, C

[A] *a, b, d, e, c*
[B] *a, b, e, d, c*
[C] *a, c, d, b, e*
[D] *a, d, c, e, b*
[E] *a, b, d, c, e*

C

16. Assume the source vertex is *a*. Running Dijkstra's algorithm for the graph, after the termination, the labels for vertices are 2.5

[A] *a(0,-), b(2, a), c(4,a), d(7,b), e(18,a)*
[B] *a(0,-), b(2, a), c(6,b), d(9,a), e(6,b)*
[C] *a(0,-), b(4, a), c(5,a), d(6,c), e(6,b)*
[D] *a(0,-), b(2, a), c(4,a), d(9,a), e(6,b)*
[E] *a(0,-), b(2, a), c(6,b), d(7,b), e(6,b)*

C

17. Assume the source vertex is *a*. Running Dijkstra's algorithm for the graph, after termination, which one of the following could be an order of vertices selected by Dijkstra's algorithm? 2.5

[A] *a, b, e, c, d*
[B] *a, b, d, e, c*
[C] *a, b, c, e, d*
[D] *a, b, e, d, c*
[E] *a, d, b, e, c*

	a	b	c	d	e
	0	4	5	9	18
1	0	4	5	7	6
2	0	4	5	6	6
3	0	4	5	6	6

Answer questions 18 and 19 by running Floyd's Algorithm for the All-Pairs Shortest-Paths Problem for the weighted graph of question 14-17 (nodes a, b, c, d, e, f are numbered as 1, 2, 3, 4, 5, 6)

D

18. What is the length of the shortest path between vertices a and d with intermediate vertices numbered not higher than 2, i.e., at most contains vertex a and b? 2.5

[A] 3
[B] 5
[C] ∞
[D] 7
[E] 9

19. What is the length of the shortest path between vertices c and d with intermediate vertices numbered not higher than 3, i.e., at most contains vertices a, b and c? 2.5

[A] 3
[B] 6
[C] ∞
[D] 11
[E] 9

1

E

20. For the statements below, 2.5

   I. A problem in the class P can be solved in worst-case by a polynomial time algorithm. √
   II. A problem in the class NP can be solved by a non-polynomial time algorithms X
   III. A problem in the class NP can be verified in polynomial time √
   IV. Finding minimum spanning tree (MST) in a weighted undirected graph is an P Problem √
   V. 0/1 Knapsack problem is an NP-Complete Problem

   Which one of the following is correct?

   [A] I, IV and V are true, II and III are false X
   [B] I, II, IV and V are true but III is false X
   [C] I and II are false but III, IV and V are true
   [D] II, IV and V are true but I and III is false X
   [E] None of the above

C

21. Assume that all text and patterns consists of letters in A, B, C, D. The value of T in the shift table built by the Horspool's Algorithm for the pattern DCCDADDC is 2.5

   ~~[A]~~ t(A) = 4, t(B)=9, t(C)=6, t(D) =4
   ~~[B]~~ t(A) = 4, t(B)=9, t(C)=1, t(D) =2
   [C] t(A) = 3, t(B)=8, t(C)=5, t(D) =1
   ~~[D]~~ t(A) = 4, t(B)=8, t(C)=1, t(D) =2
   [E] t(A) = 4, t(B)=9, t(C)=1, t(D) =1

Questions 22 to 24 refer to the following Knapsack problem: given the following instance of the 0/1 Knapsack problem

item	weight	value
1	2	\$12
2	1	\$10
3	3	\$20

The Knapsack Capacity W=3

Let V[i, j] be the value of the most valuable subset of the first i items that fit into the Knapsack of capacity j. Then V[i, j] can be recursively defined as follows:

$$V[i,j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \max\{V[i-1,j],\quad v_i + V[i-1, j-w_i]\} & \text{if } j - w_i \geq 0 \\ V[i-1,j] & \text{if } j - w_i < 0 \end{cases}$$

For the above instance, the following is an incomplete table for V[i, j] (i=0, 1, 2, 3; j=0, 1, 2, 3)

	capacity j			
Item i	0	1	2	3
0	0	0	0	0
w_1=2, v_1=12 1	0	0	12	12
w_2=1, v_2=10 2	0	10	12	22
w_3=3, v_3=20 3	0	10	12	22

A.

22. What is the value of $V[1, 2]$? **2.5**

[A] 12
[B] 10
[C] 22
[D] 0
[E] 24

C

23. What is the value of the most valuable subset? **2.5**

[A] 12
[B] 10
[C] 22
[D] 0
[E] 24

A

24. What is the value of the most valuable subset if the capacity of the knapsack is 2? **2.5**

[A] 12

[B] 10

[C] 22

[D] 0

[E] 24

PART B (40 marks)

Question 1(27 marks)
Consider the following problem. Given an array A consisting of n distinct integers A[1], … A[n]. It is known that there is a position p (1≤ p ≤n), such that A[1], .., A[p] is in increasing order and A[p], A[p+1], …, A[n] is in decreasing order.

1. Write a brute force algorithm to find the position p. What is the time complexity of your algorithm? **5**

2. Devise a "divide and conquer" algorithm to find the position p. **8**

3. Set up a recurrence relation for the number of comparisons made by your algorithm and explain it. **7**

4. Based on the recurrence relation, show the complexity of your algorithm in big-O notation and prove it using either the iterative method or the substitution method, i.e., Mathematical Induction (for simplicity, you can assume that $n = 2^k$). **7**

Question II (13 marks)

1. Briefly describe the idea of the polynomial time reduction. Explain how to use it to prove a problem is NP-complete. **5**

2. 4-SAT Problem: for a Boolean formula in CNF in which each clause has exactly 4 literals, determine if there is an assignment of Boolean value to its variables so that the formula evaluates to true? (i.e., the formula is satisfiable). Prove 4-SAT Problem is NP-Complete. **8**

END OF THE PAPER

Q1.1. Algorithm BruteForce (A[1⋯n])
//Input: An array A[1⋯n]
// Output: position p
for i←1 to n-1 do
if A[i] > A[i-1] then
return i

O(n)

2. Algorithm DivideConquer (A[1⋯n], l, r)
//Input: An array A[1⋯n], 0, n
// Output: position p
if l=r then
return l
mid = ⌊(l+r)/2⌋
if A[mid] < A[mid+1] then
return DivideConquer (A[1⋯n], mid+1, r)
else
return DivideConquer (A[1⋯n], l, mid)

3. $T(n) = \begin{cases} O(1) & n=1 \\ T(\frac{n}{2})+1 & n>1 \end{cases}$

4. $O(\log n)$