No. of Pages	**4**
No. of Questions	**4**

BRAC University

Department of Computer Science and Engineering

FINAL EXAMINATION SPRING 2022

CSE 220: Data Structures

Total Marks: 55 Time Allowed: 1 hour 40 minutes

- Answer all **Four (4)** questions
- Figure in bracket [] next to each question indicates marks for that question
- Please read all the Questions carefully
- Understanding the Question is a part of exam

Question 1 (Marks 15) [CO1, CO2]:

A. Given an array which contains distinct integer values sorted in ascending order, your task is to **complete** numRotation function which takes an array as parameter and returns the number of times that the array has been rotated. You have to modify the **Binary Search algorithm** to solve this problem.
[Assume there won't be any duplicate values in the array and the rotation follows an anti-clockwise direction.] [8 Marks]

[Note: You are not allowed to use any built-in function other than **len()** for python**.**]

Sample Input	**Output**
a = [20, 30, 40, 50, 60, 10] **numRotation(a)**	**5**
a = [-2, -1, 3, 6, 8, 10, 12] **numRotation(a)**	**0**
a = [5, 6, 7, 8, 1, 2, 3, 4] **numRotation(a)**	**4**

Python Notation:

```
def numRotation(a):
        # To do
```

Java Notation:

```
public int numRotation(int [] a) {
        // To Do
}
```

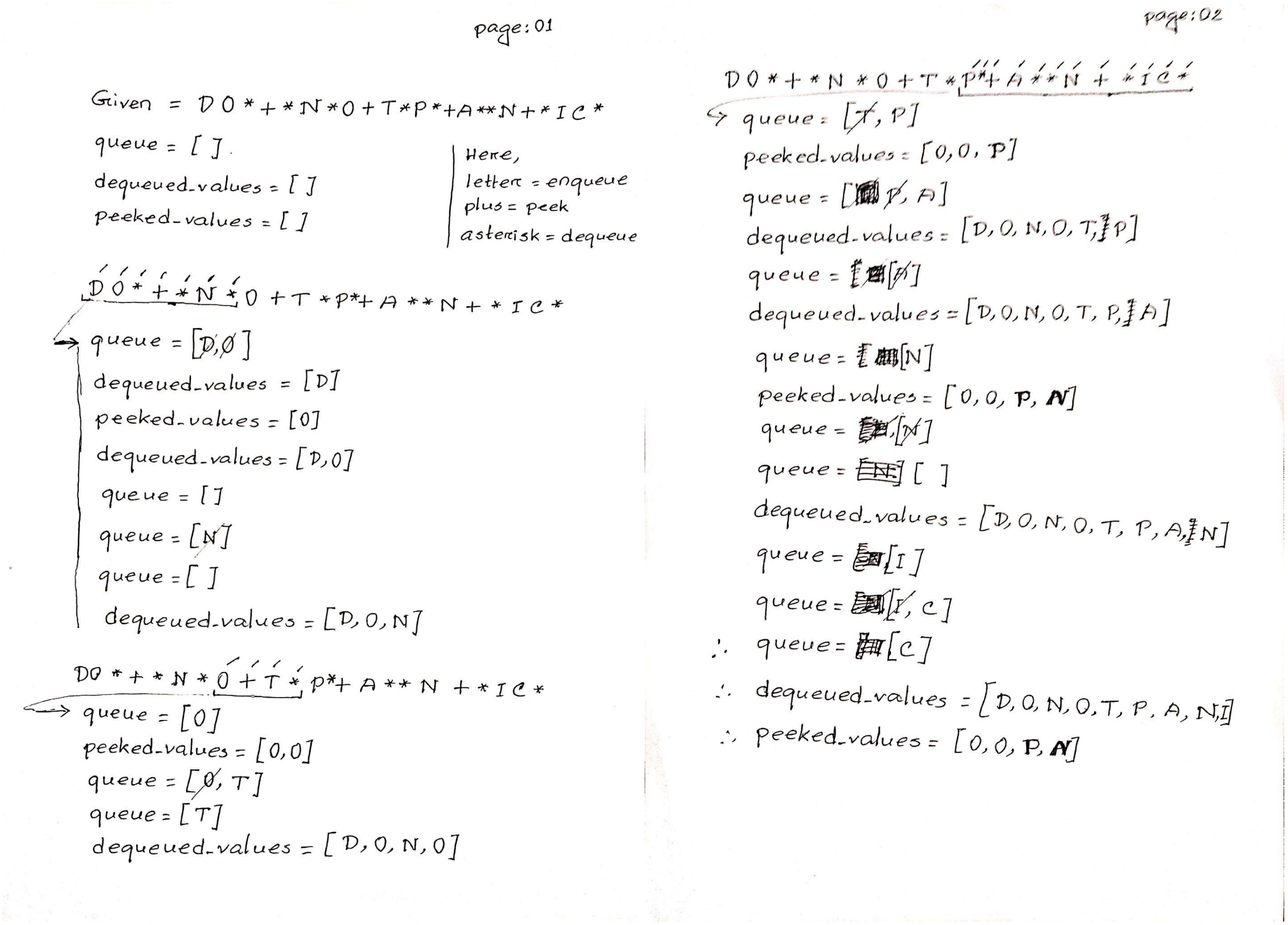
Answer:

```
def numRotation(a):
    left = 0
    right = len(a)-1
    while left <= right:
        if a[left] <= a[right]:
            return left
        mid = (left + right) // 2
        next = (mid + 1) % len(a)
        prev = (mid - 1 + len(a)) % len(a)
        if a[mid] <= a[next] and a[mid] <= a[prev]:
            return mid
        elif a[mid] <= a[right]:
            right = mid - 1
        elif a[mid] >= a[left]:
            left = mid + 1
    return -1
```

B. If a letter means enqueue, plus means peek and an asterisk means dequeue in a sequence, Draw the figure of the queue while you carry out the enqueue, dequeue and peek operations on the following sequence:

D O * + * N * O + T * P * + A * * N + * I C *

Write down the dequeued and peeked values. Show your work elaborately. [7 Marks]



page:01

Given = D O * + * N * O + T * P * + A ** N + * I C *

queue = [ ].

dequeued_values = [ ]

peeked_values = [ ]

Here,
letter = enqueue
plus = peek
asterisk = dequeue

D O * + * N * O + T * P * + A ** N + * I C *

queue = [~~D~~,~~0~~]

dequeued_values = [D]

peeked_values = [0]

dequeued_values = [D,0]

queue = [ ]

queue = [~~N~~]

queue = [ ]

dequeued_values = [D, 0, N]

D0 * + * N * 0 + T * P * + A ** N + * I C *

queue = [0]

peeked_values = [0,0]

queue = [~~0~~, T]

queue = [T]

dequeued_values = [D, 0, N, 0]

page:02

D 0 * + * N * 0 + T * P * + A ** N + * I C *

queue = [~~T~~, P]

peeked_values = [0, 0, P]

queue = [~~P~~, A]

dequeued_values = [D, 0, N, 0, T, P]

queue = [~~A~~]

dequeued_values = [D, 0, N, 0, T, P, A]

queue = [N]

peeked_values = [0, 0, P, N]

queue = [~~N~~]

queue = [ ]

dequeued_values = [D, 0, N, 0, T, P, A, N]

queue = [I]

queue = [~~I~~, C]

∴ queue = [C]

∴ dequeued_values = [D, 0, N, 0, T, P, A, N, I]

∴ peeked_values = [0, 0, P, N]

Question 2 (Marks 15) [CO4]:

Suppose you are designing an ATM machine's software. If any user gives the exact reverse of his or her PIN number, it means that the user is in trouble and the system will automatically notify the nearest police station. Your task is to **complete** the following **recursive function checkReverse** which will take two strings (actual password and given password) and an index number as parameters and check whether the second string is exactly the reverse of the first string or not. If it is exactly the reverse, then your function should return True else it should return False. **You cannot change the number of parameters of the given function.**

[Note: You are not allowed to use any built-in function other than **len()** for python.]

Sample Input	Output
print(checkReverse("1234", "4321",0))	**True**
print(checkReverse("1234", "4322",0))	**False**

print(checkReverse(“1234”, “432”,0))	**False**

```
[Python Syntax]
def checkReverse(st1, st2, idx):
          # To do

[Java Syntax]
public boolean checkReverse(String st1, String st2, int idx) {
          // To Do
}
```

Answer:

```
def checkReverse(st1, st2, idx):
  if len(st1) != len(st2):
    return False
  if idx < len(st1):
    if st1[0 + idx] == st2[len(st2) - 1 - idx]:
      return checkReverse(st1, st2, idx + 1)
    return False
  return True
```

Question 3 (Marks 10) [CO5]:

Consider the following sequence of values:

54362, 34521, 3233, 43189, 2275, 1, 672, 8892

The values above are inserted into a hashtable using the conditions below:

i) The hash functions H(k) is calculated the following way where k is the element to be hashed:

```
If k is odd:
   H(k)= number of even digits in k
else:
   H(k)= number of odd digits in k
```

ii) Linear probing is used in case of collision.

a) What is the minimum length of the hashtable needed to accommodate the values above based on the hashing conditions? Give reasons for your answer **[2]**

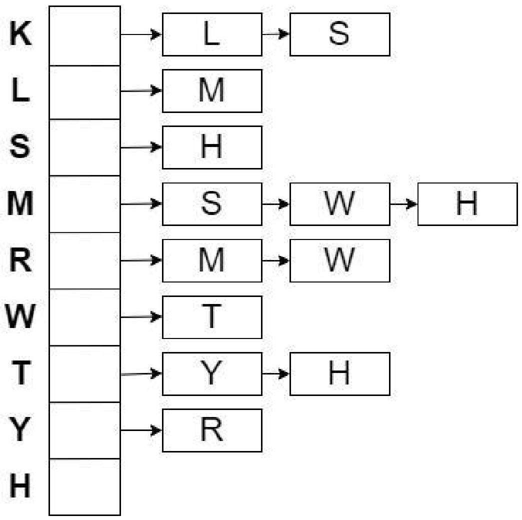
b) Using the answer found in part a) **show** the contents of the hashtable after the sequence of insertions of the values above. **[8]**
For example, the value 54362 is even, so we need to count the odd digits in it. Since there are two odd digits (5 and 3), this element should be inserted in index 2.

Question 4 (Marks 15) [CO2, CO5]:

A. Given the array representation of a binary tree: [null value means the node is empty]
[null, 12, 15, 3, 25, 13, null, 23, null, -2, null, 0, null, null, -5, -9]

a) Draw the binary tree. **[2.5]**
b) **Write** the post order and pre order traversal sequence of the tree. **[3]**
c) Use the post order traversal sequence in part b to insert the elements in that order in an initially empty binary search tree, and **show** the resulting binary search tree. Note: Consider the first element of the post order sequence as the root. **[2.5]**
d) Perform the following operations step by step on the Binary Search Tree you created in part c. **[3]**
   i. Delete node -2 with the help of its successor.
   ii. Delete node 13 with the help of its predecessor.

B. Consider the adjacency list:



i. **Convert** the adjacency list to its equivalent graph. **[2]**
ii. **Construct** the equivalent adjacency matrix representation of the graph. **[2]**

Bonus (Marks 5) [CO1, CO2]:

Write a function/method that takes the root of a Binary Tree as a parameter and counts the total number of nodes containing even values in the given binary tree. Additionally, your function/method should also print the leaf nodes of the given binary tree. Consider, the **Node** class for Binary Tree and Binary Tree class and **constructors** are already defined.

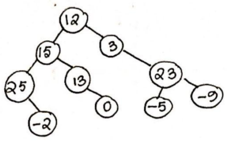
Sample Input:

```
          13
        /    \
       2      17
     /  \    /  \
    4    8  7    5
```

Sample Output:
Total Number of Nodes with even values: 3
Leaf Nodes: 4 8 7 5

Question 4 (A)

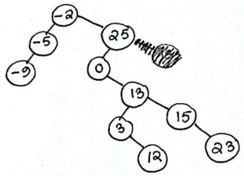
a.



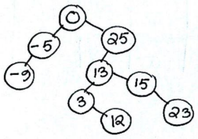
b. Post Order: -2 25 0 13 15 -5 -9 23 3 12

Pre Order: 12 15 25 -2 13 0 3 23 -5 -9

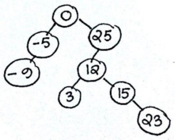
c.



d. i) successor of -2 → 0

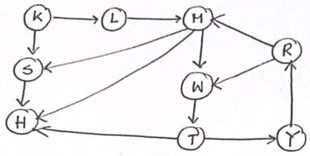


ii) predecessor of 13 → 12



B.

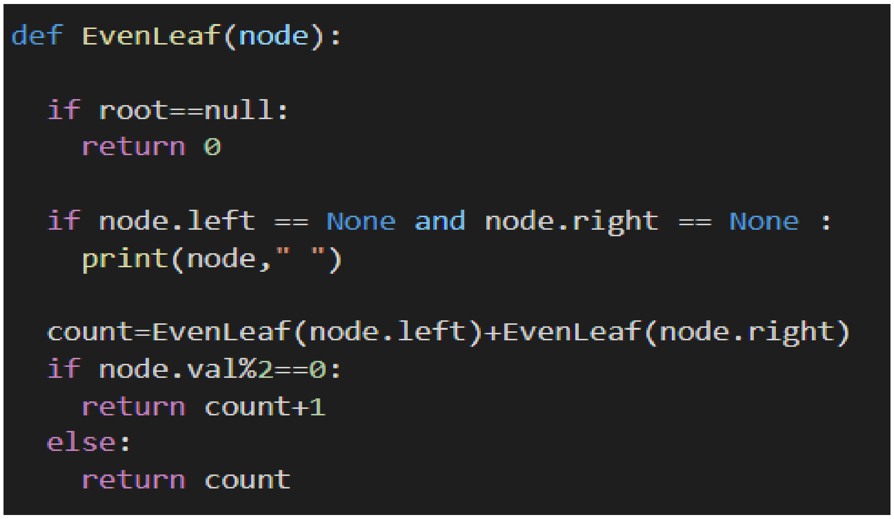
i



ii

	K	L	S	M	R	W	T	Y	H
K	0	1	1	0	0	0	0	0	0
L	0	0	0	1	0	0	0	0	0
S	0	0	0	0	0	0	0	0	1
M	0	0	1	0	0	1	0	0	1
R	0	0	0	1	0	1	0	0	0
W	0	0	0	0	0	0	1	0	0
T	0	0	0	0	0	0	0	1	1
Y	0	0	0	0	1	0	0	0	0
H	0	0	0	0	0	0	0	0	0

Bonus:



```
def EvenLeaf(node):

  if root==null:
    return 0

  if node.left == None and node.right == None :
    print(node," ")

  count=EvenLeaf(node.left)+EvenLeaf(node.right)
  if node.val%2==0:
    return count+1
  else:
    return count
```

THE END

1. the basic formula of harmonic sum of n = $1+\frac{1}{2}+\frac{1}{3}+$............$+\frac{1}{n}$.

   So, you have a task to help Adam by writing a **recursive program** in python to calculate the **harmonic sum** of any positive integer, n.

   python:

```
def harmonic_sum(n):
    If n==1:
        Return 1
    Else:
        Return 1/n + harmonic_sum(n-1)
```

2. In a sequence, an **alphanumeric character** means to enqueue, a **_**(underscore) means to dequeue and a **:**(colon) means to peek. Now consider the following sequence:

A 9 _ : : e s X _ : _ _ : D W u _ : e K : _ _ : L _ _ :

A queue has been implemented by a circular array of capacity = 8 and front = 2.

a) For the above mentioned queue, write down the enqueue and dequeue method. (5)

python:

```
 def enqueue(cir_arr, front=2, size, element):
        rear=(front+size-1)%8
        rear=(rear+1)%size
        circ_arr[rear]=element
def dequeue(cir_arr, front=2, size):
        temp=circ_arr[front]
        circ_arr[front]=0
        front=(front+1)%size
        Return temp
```

b) Simulate the operations of the given sequence in the above mentioned queue. You need to show the array change and also the enqueued, dequeued or peeked elements at each step.

(5 marks)

0	1	2	3	4	5	6	7
		A	9				
			9				
			9				
			9	e	s	x	
				e	s	x	
				e	s	x	
					s	x	
						x	
						x	
w	u					x	d
w	u						d
w	u	e	k				d
	u	e	k	L			
			K	L			
			K	L			

Question 3: Trees & Graphs (15)

I.

a. Draw a Binary Search Tree (BST) from the following sequence:
   50, 20, 55, 60, 30, 25, 27, 109, -10, 45, 70 (3)
b. Show the output of the drawn trees using Inorder and Postorder Traversal. (2)
c. Remove 109 from the tree generated in a and draw the resultant tree (1)
d. Remove 20 from the resultant tree generated in c using Successor and draw the resultant tree (2)
e. Remove root from the resultant tree generated in d using Successor and draw the resultant tree (2)

II. Draw a directed graph from the following Adjacency Matrix. (3 marks)

a.

	A	B	C	D	E	F
A	0	1	0	1	0	0
B	1	0	1	0	1	0
C	0	0	1	0	0	0
D	1	0	1	0	1	0
E	0	1	1	1	0	0
F	1	1	1	1	1	0

B. Construct the Adjacency List from the resultant graph from a. (2 marks)

4. Hashing(10)

Suppose you are making a hash table for the students of the CSE department at BRAC University. The student's ID consists of eight digits and these are taken as the key for your hash function. You have decided to create an array of size 2999 for your hash table and to resolve collisions you will use linear probing. Consider the hash function to be the summation of digits of the student ID mod 2999.
For example: Student ID: 20125105, hash value will be (2+0+1+2+5+1+0+5)%2999 = 16

Complete the following functions for your hash table:

1. Write a **hash** function that will generate the hash value for any given student id. (4)

```
Def hash(n):
        sum=0
        while (n != 0):

                 sum = sum + (n % 10)
                n = n//10

        index=sum %2999
        Return index
```

2. Given the array S containing some names. We want to perform Key Index Searching but as the values are Strings we first have to convert them into integers. Hence we have designed a conversion function. Let A = 1, B = 2, … Z = 26. The conversion function **conFun(element)** is defined by (summation of the integers associated with each character) % 10. If the element is ABC, **conFun(ABC)** = (1+2+3)%10, which is 6.

a. Your job is to create an array named EL which will have the same length as S. Then for every element *e* in S, generate an integer and store it in EL inside the same index number containing *e*. The first one is done for you. (4 marks)

S =

ABC	EDC	JKL	LKM	CAB	ACD	MLK	EGH	CED	IJK	BAC

EL =

6										

b. Draw the auxiliary array from the result you have obtained from part a of the question. (2 marks)

BRAC UNIVERSITY
Department of Computer Science and Engineering

Examination: Semester Final
Duration: 1 Hours 45 Minutes

Semester :Summer 2022
Full Marks: 45

CSE 220: Data Structures

Answer all of the following questions.
Figures in the right margin indicate marks.

Name: MASHRUR SAFIR SHABAB	ID: 20241037	Section: 14

1. **CO4** Adam wants to calculate the harmonic sum of any positive integer by using **recursion**. But he does not know how to **write** recursive code. He just knows that harmonic sum is the sum of reciprocals of the positive integers. 10

That means, the basic formula of harmonic sum of

$1+\frac{1}{2}$
$\frac{2+1}{2}$
$\Rightarrow 3/2$

$$n = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \ldots\ldots\ldots\ldots\ldots + \frac{1}{n}$$

So, you have a task to help Adam by writing a **recursive program** in python/java to calculate the **harmonic sum** of any positive integer, n.

python:

```
def harmonic_sum(n):
        #write code here
```

java:

```
public float harmonic_sum(int n){
        //write code here
}
```

Sample Input	Sample Output
harmonic_sum(4)	2.08333
harmonic_sum(7)	2.59286

$\frac{1}{2-1}$ $\frac{1}{1-1}$

$\frac{1}{4} + \frac{1}{3} + \frac{1}{2} + \frac{1}{1} + 0$

$\frac{1}{2-1}$
$\Rightarrow \frac{1}{1-1}$

$\frac{1}{4}$

$\frac{1}{2} + \frac{1}{2-1}$

$\frac{1}{2-1} = \frac{1}{1}$

3: CO2

→ Alphabet]
→ Number] -in

2. In a sequence, an **alphanumeric character** means to enqueue, a _(underscore) means to dequeue and a :(colon) means to peek. Now consider the following sequence:

A 9 : : e s X _ : _ _ : D W u _ : e K : _ _ : L _ _ :

A queue has been implemented by a circular array of capacity = 8 and front = 2.

1

CO1 a. For the above-mentioned queue, **complete** the enqueue and dequeue method. 3+2

python:

```
def enqueue(cir_arr, front=2, size, element):

        # Write your code

def dequeue(cir_arr, front=2, size):

        # Write your code
```

Java:

```
public void enqueue(char[] cir_arr, int front, int size, char
element){

        // Write  your code

}

public char dequeue(char[] cir_arr, int front, int size){

        // Write  your code

}
```

CO3 b. **Simulate** the operations of the given sequence in the above-mentioned queue. You need to show the array change and also the enqueued, dequeued or peeked elements at each step 5

BRAC UNIVERSITY

Inspiring Excellence

Name

Co

Se

3.
CO2

a. Suppose you are making a hash table for the students of the CSE department at BRAC University. The student's ID consists of eight digits and these are taken as the key for your hash function. You have decided to create an array of size 2999 for your hash table and to resolve collisions you will use linear probing. Consider the hash function to be the summation of digits of the student ID mod 2999. 4

For example: Student ID: 20125105, hash value will be (2+0+1+2+5+1+0+5)%2999 = 16

Create a **hash** function that will generate the hash value for any given student id.

b Given the array S containing some names. We want to perform Key Index Searching but as the values are Strings we first have to convert them into integers. Hence we have designed a conversion function. Let A = 1, B = 2, … Z = 26. The conversion function **conFun(element)** is defined by (summation of the integers associated with each character) % 10. If the element is ABC, **conFun(ABC)** = (1+2+3)%10, which is 6.

12%10

I. Your job is to **generate** an array named EL which will have the same length as S. Then for every element *e* in S, generate an integer and store it in EL inside the same index number containing *e*. The first one is done for you. 4

5+4+3 10+11+12 12+11+13 3+1+2 1+3+4 13+12+11 5+7+8 3+5+4 9+10+11 2+1+3

S =

ABC ✓	EDC ✓	JKL ✓	LKM ✓	CAB ✓	ACD ✓	MLK ✓	EGH ✓	CED ✓	IJK	BAC

EL =

6	2	3	6	6	8	6	0	2	0	6

$10 \overline{)12}$ 1, 10, 2

$10 \overline{)9}$ 0, 0, 9

$10 \overline{)33}$ 3, 30, 3

$10 \overline{)36}$ 3, 30, 6

II. **Show** the auxiliary array from the result you have obtained from part ~~I~~ (i) of the question. 2

$10 \overline{)6}$

$10 \overline{)20}$ 2, 20, 0

$10 \overline{)12}$ 1, 10, 2

$10 \overline{)30}$ 3, 30, 0

A → 1
B → 2
C → 3
D 4
E 5
F 6
G 7
H 8
I 9
J 10
K 11
L 12
M 13
N 14

O 15
P 16
Q 17
R 18
S 19
T 20
U 21
V 22
W 23
X 24
Y 25
Z 26

4. CO2

a. i. Draw a Binary Search Tree (BST) from the following sequence: 50, 20, 55, 60, 30, 25, 27, 109, -10, 45, 70 — 3

ii. **Show** the output of the drawn trees using Inorder and Postorder traversal. — 2

iii. Remove 109 from the tree generated in a and draw the resultant tree — 1

iv. Remove 20 from the resultant tree generated in c using Successor and draw the resultant tree — 2

v. Remove root from the resultant tree generated in d using Successor and draw the resultant tree — 2

b. **Construct** a directed graph from the following Adjacency Matrix — 3

	A	B	C	D	E	F
A	0	1	0	1	0	0
B	1	0	1 ✓	0	1 ✓	0
C	0	0	1 ✓	0	0	0
D	1 ✓	0	1 ✓	0	1 ✓	0
E	0	1 ✓	1 ✓	1 ✓	0	0
F	1 ✓	1 ✓	1 ✓	1 ✓	1 ✓	0

c. **Construct** the Adjacency List from the resultant graph from b. — 2

Date: 7/01/22

Time: 8.30 - 10 PM (Tentative)

Total marks: 30 (10 each question)

Question type: Code + Simulation

Previous Semester Question link

Please submit the questions by 6/01/22 for final examination

Total sets of questions: 3

Recursion (Code): [F18, N06]

Set 1:

Suppose you have to design a number calculation system. You need to write a recursive function that will take a 2 dimensional array / list(in python) of numbers. You need to find the maximum number of the array / list using a recursive function. Implement the following recursive function. You can not use any loop in the code.

Hint: You will need nested recursive functions.

```
def firstFunctionMax(arr, index1, index2, max):
        Return
Or

Public int firstFunctionMax(int[][] arr, int index1, int index2, int max) {
        Return
}
```

Sample Input	Sample Output	Explanation
[[10, 2, 13], [4, 15, 16], [7, 8, 9]]	16	Here, in the list the maximum number is 16.
[[15, 18], [4, 17]]	18	Here, in the list the maximum number is 18.

Set 2:

Suppose you have to design a number calculation system. You need to write a recursive function that will take a 2 dimensional array / list(in python) of numbers. You need to find the minimum number of the array / list using a recursive function. Implement the following recursive function. You can not use any loop in the code.

Hint: You will need nested recursive functions.

```
def firstFunctionMin(arr, index1, index2, min):
        Return
```

Or

```
Public int firstFunctionMin(int[][] arr, int index1, int index2, int min) {
        Return
}
```

Sample Input	Sample Output	Explanation
[[10, 2, 13], [4, 15, 16], [7, 8, 9]]	2	Here, in the list the minimum number is 2.
[[15, 18], [4, 17]]	4	Here, in the list the minimum number is 4.

Set 3:

Suppose you have to design a number calculation system. You need to write a recursive function that will take a 2 dimensional array / list(in python) of numbers. You need to find the summation of all the numbers of the array / list using a recursive function. Implement the following recursive function. You can not use any loop in the code.

Hint: You will need nested recursive functions.

```
def firstFunctionSum(arr, index1, index2):
        Return
```

Or

```
Public int firstFunctionSum(int[][] arr, int index1, int index2) {
        Return
}
```

Sample Input	Sample Output	Explanation
[[10, 2, 13], [4, 15, 16], [7, 8, 9]]	84	Here, the summation of all the numbers is 84
[[15, 18], [4, 17]]	54	Here, the summation of all the numbers is 54

Hashing/Key-Indexing (Code): [AHR, RAK]

Set 1

1. Given the array [aC@35, bD$74, hX#21, tZ%98, k1&3P].
Let, A = 65, B = 66, C = 67 ……. Y = 89, Z = 90; a = 97, b = 98, c = 99, ……, y = 121, z = 122; @ = 64, # = 35, $ = 36, % = 37, & = 38; 0 = 0, 1 = 1, …. 9 = 9.

a. Create a **hashtable** of length 5. Using the hash function f, where
f(word) = (| summation of the values of the alphabets and symbols - product of the digits |) % length of the hashtable.
Fill up the hashtable using **Linear Probing** where necessary.

b. Write a function to create an auxiliary array of key index search that takes an array containing the (summation of the values of the alphabets and symbols) of the elements of the array given in part a.

Set 2

1. Given the array [cA@53, dB$47, xH#12, zT%89, 1K&p3].
Let, A = 65, B = 66, C = 67 ……. Y = 89, Z = 90; a = 97, b = 98, c = 99, ……, y = 121, z = 122; @ = 64, # = 35, $ = 36, % = 37, & = 38; 0 = 0, 1 = 1, …. 9 = 9.

a. Create a **hashtable** of length 5. Using the hash function f, where
f(word) = (| summation of the values of the alphabets and symbols - product of the digits |) % length of the hashtable. Fill up the hashtable using **Linear Probing** where necessary.

b. Write a function to create an auxiliary array of key index search that takes an array containing the (summation of the values of the alphabets and symbols) of the elements of the array given in part a.

Set 3

1. Given the array [De@45, Ce$58, Jy#23, uA%19, L2&7q].
Let, A = 65, B = 66, C = 67 ……. Y = 89, Z = 90; a = 97, b = 98, c = 99, ……, y = 121, z = 122; @ = 64, # = 35, $ = 36, % = 37, & = 38; 0 = 0, 1 = 1, …. 9 = 9.

a. Create a **hashtable** of length 5. Using the hash function f, where f(word) = (| summation of the values of the alphabets and symbols - product of the digits |) % length of the hashtable. Fill up the hashtable using **Linear Probing** where necessary.

b. Write a function to create an auxiliary array of key index search that takes an array containing the (summation of the values of the alphabets and symbols) of the elements of the array given in part a.

4. Given the array [CAB, ADE, ABC, BCE, BAC, CBA]. Let A = 1, B = 2, … Z = 26

   a. Create a **hashtable** of length 6. Using the hash function f, where f(word) = summation of the letters % length of the hashtable Fill up the hashtable using **Linear Probing** where necessary.

   b. Create an auxiliary array, of key index search, of the same length of the hashtable. Using the calculated values is part a, fill up AUX.

5. Given the array [CAB, ADE, ABC, BCE, BAC, CBA]. Let A = 1, B = 2, … Z = 26

   a. Create a **hashtable** of length 7. Using the hash function f, where f(word) = summation of the letters % length of the hashtable Fill up the hashtable using **Linear Probing** where necessary.

   b. Create an auxiliary array, of key index search, of the same length of the hashtable. Using the calculated values is part a, fill up AUX.

6. Given the array [CAB, ADE, ABC, BCE, BAC, CBA]. Let A = 1, B = 2, … Z = 26

   a. Create a **hashtable** of length 8. Using the hash function f, where f(word) = summation of the letters % length of the hashtable Fill up the hashtable using **Linear Probing** where necessary.

   b. Create an auxiliary array, of key index search, of the same length of the hashtable. Using the calculated values is part a, fill up AUX.

Tree & Graph (Simulation): [F06, SEJ]

SET 1:

	A	B	C	D	E	F	G	H	I	J
A	0	1	0	1	0	0	0	0	0	0
B	0	0	1	0	1	0	0	0	0	0
C	0	0	0	0	0	0	1	0	0	0
D	0	1	1	0	0	0	0	0	0	0
E	0	0	0	0	0	0	1	1	0	0
F	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	1	0	0	0	1
H	0	0	0	0	0	1	1	0	0	0
I	0	0	0	0	0	1	0	1	0	1
J	0	0	0	0	0	0	0	0	0	0

a. Given the above adjacency matrix:

i. Draw the equivalent adjacency list. [2]

ii. Draw the equivalent graph. [2]

iii. Calculate the indegree and outdegree of each vertex of the graph. [1]

b. **Given the array representation of a binary tree [null value means the node is empty]:**

[null, A, E, H, D, null, B, F, I, C, null, null, null, G, J, null]

i. Draw the binary tree. [2]

ii. Write the pre-order and post-order traversal sequence of the tree. [2]

iii. Convert the tree to a complete binary tree. [1]

SET 2:

	P	Q	R	S	T	U	V	W	X	Y
P	0	1	1	0	0	0	0	0	0	0
Q	0	0	1	0	1	0	0	0	0	0
R	0	0	0	1	0	0	1	0	0	0
S	0	0	0	0	1	1	0	1	0	0
T	0	0	0	0	0	1	0	0	0	0
U	0	0	0	0	0	0	0	0	0	0
V	0	0	0	0	0	0	0	1	1	0
W	0	0	0	0	0	0	0	0	1	1
X	0	0	0	0	0	0	0	0	0	0
Y	0	0	0	0	0	1	0	0	1	0

a. Given the above adjacency matrix:

i. Draw the equivalent adjacency list. [2]

ii. Draw the equivalent graph. [2]

iii. Calculate the indegree and outdegree of each vertex of the graph. [1]

b. Given the array representation of a binary tree [null value means the node is empty]:

[null, P, T, X, null, S, Q, U, null, null, Y, W, R, null, null, V]

i. Draw the binary tree. [2]

ii. Write the post-order and in-order traversal sequence of the tree. [2]

iii. Convert the tree to a complete binary tree. [1]

SET 3:

	Z	K	L	M	N	O	P	S	R	A
Z	0	1	0	0	0	0	0	1	0	0
K	0	0	1	0	0	1	0	0	0	0
L	0	0	0	1	1	1	0	0	0	0
M	0	0	0	0	1	0	0	0	0	0
N	0	0	0	0	0	1	1	0	1	0
O	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	1	0	1	1	0
S	0	0	0	1	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	1
A	0	0	0	0	0	0	0	0	0	0

a. **Given the above adjacency matrix:**

i. Draw the equivalent adjacency list. [2]

ii. Draw the equivalent graph. [2]

iii. Calculate the indegree and outdegree of each vertex of the graph. [1]

b. Given the array representation of a binary tree [null value means the node is empty]:

[null, Z, K, M, null, L, O, P, null, null, S, null, null, R, N, A]

i. Draw the binary tree. [2]

ii. Write the pre-order and in-order traversal sequence of the tree. [2]

iii. Convert the tree to a complete binary tree. [1]

Set Distribution	Sum of ID
1	0, 3, 6, 9, 12, 15, 18
2	1, 4, 7, 10, 13, 16

3	2, 5, 8, 11, 14, 17