SULIT

FACULTY OF COMPUTING
UTM Johor Bahru

UNIVERSITI TEKNOLOGI MALAYSIA

FINAL EXAMINATION SEMESTER I, 2022 / 2023

SUBJECT CODE	:	**SECJ2013**
SUBJECT NAME	:	**DATA STRUCTURES AND ALGORITHMS**
YEAR / COURSE	:	**2SECB, 2SECP, 2SECJ, 2SECR, 2 SECV, MJIT**
TIME	:	**3 Hours**
DATE/DAY	:	**25 JAN 2023**
VENUES	:	

INSTRUCTIONS TO THE STUDENTS:

This examination book consists of 2 parts:

Part A: 20 Objective Questions 20 marks

Part B: 5 Structured Questions 80 marks

Question 1 (Searching) – 15 marks
Question 2 (Linked List) – 20 marks
Question 3 (Stack) – 15 marks
Question 4 (Queue) – 15 marks
Question 5 (Tree) – 15 marks

ANSWER ALL QUESTIONS IN THE ANSWER BOOKLET PROVIDED.

Name	
Identity card / Matric Number	
Name of Lecturer	
Subject Code and Section	

This examination book consists of **18** printed pages excluding this page.

PART A – OBJECTIVE QUESTIONS **[20 marks]**

Part A consists of 20 objective questions. Each question carries 1 mark.
Choose the <u>correct answer</u> and write your answer in the test booklet.

1. Which of the following is **FALSE** about searching algorithm?
   a. Binary search begins by examining the middle value on the array.
   b. Finding element on unsorted array using Sequential Search faster than Binary Search.
   c. Unsorted data can be used to implement improved Sequential Search.
   d. The efficiency of Sequential Search and Binary Search is same for worst case.

2. Which of the following is **TRUE** in Binary Search algorithm if the element being searched was not found?
   a. The first index value will be returned.
   b. The middle index value will be returned.
   c. The index with the last index value will be returned.
   d. The index with default value which is -1 will be returned.

3. Which of the following is **TRUE** when comparing the search key and the element at the middle of the list in Binary Search algorithm?

   i. When middle element is equal to search key, the middle index will be returned as index value.
   ii. If the search key is smaller than the middle element, search will be focused on the elements between the first element to the element before the middle element.
   iii. When middle element is equal to search key, search will be focused on the elements from the right until the middle element
   iv. If the value is bigger, search will only be focused on the elements between the second MIDDLE element to the first MIDDLE element.

   a. i, ii, and iii
   b. ii, iii, and iv
   c. i, ii, iii, and iv
   d. i, ii, and iv

4. Deleting unsorted data for circular doubly-linked list from the end required _____ complexity time.
   a. O(1) → have back pointer
   b. O(n) → singly linked list, circular
   c. O(n*n)
   d. O(logn)

5. Deletion a node from the beginning of doubly linked list required the modification of _____

   a. two pointers
   b. one pointer
   c. three pointer
   d. four pointer

6. Which of the following **TRUE** about array compared to linked list?
   i. Fast indexing
   ii. Grow dynamically without size limitations
   iii. It required checking whether it is full or not before inserting a new element
   iv. No need large memory space per element

   a. i and iv
   b. i and ii
   c. ii and iii
   d. iii and iv

7. If the elements added in a stack start with 'X', 'Y', '11', and '22', so what would be the order for the removal?
   a. 2211YX
   b. XY1122
   c. 1122XY
   d. X11Y22

8. Which one of the following is **FALSE** about the stack?
   i. It could be implemented using arrays.
   ii. It follows FIFO.
   iii. The top of the stack contains the first inserted element.
   iv.The elements in the stack are stored sequentially.

   a. i & iv
   b. i & iii
   c. ii & iii
   d. iii & iv

9. Choose the correct final output for the following sequence of stack operations?
   push(1)
   pop
   push(2)
   push(3)
   pop
   pop
   push(6)

   a. 1 2 3 6
   b. 2 3 6
   c. 2 3
   d. 6

10. Which of the following is the prefix form of E + H * T expression?
   a. (H*T)+E
   b. +EH*T
   c. EHT+*
   d. +E*HT

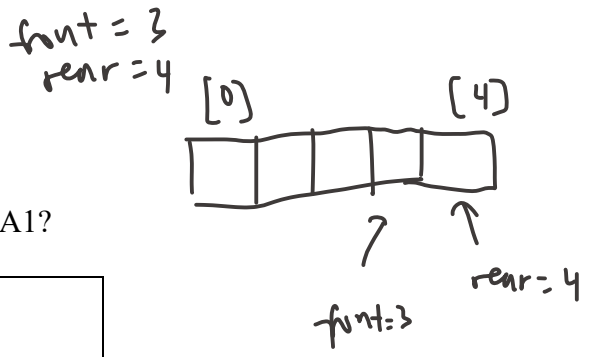
11. Which is the correct operations used to insert and delete items from the Stack?
   a. Push and pop
   b. Add and remove
   c. Enqueue and peek
   d. Enqueue and dequeue

12. If the elements P,Q,R,S are placed in a queue and are deleted one by one, in what order will they be deleted?
   a. PQRS
   b. SRQP
   c. PRQS
   d. SRQP

13. Which of the following is NOT the application of the Queue data structure?
   a. Call customer service centre
   b. Checkout at store
   c. Checkout at Car wash
   d. Web Page Browsing History

14. In a circular queue implementation using array of size 5, the array index starts with 0 where front and rear values are 3 and 4 respectively. Determine the array index at which the insertion of the next element will take place.

a. 5
b. 0
c. 1
d. 2



15. Which statement is **TRUE** for tree in Figure A1?

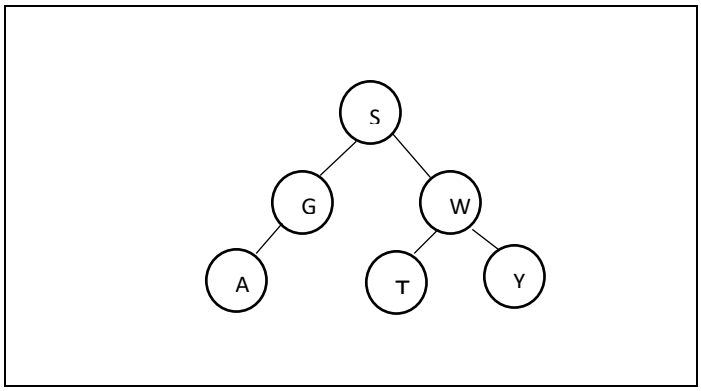


Figure A1

i. The tree is complete ✗
ii. The tree is full ✗
iii. The tree is balanced ✓
iv. The tree is binary tree ✓

a. i and ii b. ii, iii and iv c. iii and iv d. iv only

16. The following numbers are inserted into an empty binary search tree in the given order: 11, L, M, N, S, T, Y. What is the height of this binary search tree?

a. 5 b.4 c.3 d.2

17. Based on Figure A1, how many leaves node?

a. 2 b. 5
c. 3 d. 4

18. A binary tree is a tree in which _____.
    a. Every node must have two children
    b. Every node must have at least two children
    c. Height of the tree is two
    d. No node can have more than two children

inorder : H, D, I, B, E, A, F, C, J, G, K

left root right

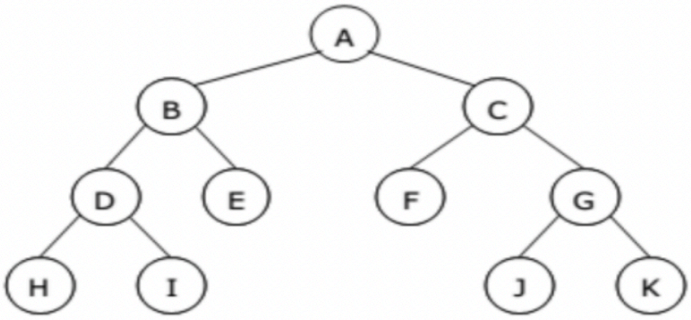


Figure B1

root left right

19. What is the **preorder** for binary tree in Figure B1?
    a. A, B, C, D, E, F, G, H, I, J, K
    b. A, B, D, H, I, E, C, F, G, J, K
    c. H, I, D, E, B, F, J, K, G, C, A
    d. H, D, I, B, E, A, F, C, J, G, K

A, B, D, H, I, E, C, F, G, J, K

left right root

20. What is the **postorder** for binary tree in Figure B1?
    a. H, I, D, E, B, F, J, K, G, C, A
    b. H, D, I, B, E, A, F, C, J, G, K
    c. A, B, C, D, E, F, G, H, I, J, K
    d. A, B, D, H, I, E, C, F, G, J, K

H, I, D, E, B, F, J, K, G, C, A

PART B - STRUCTURED QUESTIONS

[80 MARKS]

Part B consists of 5 structured questions. Answer all questions in the space provided. The marks for each part of the question is as indicated.

Question 1 **[15 MARKS]**

[5]	Protein
[4]	Genomes
[3]	Enzymes
[2]	Oxygen
[1]	Actin
[0]	Metabolic

BIO_DICT

not sorted → uses sequential search

a. Given an array named BIO_DICT with the words stored as above. What is the suitable searching technique? Give the reason to justify your answer.

Sequential search because the array is unsorted. [2 marks]

b. What is the time complexity of the searching technique in 1-a?

O(1) for first element matches, [1 mark]
O(n) for element not found or last element found

c. What should be done to the BIO_DICT array in 1-a to reduce complexity of search time? Explain your answer.

Sort the array alphabetically or in order. [3 marks]
Apply binary search can reduce complexity of
O(n) to $O(\log_2 n)$

d. What is the complexity of binary search and explain.

Best case: O(1) (if search key is found in the middle array)
Worst case: O(log₂n) (if search key not found in the list/ searching size equal to 1) [2 mark]

Instead of traverse entire list, search process using sorted array and middle of the list.

Given DATA array of integers as in Figure 1, answer question (e) and (f).

index	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
	3	8	10	12	17	22	24	38	42	46

Figure 1: DATA array

e. Perform binary search for searching key 10 on the DATA array in Figure 1. Show the tracing of your search using variables LEFT, RIGHT, MIDDLE, DATA [MIDDLE] and found as shown in the table format below.

LEFT	RIGHT	MIDDLE	DATA[MIDDLE]	found
0	9	4	17	NO
0	3	1	8	NO
2	3	2	10	yes

[3 marks]

f. Perform binary search for searching key 26 on the DATA array in Figure 1. Show the tracing of your search using variables LEFT, RIGHT, MIDDLE, DATA [MIDDLE] and found as shown in the table format below.

LEFT	RIGHT	MIDDLE	DATA[MIDDLE]	found
0	9	4	17	NO
5	9	7	38	NO
5	6	5	22	NO
6	6	6	24	NO

[4 marks]

Question 2 **[20 MARKS]**

a. What type of linked list variation does not have a node with value NULL?

circular linked list [1 mark]

b. What is the advantage of linked list compared to the array?

Grow dynamically without size limitation [1 mark]

no need to check whether full/empty before inserting / deleting element

c. What is the drawback of linked list implementation?

Need large memory space no sequential access (unlike array) [1 mark]

d. Following are the codes for a linked list structure.

```
struct node {
        int data;
        node* next;
};
```

Consider the next pointer of the node. Write a function name listSize that returns the number of items in the linked list. For example, if the list consist of 8, 3, 5, and 7 then listSize() will return 4.

[3 marks]

```
int listSize() {
    node *temp = head;
    int count=0;
    while (temp != NULL) {
        count++;
        temp = temp->next; }
    return count; }
```

e. Modify the node in the linked list above, the node contains attributes name, matric_no, age, and next, which points to the next node in the linked list.

[2 marks]

```
struct node {
    char name[50];
    char matric_no[10];
    int age;
    node *next; }
```

f. Write a declaration of List class which contain an instance variable head and initial it to NULL, DisplayList, Function1, and Function2.

[3 marks]

```
class List {
    private:
        node *head = NULL;
    public:
        void DisplayList();
        node* Function1();
        void Function2(); };
};
```

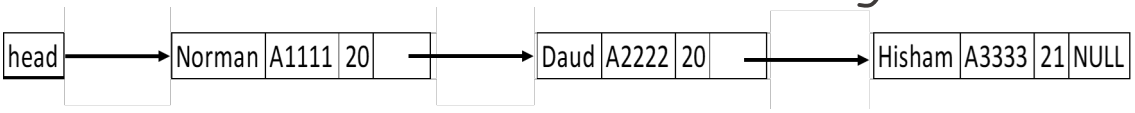


Figure 2.1: A linked list with student data.

g. Write a function named DisplayList() to accessing the node through pointer variable head and that will print the content of the node pointed by head. Assuming the list contain the data as shown in Figure 2.1. Then, the output is as shown in below:

Student Name : Norman, Matric No : A1111, Age : 20
Student Name : Daud, Matric No : A2222, Age : 20
Student Name : Hisham, Matric No : A3333, Age : 21

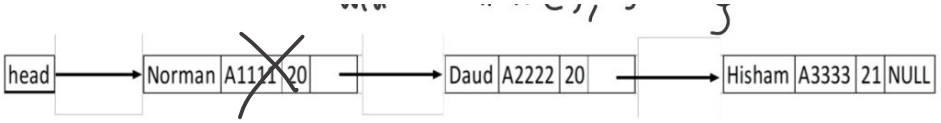
[3 marks]

```
void List::DisplayList() {
    node *temp = head;
    while (temp != NULL) {
        cout << "Student Name: " << temp->name
             << ", matric no: " << temp->matric_no
             << ", Age: " << temp->age << endl;
        temp = temp->next;
    }
```

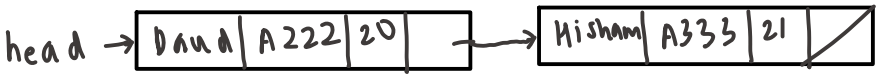
```
node* List::Function1()
{
        if (head == NULL) return NULL;
        node* temp = head;
        head = head->next;

        delete temp;
        return head;
}
```

h. Redraw the Figure 2.1 after Function1() has been executed.



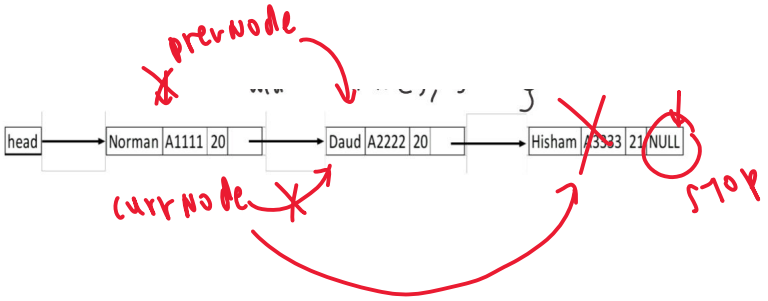
[3 marks]



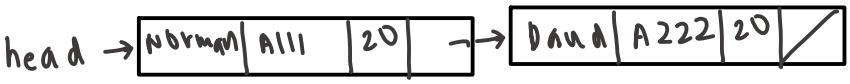

```
void List::Function2()
{
        node* prevNode = NULL;
        node* currNode = head;

        while (currNode && currNode->next != NULL)
        {
                prevNode = currNode;
                currNode = currNode->next;
        }
        prevNode->next = NULL;
}
```

i. Redraw the Figure 2.1 after Function2() has been executed.



[3 marks]



Question 3 **[15 MARKS]**

a) In the table given as follows, show step-by-step conversion of the expression to its postfix notation using stack operations. **[8 marks]**

A / (B + C * D) - E + F

A/(B+C*D) - E+F

A B C D * + / E - F +

Infix Char	Stack	Postfix
A/(B+C*D)-E+F	#	
/(B+C*D)-E+F	#	A
(B+C*D)-E+F	#/	A
B+C*D)-E+F	#/(	A
+C*D)-E+F	#/(	AB
C*D)-E+F	#/(+	AB
*D)-E+F	#/(+	ABC
D)-E+F	#/(+*	ABC
)-E+F	#/(+*	ABCD
-E+F	#/	ABCD*+
E+F	#-	ABCD*+/
+F	#-	ABCD*+/E
F	#+	ABCD*+/E-
	#+	ABCD*+/E-F
	#	ABCD*+/E-F+

b) Evaluate the following postfix expression using stack concept :

`8 48 / 36 * 2 -` op2 op op1 **[7 marks]**

Postfix	Ch	Op	Oprn1	Oprn2	Result	Stack
8 48 / 36 * 2 -						
48 / 36 * 2 -	8					8
/ 36 * 2 -	48					8 48
36 * 2 -	/	/	48	8	$\frac{1}{6}$	$\frac{1}{6}$
* 2 -	36					$\frac{1}{6}$ 36
2 -	*	*	36	$\frac{1}{6}$	6	6
-	2					6 2
	-	-	2	6	4	4

Question 4 **[15 MARKS]**

a. Describe the problem of linear queue arrays after several time of additions and removals.

Rightward drifting After a sequence additions/removal, items will drift towards the end of array. [1 mark]

b. What is the possible solution can be performed to solve rightward drift in linear queue array?

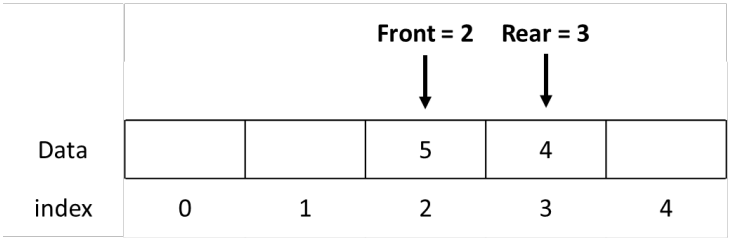
Using circular array implementation [1 mark]

c. State one disadvantage that you suggest in (i).

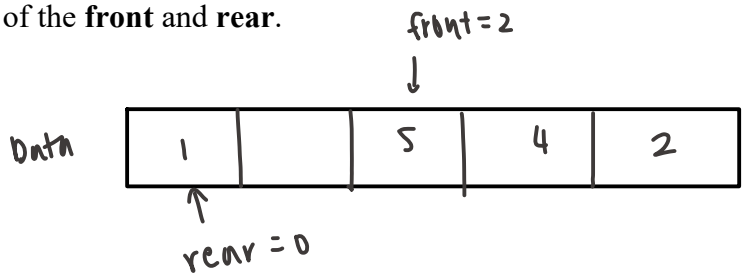
Overhead of maintaining counter / flag. [1 mark]

initial front = 0 rear = -1

d. Figure below shows a circular queue called myQueue, which fits maximum of 5 items. The current content of myQueue are as shown in the figure below. Given that **item1 = 5, item2 = 4, item3 = 2, and item4 = 1** are integer variables.

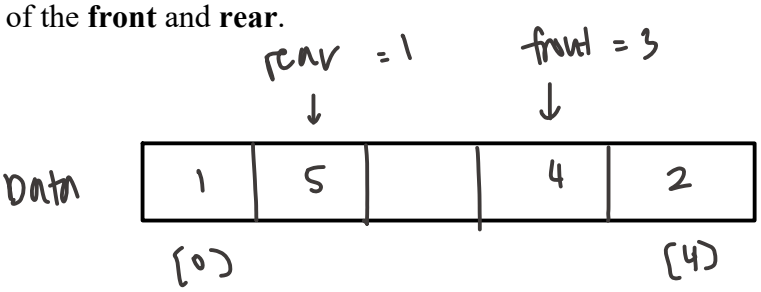


i. Redraw myQueue after **enQueue(item3)** and **enQueue(item4)** are executed. Label the correct location of the **front** and **rear**.



[2 marks]

ii. Redraw updated queue in (i) after **deQueue()** and **enQueue(item1)** are executed. Label the correct location of the **front** and **rear**.



[2 marks]

iii. After the updated queue in (ii). What is the output when the following pseudocode executed?

```
myQueue.deQueue()
myQueue.enQueue(item2 + item4)


item1 = myQueue.getFront()
item2 = myQueue.getRear()
cout << item1 << " " << item2 << endl
```

delete 4 → front = 4
[2] → 5 → rear = 2

[2 marks]

output: 2 5

e. Given Algorithm 1 – Queue Implementing Circular Linear Linked List. Answer the following questions.

```
// Algorithm 1– Queue Implementing Circular Linear Linked List
struct nodeQ {
	char item;
	nodeQ* next;
};

class queue {
private:
	nodeQ* backPtr;
	nodeQ* frontPtr;
public:
	queue() { backPtr = NULL ; frontPtr = NULL; }
	bool isEmpty() { return (backPtr == NULL && frontPtr == NULL) }
	void enqueue(char insertItem){
		nodeQ* newPtr = new nodeQ;

		if (isEmpty())
		{//empty queue
			newPtr->next = NULL ;
			newPtr->item = insertItem
			frontPtr = newPtr ,
			backPtr = newPtr ;
```

```
            }
            else {//non-empty queue
                  newPtr->next = _____NULL_____
                  newPtr->item = ____insertItem____
                  backPtr->next = ____newptr____
                  backPtr = ____newptr____
            }
      }
}
```

i. Complete the code below to initialize backPtr and frontPtr.

```
queue() { _____________________________ }
```

[1 mark]

ii. Complete the code below to check whether the queue is empty.

```
bool isEmpty() { return ___________________ }
```

[1 mark]

iii. Complete the code below to insert item into an empty queue.

```
newPtr->next = ________
newPtr->item = ____________
frontPtr = ________
backPtr = ________
```

[2 marks]

iv. Complete the code below to insert item into a non-empty queue.

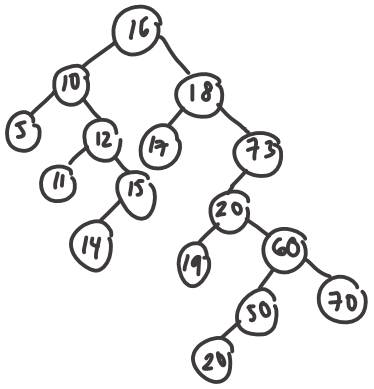
```
newPtr->next = _________________
newPtr->item = ____________
backPtr->next = ________
backPtr = ________
```

[2 marks]

Question 5 **[15 MARKS]**

a. Draw the binary search tree based on the sequence of numbers 16, 10, 18, 17, 12, 73, 20, 15, 5, 60, 19, 50, 14, 70, 11, 22. The insertion starts from number 16 to 11 in a sequential manner.

[4 marks]



b. What is the height of the binary search tree produced in Question 5-a?

[1 mark]

7

c. At which level would be considered a full binary search tree for binary search tree produced in question 5-a?

[1 mark]

level 2

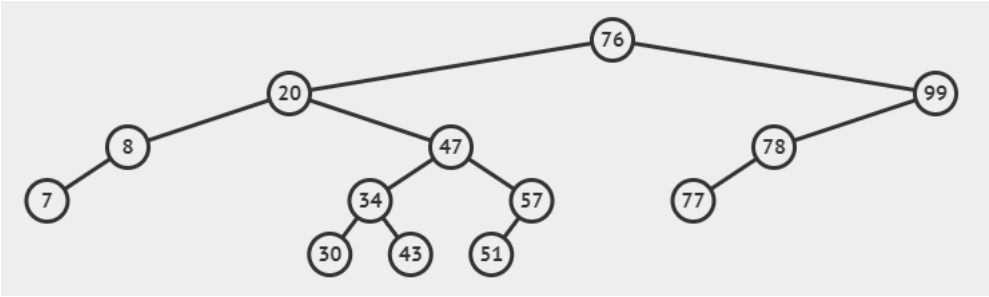
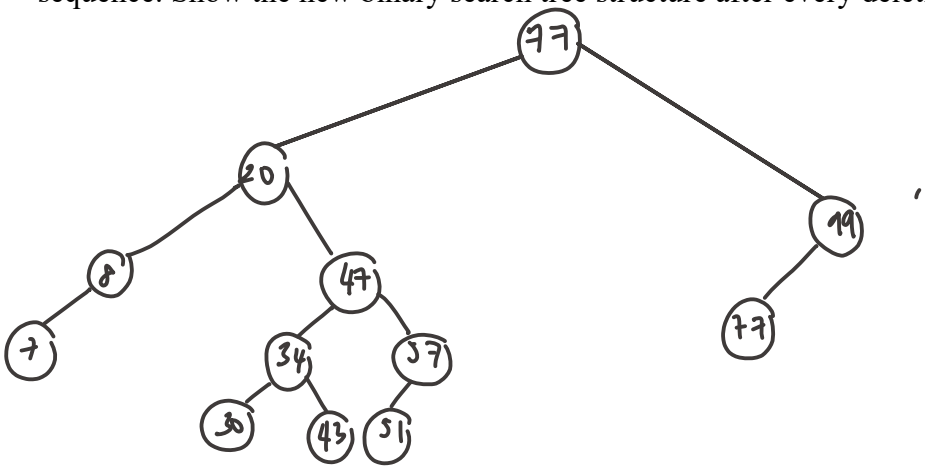
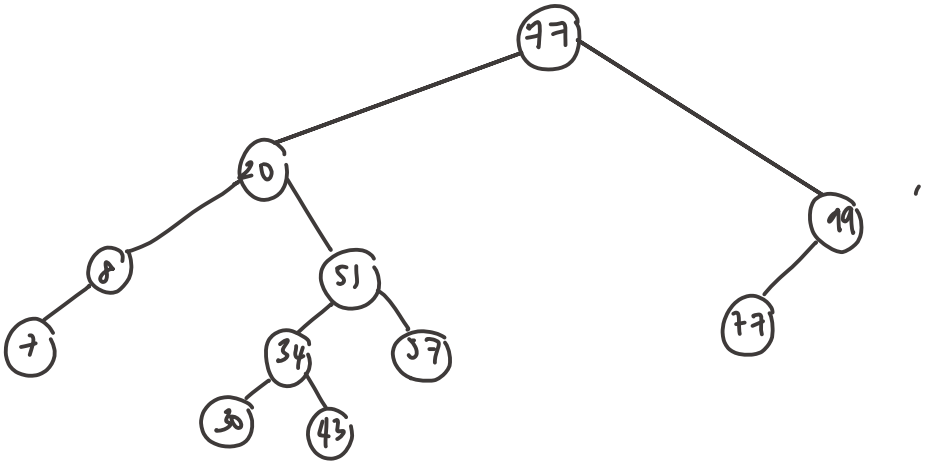


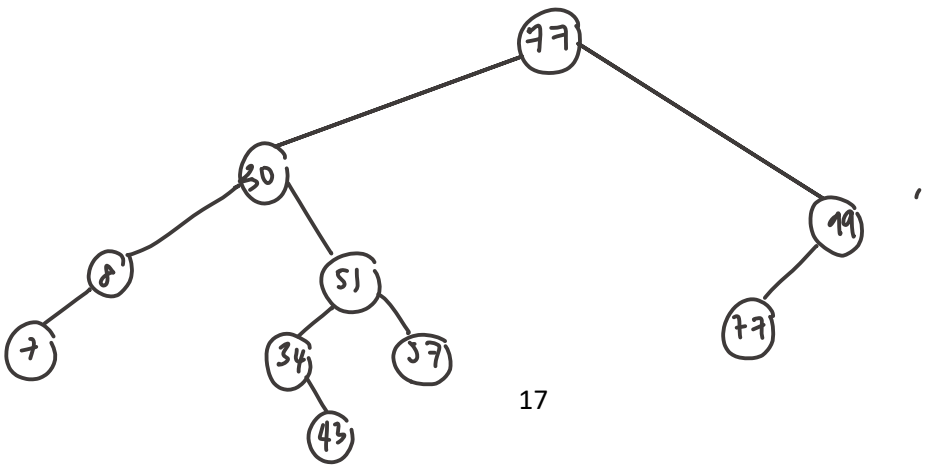
Figure 5.1: A binary search tree with root 76

d. Redraw the binary search tree in Figure 5.1 if node with values 76, 47 and 20 are deleted in sequence. Show the new binary search tree structure after every deletion.





[6 marks]



e. Give the inorder, preorder and postorder traversal on the binary search tree in Figure 5.2.

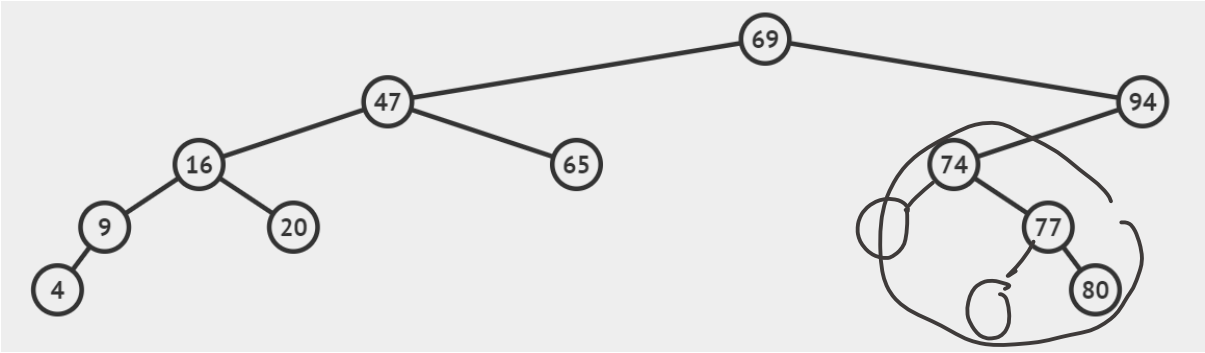


Figure 5.2: A binary search tree with root 69

[3 marks]

Inorder: 4, 9, 16, 20, 47, 65, 74, 77, 80, 94

Preorder: 69, 47, 16, 9, 4, 20, 65, 94, 74, 77, 80

Postorder: 4, 9, 20, 16, 65, 47, 80, 77, 74, 94, 69