Name of the Paper: Design Lab
Paper Code: CS 891

1. **Write a 'C' Program to create a structure of student having fields roll_no, stud_name, mark1, mark2, mark3. Calculate total marks and average marks. Arrange the records in descending order of marks.**

   Solution:

```
#include<stdio.h>
struct student
{
    int rno;
    char name[20];
    int marks[3];
    int total;
    float avg;
}stud[2];
int main()
{
    int i,j;
    struct student s;
    for(i=0;i<2;i++)
    {
        printf("Enter Record for Student-%d \n",i+1);
        printf("-----------------------------------\n");
        printf("Enter Roll-No. : ");
        scanf("%d",&stud[i].rno);
        printf("Enter Name     : ");
        scanf("%s",stud[i].name);
        stud[i].total=0;
        for(j=0;j<3;j++)
        {
            printf("Enter Marks of Subject %d : ",j+1);
            scanf("%d",&stud[i].marks[j]);
            stud[i].total=stud[i].total+stud[i].marks[j] ;
```

```
            stud[i].avg=stud[i].total/3.0;
        }
        printf("\n-----------------------------------\n");
    }
    for(i=0;i<2;i++)
    {
        for(j=i+1;j<2;j++)
        {
            if(stud[i].total<stud[j].total)
            {
                s=stud[i];
                stud[i]=stud[j];
                stud[j]=s;
            }
        }
    }
    printf("Records in Descending Order.\n (According to Total-Marks)");
    printf("\n-----------------------------------\n");
    printf("\n ROLLNO   NAME  TOTAL-MARKS  AVG\n");
    for(i=0;i<2;i++)
    {
        printf("\n %d\t %s\t %d\t %.2f",stud[i].rno,stud[i].name,stud[i].total,stud[i].avg);
    }
    return 0;
}
```

Output :

```
Enter the no of students :3
Enter Record for Student-1
---------------------------------
Enter Roll-No. : 1
Enter Name     : Prabir
Enter Marks of Subject 1 : 75
Enter Marks of Subject 2 : 56
```

```
Enter Marks of Subject 3 : 66

---------------------------------
Enter Record for Student-2
---------------------------------
Enter Roll-No. : 3
Enter Name     : Roshan
Enter Marks of Subject 1 : 67
Enter Marks of Subject 2 : 66
Enter Marks of Subject 3 : 71

---------------------------------
Enter Record for Student-3
---------------------------------
Enter Roll-No. : 4
Enter Name     : Rahul
Enter Marks of Subject 1 : 77
Enter Marks of Subject 2 : 57
Enter Marks of Subject 3 : 67

---------------------------------
Records in Descending Order.
 (According to Total-Marks)
---------------------------------

 ROLLNO   NAME  TOTAL-MARKS  AVG

 3       Roshan  204     68.00
 4       Rahul   201     67.00
 1       Prabir  197     65.67
```

2. Write a C program to delete a specific line from an existing file.

```
#include <stdio.h>
    int main() {
      FILE *fp1, *fp2;
      //consider 40 character string to store filename
      char filename[40];
      char c;
      int del_line, temp = 1;
      //asks user for file name
      printf("Enter file name: ");
      //receives file name from user and stores in 'filename'
      scanf("%s", filename);
      //open file in read mode
      fp1 = fopen(filename, "r");
      c = getc(fp1);
```

```
  //until the last character of file is obtained
  while (c != EOF)
  {
    printf("%c", c);
    //print current character and read next character
    c = getc(fp1);
  }
  //rewind
  rewind(fp1);
  printf(" \n Enter line number of the line to be deleted:");
  //accept number from user.
  scanf("%d", &del_line);
  //open new file in write mode
  fp2 = fopen("copy.c", "w");
  c = getc(fp1);
  while (c != EOF) {
    c = getc(fp1);
    if (c == '\n')
    temp++;
    //except the line to be deleted
    if (temp != del_line)
    {
      //copy all lines in file copy.c
      putc(c, fp2);
    }
  }
  //close both the files.
  fclose(fp1);
  fclose(fp2);
  //remove original file
  remove(filename);
  //rename the file copy.c to original name
  rename("copy.c", filename);
  printf("\n The contents of file after being modified are as  follows:\n");
  fp1 = fopen(filename, "r");
  c = getc(fp1);
  while (c != EOF) {
      printf("%c", c);
      c = getc(fp1);
  }
  fclose(fp1);
  return 0;
}
```

Program Output:

Enter file name:abc.txt

hi.

Hello

how are you?

I am fine

hope the same

Enter line number of the line to be deleted:4

The contents of file after being modified are as follows:

hi.

hello

how are you?

hope the same

3. **Write a C program to add two polynomials using linked list.**

```
#include <stdio.h>
#include <conio.h>
#include <malloc.h>
#include <stdlib.h>
typedef struct pnode
          {
   float coef;
           int exp;
           struct pnode *next;
          }p;
p *getnode();
//Main function starts here
```

```
void main()
        {
   p *p1,*p2,*p3;
        p *getpoly(),*add(p*,p*);
        void display(p*);
   clrscr();
        printf("\n Enter first polynomial");
        p1=getpoly();
        printf("\n Enter second polynomial");
        p2=getpoly();
        printf("\nThe first polynomial is");
        display(p1);
        printf("\nThe second polynomial is");
        display(p2);
        p3=add(p1,p2);
        printf("\nAddition of two polynomial is :\n");
        display(p3);
       }

/*Funtion to get polynomial*/
p *getpoly()
  {
        p *temp,*New,*last;
        int flag,exp;
        char ans;
        float coef;
        temp=NULL;
        flag=1;
        printf("\nenter the polynomial in descending order of exponent");
        do
                {
                 printf("\nenter the coef & exponent of a term");
                 scanf("%f%d",&coef,&exp);
                 New=getnode();
                 if(New==NULL)
                 printf("\nmemory cannot be allocated");
                 New->coef=coef;
                 New->exp=exp;
                 if(flag==1)
                        {
                                temp=New;
                                last=temp;
                                flag=0;
                         }
                 else
                        {
                                last->next=New;
                                last=New;
                         }
```

```
                    printf("\ndou want to more terms");
                    ans=getch();
                    }
                  while(ans=='y');
                  return(temp);
          }
/*Function to get the Nodes of Polynomial*/
p *getnode()
  {
          p *temp;
          temp=(p*) malloc (sizeof(p));
          temp->next=NULL;
          return(temp);
  }
/*Function to display Polynomial*/
void display(p*head)
  {
          p*temp;
          temp=head;
          if(temp==NULL)
          printf("\npolynomial empty");
          while(temp->next!=NULL)
                  {
                    printf("%0.1fx^%d+",temp->coef,temp->exp);
                    temp=temp->next;
                  }
          printf("%0.1fx^%d",temp->coef,temp->exp);
          getch();
         }
/*Function to add Polynomials*/
p*add(p*first,p*second)
  {
          p *p1,*p2,*temp,*dummy;
          char ch;
          float coef;
          p *append(int,float,p*);
          p1=first;
          p2=second;
          temp=(p*)malloc(sizeof(p));
          if(temp==NULL)
          printf("\nmemory cannot be allocated");
          dummy=temp;
          while(p1!=NULL&&p2!=NULL)
                  {
                    if(p1->exp==p2->exp)
                                    {
                                  coef=p1->coef+p2->coef;
                                  temp=append(p1->exp,coef,temp);
                                  p1=p1->next;
```

```
                        p2=p2->next;
                            }
                else
                if(p2->exp>p1->exp)
                        {
                                coef=p2->coef;
                                temp=append(p2->exp,coef,temp);
                                p2=p2->next;
                         }
                else
                if(p1->exp>p2->exp)
                        {
                                coef=p1->coef;
                                temp=append(p1->exp,coef,temp);
                                p1=p1->next;
                         }
                       }
        while(p1!=NULL)
                       {
                        temp=append(p1->exp,p1->coef,temp);
                        p1=p1->next;
                        }
        while(p2!=NULL)
                       {
                        temp=append(p2->exp,p2->coef,temp);
                        p2=p2->next;
                        }
        temp->next=NULL;
        temp=dummy->next;
        free(dummy);
        return(temp);
       }
/*Function to append the coefficients with Polynomial*/
p*append(int Exp,float Coef,p*temp)
  {
        p*New,*dum;
        New=(p*)malloc(sizeof(p));
        if(New==NULL)
        printf("\ncannot be allocated");
        New->exp=Exp;
        New->coef=Coef;
        New->next=NULL;
        dum=temp;
        dum->next=New;
        dum=New;
        return(dum);
  }
```

Output :
Enter first polynomial
enter the polynomial in descending order of exponent
enter the coef & exponent of a term5
3
do u want to more terms
enter the coef & exponent of a term
4
1

do u want to more terms
Enter second polynomial
enter the polynomial in descending order of exponent
enter the coef & exponent of a term
4
2

do u want to more terms
enter the coef & exponent of a term2
1

do u want to more terms
enter the coef & exponent of a term7
0

do u want to more terms
The first polynomial is5.0x^3+4.0x^1
The second polynomial is4.0x^2+2.0x^1+7.0x^0
Addition of two polynomial is :
5.0x^3+4.0x^2+6.0x^1+7.0x^0

4. **Write a C program to reverse a Singly Linked List.**

```
#include <stdio.h>
#include <stdlib.h>


/* Structure of a node */
struct node {
    int data; //Data part
    struct node *next; //Address part
}*head;


/* Functions used in the program */
void createList(int n);
void reverseList();
void displayList();
```

```
int main()
{
    int n, choice;

    /*
     * Create a singly linked list of n nodes
     */
    printf("Enter the total number of nodes: ");
    scanf("%d", &n);
    createList(n);

    printf("\nData in the list \n");
    displayList();

    /*
     * Reverse the list
     */
    printf("\nPress 1 to reverse the order of singly linked list\n");
    scanf("%d", &choice);
    if(choice == 1)
    {
        reverseList();
    }

    printf("\nData in the list\n");
    displayList();

    return 0;
}


/*
 * Create a list of n nodes
 */
void createList(int n)
{
    struct node *newNode, *temp;
    int data, i;

    if(n <= 0)
    {
        printf("List size must be greater than zero.\n");
        return;
    }

    head = (struct node *)malloc(sizeof(struct node));

    /*
     * If unable to allocate memory for head node
```

```
    */
    if(head == NULL)
    {
       printf("Unable to allocate memory.");
    }
    else
    {
            /*
        * Read data of node from the user
        */
       printf("Enter the data of node 1: ");
       scanf("%d", &data);

            head->data = data; // Link the data field with data
       head->next = NULL; // Link the address field to NULL

       temp = head;

       /*
                    * Create n nodes and adds to linked list
        */
       for(i=2; i<=n; i++)
       {
          newNode = (struct node *)malloc(sizeof(struct node));

                            /* If memory is not allocated for newNode */
          if(newNode == NULL)
          {
             printf("Unable to allocate memory.");
             break;
          }
                            else
          {
             printf("Enter the data of node %d: ", i);
             scanf("%d", &data);

             newNode->data = data; // Link the data field of newNode with data
                                newNode->next = NULL; // Link the address field of newNode
with NULL

             temp->next = newNode; // Link previous node i.e. temp to the newNode
             temp = temp->next;
          }
       }

       printf("SINGLY LINKED LIST CREATED SUCCESSFULLY\n");
    }
}


/*
```

```
 * Reverse the order of nodes of a singly linked list
 */
void reverseList()
{
   struct node *prevNode, *curNode;

   if(head != NULL)
   {
      prevNode = head;
      curNode = head->next;
      head = head->next;

      prevNode->next = NULL; // Make first node as last node

      while(head != NULL)
      {
         head = head->next;
                              curNode->next = prevNode;

         prevNode = curNode;
         curNode = head;
      }

            head = prevNode; // Make last node as head

      printf("SUCCESSFULLY REVERSED LIST\n");
   }
}


/*
 * Display entire list
 */
void displayList()
{
   struct node *temp;

   /*
    * If the list is empty i.e. head = NULL
    */
   if(head == NULL)
   {
      printf("List is empty.");
   }
   else
   {
      temp = head;
            while(temp != NULL)
      {
         printf("Data = %d\n", temp->data); // Print the data of current node
         temp = temp->next;                 // Move to next node
```

```
        }
    }
}
```

Output :

```
Enter the total number of nodes: 5
Enter the data of node 1: 10
Enter the data of node 2: 20
Enter the data of node 3: 30
Enter the data of node 4: 40
Enter the data of node 5: 50
SINGLY LINKED LIST CREATED SUCCESSFULLY

Data in the list
Data = 10
Data = 20
Data = 30
Data = 40
Data = 50
Press 1 to reverse the order of singly linked list
1
SUCCESSFULLY REVERSED LIST

Data in the list
Data = 50
Data = 40
Data = 30
Data = 20
Data = 10
```