USN:___________________

Department of Artificial Intelligence and Machine Learning

Course Code: 21AI33
Sem:III

Date: 18-Jan-2023
Duration: 90 Minutes

CIE-I

Data Structures and Data Analysis (DSDA)

Scheme and Solutions

SL. No		Questions	M	BT	CO
1		Write a complete C program to create and merge **THREE** linked lists to produce one final list in ascending order of data items. Assume the following node structure for the list;	10	03	01

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

ANS: Scheme

Correct declaration of all variables and data types	02 M
Creation of lists logic and code	02 M
Displaying the list logic and code	02 M
Merging of lists and display	04 M

ANS: Sample Solution

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
  int data;
  struct node *link;
};

struct node *listinsert(struct node *l, int x)
{
     struct node *temp, *newnode;
     newnode = (struct node*)malloc(sizeof(struct node));
     newnode->data=x; newnode->link = NULL;
     if (l==NULL)
      l=newnode;
     else
      {
        temp = l;
        while (temp->link != NULL)
        temp=temp->link; //Change the temp to next node of the list
        temp->link = newnode;  // Insert the new node in the end
      }
      return l;
}

void listdisplay(struct node *l)
{
    struct node *temp;
    printf("The list contents are\n");
    temp=l;
    while(temp)
    {
      printf("%d\n", temp->data);
      temp=temp->link;
    }
}

void main()
{
   struct node *l1, *l2, *l3, *l4, *l5,*newnode,*temp1, *temp2,*temp3;
   int x;

   l1=l2=l3=l4=l5=NULL;
   printf("Creating first list \n");
   while(1)
   {
      printf("Enter the data \n"); scanf("%d",&x);
      if (x == -1) break;
      l1=listinsert(l1, x);
    }
    printf("First list \n");
    listdisplay(l1);

    printf("Creating second list \n");
    while(1)
    {
      printf("Enter the data \n"); scanf("%d",&x);
      if (x == -1) break;
      l2=listinsert(l2, x);
    }

    printf("Second list \n");
    listdisplay(l2);

    printf("Creating Third list \n");
    while(1)
    {
      printf("Enter the data \n"); scanf("%d",&x);
      if (x == -1) break;
      l3=listinsert(l3, x);
```

```
}

printf("Third list \n");
listdisplay(l3);

// Merging First and Second List into New List
temp1=l1;
temp2=l2;
while (temp1 && temp2)
{
  if (temp1->data <= temp2->data) // First list node has lower or equal value
  {
     l4=listinsert(l4,temp1->data);
     temp1=temp1->link;
  }
  else
  {
    l4=listinsert(l4, temp2->data);
    temp2=temp2->link;
  }
}

// Merge remaining elements of the first and second lists
while (temp1)
   {
     l4=listinsert(l4,temp1->data);
     temp1=temp1->link;
   }
while (temp2)
   {
     l4=listinsert(l4,temp2->data);
     temp2=temp2->link;
   }

// Merging Third and Previously Merged List into New List

temp1=l3;
temp2=l4;
while (temp1 && temp2)
{
  if (temp1->data <= temp2->data) // First list node has lower or equal value
  {
     l5=listinsert(l5,temp1->data);
     temp1=temp1->link;
  }
  else
  {
    l5=listinsert(l5, temp2->data);
    temp2=temp2->link;
```

```
      }
    }
    // Merge remaining elements
    while (temp1)
      {
        l5=listinsert(l5,temp1->data);
        temp1=temp1->link;
      }
    while (temp2)
      {
        l5=listinsert(l5,temp2->data);
        temp2=temp2->link;
      }

    printf("List after merging \n");
    listdisplay(l5);
}
```

2 a) You are asked to convert the given infix expression to postfix using Stacks. Give the tracing by highlighting the value of TOP and Stack Contents **(Do not write C program).** (A + B) / (C - D) - (E * F)

04 02 01

ANS:

TOP	Stack Content	Output
0	(	
		A
1	+	
		AB
0		AB+
1	/	AB+
2	**(-> /**	
3	**…..**	**……**

b) Write a C program to reverse the String content using Stacks. Your program should have PUSH and POP functions. Example: Input: PEELS Output: SLEEP

06 03 01

ANS: Scheme

Correct declaration of all variables and data types	01 M
Creation of PUSH and POP functions	01 M
Input and Output	01 M
Using Stack to Reverse the input	03 M

ANS: Sample Solution

```
#include <stdio.h>
#include <string.h>
#define MAXSIZE 1000
```

```
char s[MAXSIZE]; //Stack Declaration
int top;
char str[MAXSIZE]; // Input String

void push(char c)
{
   if (top == MAXSIZE)
      printf("Stack Overflow\n");
   else
      s[++top]=c;
}

char pop()
{
   char c;
   if (top == -1)
      {
        printf("Stack Empty\n");
        return '\0';
      }
   else
   {
      c=s[top];
      top--;
      return c;
   }
}

void main()
{
    int i;
    top = -1;

    printf("Enter the String\n"); scanf("%s",&str);

    for(i=0;i<strlen(str);i++)
        push(str[i]);

    printf("Reversing the String\n");
    for(i=0;i<strlen(str);i++)
        str[i]=pop();

    str[i]='\0';
    printf("The reversed list\n");
    printf("%s",str);
}
```

3		Consider the following scenario; A theatre has 100 seats, and you are getting a series of N requests for booking them. Write a C program to process these requests using a FIFO manner using	10	03	03

a static linear queue of size 1000. Your program should take care of the following:

1. Queue Overflow and Queue Underflow conditions
2. Allot the seats, if available, based on the requested number
3. Display whether the allotment is successful or not for every request

ANS: Scheme

Correct declaration of all variables and data types	02 M
Input and Output	02 M
Requests Queue creation with overflow condition handlinig	02 M
Requests Queue deletion with underflow condition handling	02 M
Allocating/Not-allocating the seats	02 M

4 a) Complete the following C function, which is used to delete **all the nodes** in a Linked List, where the current_node initial value is the address of the starting node of a Linked List passed from the main() function. Assume the node structure has int data, struct node *link pointer fields. **(Do not write complete C program).**

04 02 01

ANS: Sample Solution

```
struct node *deleteall(struct node *current_node)
{
    if (current_node->link == NULL)
    {
          free(current_node);
          return NULL;
     }
    else
    return(deleteall(current_node->link));
}
```

b) Assume you have a double-linked list created with the following node structure;

06 03 03

```
struct node
{
   char word[80];
   struct node *llink, *rlink;
};
```

Complete the following C function, which deletes all the occurrences of a keyword from the list; Consider the starting node address as **First** and the ending node address as **Last (Do not write complete C program).**

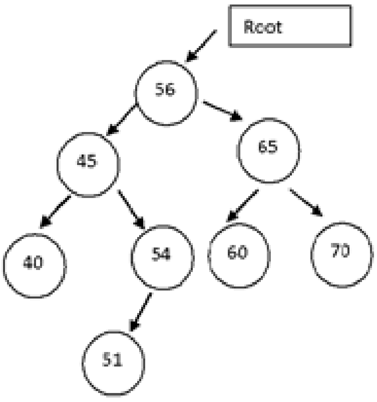
ANS: Sample Solution

```
void delete(char keyword[80])
{
  struct node *prev, *current;
   if (First==NULL && Last==NULL)
```

```
        printf(“Empty list\n”);
   else
    {
         prev=NULL;
         current=First;

     while (current !=NULL) // Process the complete list
         {
         if (strcmp(current->word, keyword)==0)
         {
           if (current == Last)  // If the keyword present in the last node of the list
           {
               Last=Last->llink;
               Last->rlink=NULL;
           }
           else
           {
            prev->rlink = current->rlink; // If the keyword is in between node
           current->rlink->llink=prev;
           }
          current->llink=current->rlink=NULL;  //  Delete  the  node  with  the
keyword
          free(current);
         }
          current=prev->rlink;
      } // end of while
  }
}
```

5	a)	Prove that the height of a binary tree with ‘n’ internal nodes is at least $\log_2$(n + 1) and at most n − 1. **ANS**: Suppose a binary tree has n nodes. There's at most 1 node (the root) at height 0, at most 2 nodes (2 children of the root) at height 1, at most 4 nodes (2 children each for the 2 children of the root) at height 2, and so on. So, for a tree with a given height *H*, the maximum number of nodes on all levels is 1+2+4+8+...+2*H*=2*H*+1−1. Therefore, if we know that there are *N* nodes, we have 2*H*+1−1≥*N*, so *H*≥log2(*N*+1)−1. This is the lower bound on height. To get the upper bound, we consider that there cannot be a node at height *H* without there being a node at height *H*−1 (except in the case of *H*=0). Therefore, if a tree has height *H*, it must have at least one node at height *H*, then a node at height *H*−1, then a node at *H*−2, all the way to 0. The number of nodes *N* therefore satisfies *N*≥*H*+1 and therefore *H*≤*N*−1. So, the overall result is:log2(*N*+1)−1≤*H*≤*N*−1	04	03	01

<table>
<tr>
<td></td>
<td>b)</td>
<td>Write a complete C Program to do the following;
1. Creating a Binary Tree like below
2. To display Binary Tree using various traversals

<b>ANS: Scheme</b>
<table><tr><td>Correct declaration of all variables and data types</td><td>01 M</td></tr><tr><td>Input</td><td>01 M</td></tr><tr><td>Creating Binary Search Tree Code</td><td>02 M</td></tr><tr><td>Traversals Codes</td><td>02 M</td></tr></table></td>
<td>06</td>
<td>02</td>
<td>03</td>
</tr>
</table>

Course Outcome	
CO1	Apply the knowledge of data structures in providing solutions to some software development requirements.
CO2	Perform data analysis of some real-world scientific/business use cases and present the analysis results.
CO3	Investigate appropriate data structures and understand requirements in solving some problems of industry and society.
CO4	Use data analysis tools to illustrate the principles of data interpretation, statistical analysis, and graphical visualizations of the datasets.
CO5	Appraise data structures and analysis knowledge to build a successful career as an AIML engineer, work in teams, and communicate their ideas effectively.

M-Marks, BT-Blooms Taxonomy Levels, CO-Course Outcomes

Marks Distribution	Particulars	CO1	CO2	CO3	CO4	CO5	L1	L2	L3	L4	L5
	Max Marks	28	--	22	--	--	--	14	36	--	--

RV College of Engineering®

Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi

Approved by AICTE New Delhi

Academic Year 2022-2023 (ODD Semester)

USN:____________________

Department of Artificial Intelligence and Machine Learning

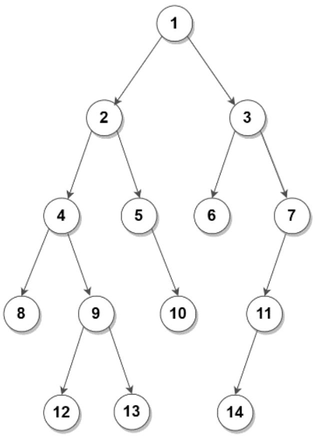
QUIZ-I

Course Code: 21AI33 **Date:** 18-Jan-2023

Sem:III **Duration:** 20 Minutes

Data Structures and Data Analysis (DSDA)

Answer all the Questions

SL. No	Questions	M	BT	CO
1	Imagine you are developing a Text Editor, Undo and Redo commands. Which data structure is suitable for this purpose and why? **ANS**: Double linked list / Stacks with Justification	02	02	01
2	Demonstrate the advantage of using Static Circular Queues against Static linear Queues (Program writing not required). **ANS**: Deletion of elements in static linear queues will create empty memory fragments. Based on the FRONT pointer value the queue status is shown as FULL.	02	01	03
3	Consider int *ptr; What does each of the following two statements do? 1. ptr = (int*) malloc(n * sizeof(int)); 2. ptr = (int*) calloc(n, sizeof(int)); **ANS**: malloc() allocates memory from the heap without any initialization, whereas calloc() does initialization of size n integers as a contiguous block.	02	02	01
4	Write the list representation and the postorder traversal of the following Binary Tree  1 2 3 4 5 6 7 8 9 10 11 12 13 14 **ANS**: List representation : 1(2(4(8,9(12,13)),5(10)),3(6,7(11(14)))) Postorder traversal: 8,12,13,9,4,10,5,2,6,14,11,7,3,1	02	02	03
5	Trace the application of Stack in the following recursive function to find the factorial, assuming an initial value of n=4.	02	02	01

```
int fact(int n)
{
  if (n == 0 or n == 1) return 1;
  return(n*fact(n-1));
}
```

ANS:

TOP	STACK CONTENT (TOP->…BOTTOM)	RETURN VALUE
-1	EMPTY	
0	4	
1	3 -> 4	
2	2 -> 3 -> 4	
		1
2	3->4	2
1	4	6
0	EMPTY	24
-1		

Marks Distribution	Particulars	CO1	CO2	CO3	CO4	CO5	L1	L2	L3	L4	L5
	Max Marks	06	---	04	--	--	02	08	--	--	--