DATA STRUCTURES THROUGH C LABORATORY

N.MARIYA BABU

(N190750)

CSE-03

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, RGUKT-NUZ-AP

E-1 SEMESTER-II, AY-2021-2022

Table of Contents

LAB-1

```
/*              Author Name : N.Mariya Babu
                        Id  No : N190750
      Assignment Number : 1
      programme Number : 1
 programme Description :  C Programme to find the gcd of two numbers
                        Date :  6/21/2022 (MM/DD/YYYY)
                certification :  I hereby certify that this work is my own and none of it is the work
of any other person     */
//Header Files
#include<stdio.h>
//Function Definition
int gcd(int,int);
//Main Function
int main(){
   int a,b,result;
   printf("Enter the a value :");
   scanf("%d",&a);
   printf("Enter the b value :");
   scanf("%d",&b);
   if(a>b){
       result = gcd(a,b);
   }else{
      result = gcd(b,a);
   }
   printf("The gcd of the %d and %d is %d",a,b,result);
}
//Function to find GCD
int gcd(int a ,int b){
   int rem;
   rem = a%b;
   if(rem==0){
      return b;
   }
   else{
      return gcd(b,rem);
   }
}
```

Output:

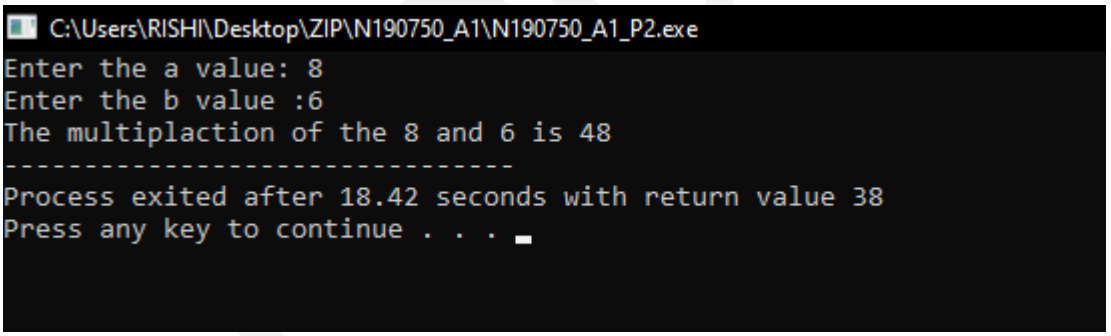
```
C:\Users\RISHI\Desktop\ZIP\N190750_A1\N190750_A1_P1.exe
Enter the a value :6
Enter the b value :9
The gcd of the 6 and 9 is 3
--------------------------------
Process exited after 12.12 seconds with return value 27
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                        Id  No : N190750
      Assignment Number : 1
      programme Number : 2
 programme Description : C Programme for multiplication  without using the
division operator
                        Date : 6/21/2022 (MM/DD/YYYY)
```

certification : I hereby certify that this work is my own and none of it is the work of any other person */

```
//Header Files
#include<stdio.h>
#include<math.h>
int mult(int,int);
//Main Function
int main(){
   int a,b,i,result;
   printf("Enter the a value: ");
   scanf("%d",&a);
   printf("Enter the b value :");
   scanf("%d",&b);
   result = mult(a,b);
   printf("The multiplication of the %d and %d is %d",a,b,result);
}
//Function to multiply without out * operator
int mult(int a,int b){
   if(b>0){
     a = a + mult(a,b-1);
     return a;
   }
   else{
      return 0;
   }
}
```

Output:

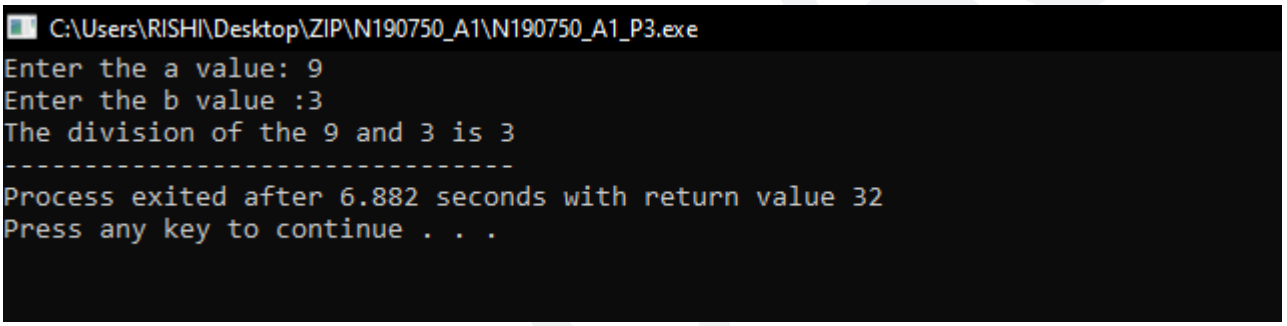


/* **Author Name : N.Mariya Babu**
Id No. : N190750
Assignment Number : 1
programme Number : 3
programme Description : C Programme for division without using the division operator
Date : 6/28/2022 (MM/DD/YYYY)
certification : I hereby certify that this work is my own and none of it is the work of any other person */

```
//Header Files
#include<stdio.h>
#include<math.h>
//Function Declaration
int div(int,int);
//Main Function
int main(){
   int a,b,i,result;
   printf("Enter the a value: ");
```

```
   scanf("%d",&a);
   printf("Enter the b value :");
   scanf("%d",&b);
   result = div(a,b);
   printf("The division of the %d and %d is %d",a,b,result);
}
//Function to find the division of two numbers
int div(int a,int b){
   int ;
   if(a>=b){
      return 1 + div(a-b,b);
   }
   else{
      return 0;
   }
}
```

Output:




```
/*                    Author Name : N.Mariya Babu
                            Id  No. : N190750
      Assignment Number : 1
      programme Number : 4
 programme Description : C Programme to find the power of two numbers without using the
power function
                              Date : 6/21/2022 (MM/DD/YYYY)
                      certification : I hereby certify that this work is my own and none of it is the
work of any other person      */
//Header Files
#include<stdio.h>
//Function Declaration
int power(int,int);
//Main Function
int main(){
   int a,b,i,result;
   printf("Enter the a value: ");
   scanf("%d",&a);
   printf("Enter the b value :");
   scanf("%d",&b);
   result = power(a,b);
   printf("The %d power  %d is %d",a,b,result);
}
//Power Function
int power(int a ,int b){
   if(b>0){
      a = a * power(a,b-1);
      return a;
```

```
    }
    else{
        return 1;
    }
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A1\N190750_A1_P4.exe
Enter the a value: 8
Enter the b value :3
The 8 power  3 is 512
--------------------------------
Process exited after 16.92 seconds with return value 21
Press any key to continue . . .
```

```
/*                    Author Name : N.Mariya Babu
                                Id  No  : N190750
      Assignment Number : 1
     programme Number : 5
programme Description : C programme to find the given number is palindrome or not
                                Date : 6/21/2022 (MM/DD/YYYY)
                    certification : I hereby certify that this work is my own and none of it is the work
of any other person    */
//Header Files
#include<stdio.h>
//Function Declaration
int rev(int);
//Main Function
int main(){
    int n,revers;
    printf("Enter the n value ");
            printf("to check the number is palindrome or not :");
    scanf("%d",&n);
    revers = rev(n);
    if(revers==n){
        printf("The given number is palindrome!");
    }
    else{
        printf("the given number is not palindrome");
    }
}
//Function to reverse the given number
int rev(int n){
    static int rem = 0 ;
    if(n>0){
        rem = rem * 10 + n%10;
        return rev(n/10);
    }
    else{
        return rem;
    }
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A1\N190750_A1_P5.exe
Enter the n value to check the number is polindrome or not :89698
The given number is polindrome!
--------------------------------
Process exited after 16.52 seconds with return value 31
Press any key to continue . . .
```

LAB-2

/* **Author Name : N.Mariya Babu**
Id No. : N190750
Assignment Number : 2
programme Number : 1
programme Description : C Programme to create a SLL and perform all insertion and deletion cases
Date : 6/28/2022 (MM/DD/YYYY)
certification : I hereby certify that this work is my own and none of it is the work of any other person */

```
//Header Files
#include<stdio.h>
#include<stdlib.h>
// function declaration
int read(void);
int display(void);
int insertion(void);
int deletion(void);
int searching(void);
//structure creation
struct node{
	int data;
	struct node *next;
}*head= NULL,*p,*qtr,*r,*nn,*ptr;
// main function
int main(){
	while(1){
	int n;
	printf("\n1 create list 2 insertion ");
	printf(" 3 deletion 4 searching 5 exit \n");
	printf("\nChoose your option :");
	scanf("%d",&n);
	switch(n){
		case 1:
			read();
			break;
		case 2:
			insertion();
			break;
		case 3:
			deletion();
			break;
		case 4:
			searching();
			break;
		case 5:
```

```
                        exit(1);
                        break;
                default:
                        printf("Invalid Syntax");
        }
        display();
}
}
// funcion to read the node element's
int read(){
        int i =1;
        while(i != 0){
                nn = (struct node *)malloc(sizeof(struct node));
                printf("Enter the node data :");
                scanf("%d",&nn->data);
                if(head == NULL){
                        head = nn;
                        nn->next = NULL;
                }
                else{
                        ptr = head;
                        while(ptr->next !=NULL){
                                ptr = ptr->next;
                        }
                        ptr->next = nn;
                        nn->next = NULL;
                }
                printf("Enter the 0 to exit :");
                scanf("%d",&i);
        }
        return 0;
}
// function to display the node element's
int display(){
        ptr = head;
        int i = 0;
        while(ptr!=NULL){
                printf("\nThe data at node %d is %d ",i,ptr->data);
                ptr = ptr->next;
                i++;
        }
        return 0 ;
}
//function for inserting the element at required position
int insertion(){
        int x=0,y = 0,k = 1;
        ptr = head;
                nn = (struct node *)malloc(sizeof(struct node));
                printf("Enter the node data :");
                scanf("%d",&nn->data);
                nn->next = NULL;
        while(ptr!=NULL){
                ptr= ptr->next;
                y++;
        }
        int n;
        while(x==0){
```

```
		printf("\nEnter the position where you want");
		printf(" to insert the data :");
		scanf("%d",&n);
		if(n<=y+1 && n>0){
			x++;
		}
		else{
			printf("Invalid position");
		}
	}
	if(n==1){
		nn->next = head;
		head = nn;

	}else if(n==y+1){
		ptr = head;
		while(ptr->next != NULL){
			ptr = ptr->next;
		}
		ptr->next = nn;

	}else{
		ptr = head;
		for(k=1;k<n;k++){
			ptr = ptr->next;
		}
		nn->next = ptr->next;
		ptr->next = nn;
	}
	return 0;
}
// function to delete the element from the list
int deletion(){
	int x=0,y = 0,k = 1;
	ptr = head;
	while(ptr!=NULL){
		ptr= ptr->next;
		y++;
	}
	int n;
	while(x==0){
		printf("\nEnter the position of the element  ");
		printf("Where you want the delete : ");
		scanf("%d",&n);
		if(n<=y && n>0){
			x++;
		}
		else{
			printf("Invalid position");
		}
	}
	if(n==1){
		ptr = head;
		head = ptr->next;
		free(ptr);
	}else if(n==y){
		ptr = head;
```

```
                while(ptr->next != NULL){
                        qtr = ptr;
                        ptr = ptr->next;
                }
                qtr->next = NULL;
                free(ptr);
        }else{
                ptr = head;
                for(k=1;k<n;k++){
                        qtr = ptr;
                        ptr = ptr->next;
                }
                qtr->next = ptr->next;
                free(ptr);
        }
        return 0;
}
//Searching
int searching(){
        int n,i;
        printf("Enter the position of the element :");
        scanf("%d",&n);
        ptr = head;
        for(i=1;i<n;i++){
                ptr=ptr->next;
        }
        printf("The element at position %d is %d",n,ptr->data);
        ptr = head;
        return 0;
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A2\N190750_A2_P1.c.exe

1 create list 2 insertation  3 deletion 4 searching 5 exit

Choose your option :1
Enter the node data :4
Enter the 0 to exit :1
Enter the node data :5
Enter the 0 to exit :0

The data at node 0 is 4
The data at node 1 is 5
1 create list 2 insertation  3 deletion 4 searching 5 exit

Choose your option :2
Enter the node data :9

Enter the position where you want to insert the data :1

The data at node 0 is 9
The data at node 1 is 4
The data at node 2 is 5
1 create list 2 insertation  3 deletion 4 searching 5 exit

Choose your option :3

Enter the position of the element  Where you want the delete : 2

The data at node 0 is 9
The data at node 1 is 5
1 create list 2 insertation  3 deletion 4 searching 5 exit

Choose your option :4
Enter the position of the element :2
The element at position 2 is 5
The data at node 0 is 9
The data at node 1 is 5
1 create list 2 insertation  3 deletion 4 searching 5 exit

Choose your option :5

--------------------------------
Process exited after 68.1 seconds with return value 1
Press any key to continue . . .
```

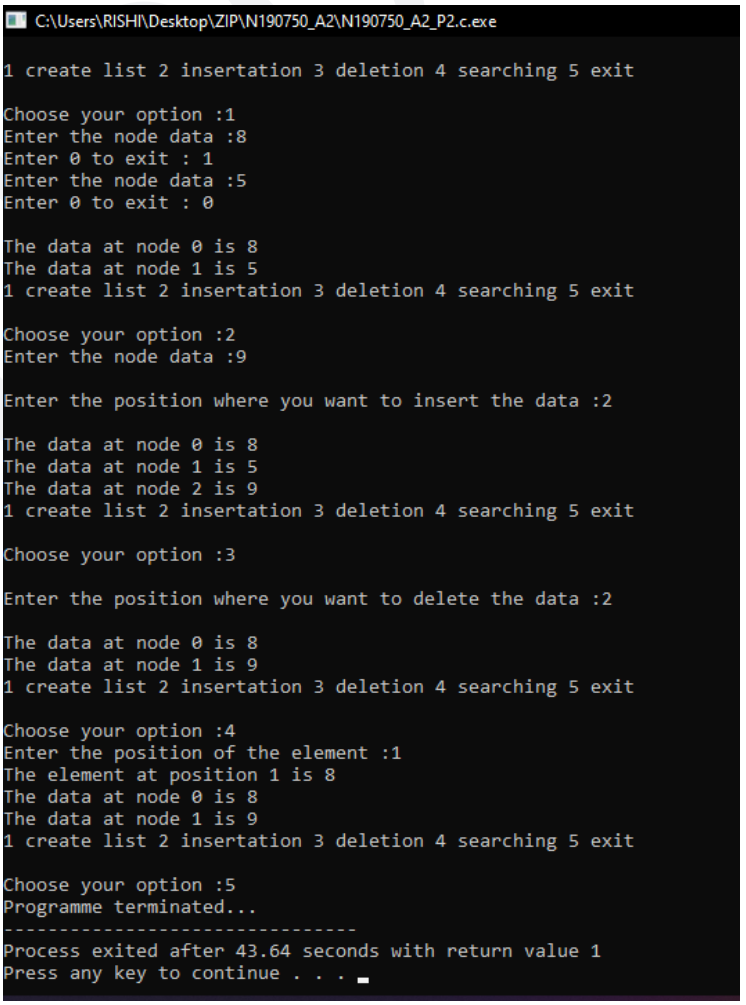
```
/*                  Author Name : N.Mariya Babu
                          Id  No. : N190750
      Assignment Number : 2
     programme Number : 2
  programme Description : C Programme to create a CLL and perform insertion and deletion at
beginning and end
                              Date : 6/28/2022 (MM/DD/YYYY)
                      certification : I hereby certify that this work is my own and none of it is the
work of any other person      */
//Header Files
#include<stdio.h>
#include<stdlib.h>
// function declaration
int read(void);
int display(void);
int insertion(void);
int deletion(void);
int searching(void);
//structure creation
struct node{
        int data;
        struct node *next;
}*head= NULL,*p,*qtr,*r,*nn,*ptr;
// main function
int main(){
        while(1){
        int n;
        printf("\n1 create list 2 insertion ");
        printf("3 deletion 4 searching 5 exit \n");
        printf("\nChoose your option :");
        scanf("%d",&n);
        switch(n){
                case 1:
                        read();
                        break;
                case 2:
                        insertion();
                        break;
                case 3:
                        deletion();
                        break;
                case 4:
                        searching();
                        break;
                case 5:
                        printf("Programme terminated...");
                        exit(1);
                        break;
                default:
                        printf("Invalid Syntax");
        }
        display();
}
}
// funcion to read the node element's
int read(){
        int i =1;
```

```
	while(i != 0){
		nn = (struct node *)malloc(sizeof(struct node));
		printf("Enter the node data :");
		scanf("%d",&nn->data);
		if(head == NULL){
			head = nn;
			nn->next = head;
		}
		else{
			ptr = head;
			while(ptr->next !=head){
				ptr = ptr->next;
			}
			ptr->next = nn;
			nn->next = head;
		}
		printf("Enter 0 to exit : ");
		scanf("%d",&i);
	}
	return 0;
}
// function to display the node element's
int display(){
	ptr = head;
	int i = 0;
	while(ptr->next!=head){
		printf("\nThe data at node %d is %d ",i,ptr->data);
		ptr = ptr->next;
		i++;
	}
	printf("\nThe data at node %d is %d ",i,ptr->data);
	return 0 ;
}
//function for inserting the element at required position
int insertion(){
	int x=0,y = 1,k = 1;
	ptr = head;
		nn = (struct node *)malloc(sizeof(struct node));
		printf("Enter the node data :");
		scanf("%d",&nn->data);
		nn->next = NULL;
	while(ptr->next!=head){
		ptr= ptr->next;
		y++;
	}
	int n;
	while(x==0){
		printf("\nEnter the position where ");
		printf("you want to insert the data :");
		scanf("%d",&n);
		if(n<=y+1 && n>0){
			x++;
		}
		else{
			printf("Invalid position");
		}
	}
```

```
	if(n==1){
		ptr = head->next;
		while(ptr->next!=head){
			ptr = ptr->next;
		}
		nn->next = head;
		head = nn;
		ptr->next =  head;

	}else if(n==y+1){
		ptr = head->next;
		while(ptr->next != head){
			ptr = ptr->next;
		}
		ptr->next = nn;
		nn->next = head;
	}else{
		ptr = head;
		for(k=1;k<n;k++){
			ptr = ptr->next;
		}
		nn->next = ptr->next;
		ptr->next = nn;
	}
	return 0;
}
// function to delete the element from the list
int deletion(){
	int x=0,y = 1,k = 1;
	ptr = head->next;
	while(ptr!=head){
		ptr= ptr->next;
		y++;
	}
	int n;
	while(x==0){
		printf("\nEnter the position where ");
		printf("you want to delete the data :");
		scanf("%d",&n);
		if(n<=y && n>0){
			x++;
		}
		else{
			printf("Invalid position");
		}
	}
	if(n==1){
		ptr = head->next;
		while(ptr->next!=head){
			ptr = ptr->next;
		}
		ptr->next = head->next;
		ptr= head;
		head = head->next;
		free(ptr);
	}else if(n==y){
		ptr = head->next;
```

```
                while(ptr->next != head){
                        qtr = ptr;
                        ptr = ptr->next;
                }
                qtr->next = head;
                free(ptr);
        }else{
                ptr = head;
                for(k=1;k<n;k++){
                        qtr = ptr;
                        ptr = ptr->next;
                }
                qtr->next = ptr->next;
                free(ptr);
        }
        return 0;
}
//function to get the element at required position
int searching(){
        int n,i;
        printf("Enter the position of the element :");
        scanf("%d",&n);
        ptr = head;
        for(i=1;i<n;i++){
                ptr=ptr->next;
        }
        printf("The element at position %d is %d",n,ptr->data);
        ptr = head;
        return 0;
}
```

Output:



```
C:\Users\RISHI\Desktop\ZIP\N190750_A2\N190750_A2_P2.c.exe

1 create list 2 insertation 3 deletion 4 searching 5 exit

Choose your option :1
Enter the node data :8
Enter 0 to exit : 1
Enter the node data :5
Enter 0 to exit : 0

The data at node 0 is 8
The data at node 1 is 5
1 create list 2 insertation 3 deletion 4 searching 5 exit

Choose your option :2
Enter the node data :9

Enter the position where you want to insert the data :2

The data at node 0 is 8
The data at node 1 is 5
The data at node 2 is 9
1 create list 2 insertation 3 deletion 4 searching 5 exit

Choose your option :3

Enter the position where you want to delete the data :2

The data at node 0 is 8
The data at node 1 is 9
1 create list 2 insertation 3 deletion 4 searching 5 exit

Choose your option :4
Enter the position of the element :1
The element at position 1 is 8
The data at node 0 is 8
The data at node 1 is 9
1 create list 2 insertation 3 deletion 4 searching 5 exit

Choose your option :5
Programme terminated...
--------------------------------
Process exited after 43.64 seconds with return value 1
Press any key to continue . . .
```

```
/*      Author Name : N.Mariya Babu
                Id  No  : N190750
    Assignment Number : 2
    programme Number : 3
 programme Description : C Programme to creat a DLL  and perform all insertion and deletion cases
                Date : 6/28/2022 (MM/DD/YYYY)
            certification : I hereby certify that this work is my own and none of it is the work of any other person     */
//Header Files
#include<stdio.h>
#include<stdlib.h>
// function declaration
int read(void);
int display(void);
int insertion(void);
int deletion(void);
int searching(void);
//structure creation
struct node{
        struct node *previous;
        int data;
        struct node *next;
}*head= NULL,*p,*qtr,*r,*nn,*ptr;
// main function
int main(){
        while(1){
        int n;
        printf("\n1 create list  2 insertion ");
        printf(" 3 deletion  4 searching  5 exit \n");
        printf("\nChoose your option :");
        scanf("%d",&n);
        switch(n){
                case 1:
                        read();
                        break;
                case 2:
                        insertion();
                        break;
                case 3:
                        deletion();
                        break;
                case 4:
                        searching();
                        break;
                case 5:
                        exit(1);
                        break;
                default:
                        printf("Invalid Syntax");
        }
        display();
}
}
// funcion to read the node element's
int read(){
        int i =1;
```

```
        while(i != 0){
                nn = (struct node *)malloc(sizeof(struct node));
                printf("Enter the node data :");
                scanf("%d",&nn->data);
                if(head == NULL){
                        head = nn;
                        nn->previous = NULL;
                        nn->next = NULL;
                }
                else{
                        ptr = head;
                        while(ptr->next !=NULL){
                                ptr = ptr->next;
                        }
                        nn->previous = ptr;
                        ptr->next = nn;
                        nn->next = NULL;
                }
                printf("Enter 0 to exit :");
                scanf("%d",&i);
        }
        return 0;
}
// function to display the node element's
int display(){
        ptr = head;
        int i = 0;
        while(ptr!=NULL){
                printf("\nThe data at node %d is %d ",i,ptr->data);
                ptr = ptr->next;
                i++;
        }
        return 0 ;
}
//function for inserting the element at required position
int insertion(){
        int x=0,y = 0,k = 1;
        ptr = head;
                nn = (struct node *)malloc(sizeof(struct node));
                printf("Enter the node data :");
                scanf("%d",&nn->data);
                nn->next = NULL;
                nn->previous = NULL;
        while(ptr!=NULL){
                ptr= ptr->next;
                y++;
        }
        int n;
        while(x==0){
                printf("\nEnter the position where ");
                printf("you want to insert the data :");
                scanf("%d",&n);
                if(n<=y+1 && n>0){
                        x++;
                }
                else{
                        x =0;
```

```
                printf("Invalid position");
            }
        }
        if(n==1){
            nn->next = head;
            head->previous = nn;
            head = nn;

        }else if(n==y+1){
            ptr = head;
            while(ptr->next != NULL){
                ptr = ptr->next;
            }
            ptr->next = nn;
            nn->previous = ptr;
        }else{
            ptr = head;
            for(k=1;k<n;k++){
                qtr = ptr;
                ptr = ptr->next;
            }
            nn->next = ptr;
            nn->previous = qtr;
            ptr->previous = nn;
            qtr->next = nn;
        }
        return 0;
}
// function to delete the element from the list
int deletion(){
        int x=0,y = 0,k = 1;
        ptr = head;
        while(ptr!=NULL){
            ptr= ptr->next;
            y++;
        }
        int n;
        while(x==0){
            printf("\nEnter the position where ");
            printf(" you want to delete the data :");
            scanf("%d",&n);
            if(n<=y && n>0){
                x++;
            }
            else{
                printf("Invalid position");
            }
        }
        if(n==1){
            ptr = head;
            head = ptr->next;
            head->previous = NULL;
            free(ptr);
        }else if(n==y){
            ptr = head;
            while(ptr->next != NULL){
                qtr = ptr;
```

```
                ptr = ptr->next;
            }
            qtr->next = NULL;
            free(ptr);
        }else{
            ptr = head;
            for(k=1;k<n;k++){
                ptr = ptr->next;
            }
            ptr->previous->next = ptr->next;
            ptr->next->previous = ptr->previous;
            free(ptr);
        }
        return 0;
}
//function to get the element at required position
int searching(){
        int n,i;
        printf("Enter the position of the element :");
        scanf("%d",&n);
        ptr = head;
        for(i=1;i<n;i++){
            ptr=ptr->next;
        }
        printf("The element at position %d is %d",n,ptr->data);
        ptr = head;
        return 0;
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A2\N190750_A2_P3.c.exe

1 create list  2 insertation  3 deletion  4 searching  5 exit

Choose your option :1
Enter the node data :9
Enter 0 to exit :1
Enter the node data :4
Enter 0 to exit :0

The data at node 0 is 9
The data at node 1 is 4
1 create list  2 insertation  3 deletion  4 searching  5 exit

Choose your option :2
Enter the node data :1

Enter the position where you want to insert the data :5
Invalid position
Enter the position where you want to insert the data :2

The data at node 0 is 9
The data at node 1 is 1
The data at node 2 is 4
1 create list  2 insertation  3 deletion  4 searching  5 exit

Choose your option :3

Enter the position where  you want to delete the data :1

The data at node 0 is 1
The data at node 1 is 4
1 create list  2 insertation  3 deletion  4 searching  5 exit

Choose your option :
```

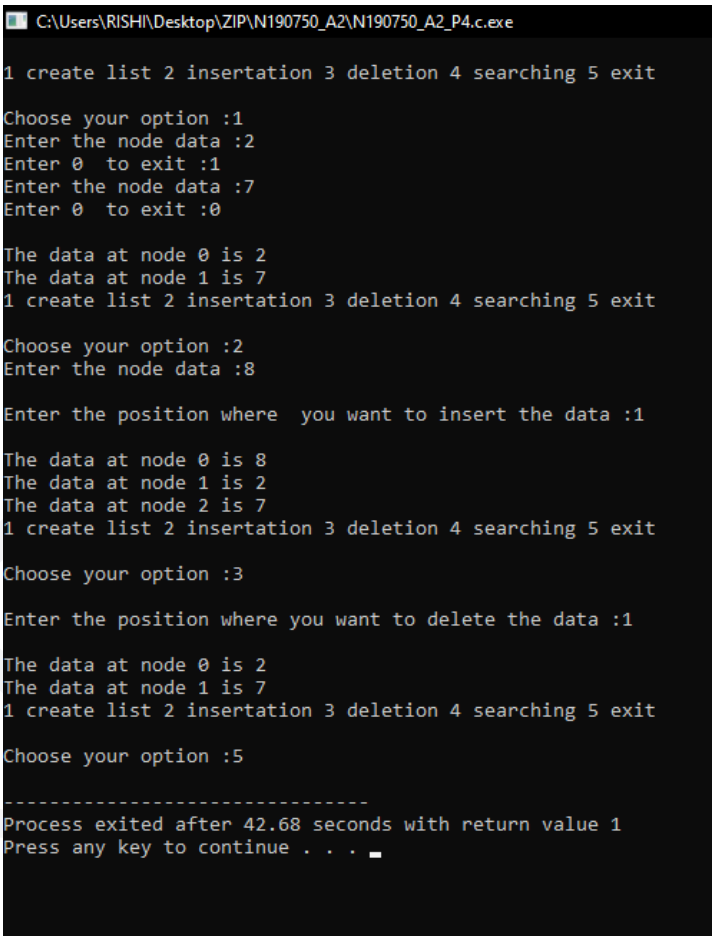
```
/*                    Author Name : N.Mariya Babu
                              Id  No. : N190750
        Assignment Number : 2
        programme Number : 4
  programme Description : C Programme to create a DLL and perform insertion and deletion at
beginning and end
                                  Date : 6/28/2022 (MM/DD/YYYY)
                        certification : I hereby certify that this work is my own and none of it is the
work of any other person    */
//Header Files
#include<stdio.h>
#include<stdlib.h>
// function declaration
int read(void);
int display(void);
int insertion(void);
int deletion(void);
int searching(void);
//structure creation
struct node{
        struct node *previous;
        int data;
        struct node *next;
}*head= NULL,*p,*qtr,*r,*nn,*ptr;
// main function
int main(){
        while(1){
        int n;
        printf("\n1 create list 2 insertion ");
        printf("3 deletion 4 searching 5 exit\n");
        printf("\nChoose your option :");
        scanf("%d",&n);
        switch(n){
                case 1:
                        read();
                        break;
                case 2:
                        insertion();
                        break;
                case 3:
                        deletion();
                        break;
                case 4:
                        searching();
                        break;
                case 5:
                        exit(1);
                        break;
                default:
                        printf("Invalid Syntax");
        }
        display();
}
}
// funcion to read the node element's
int read(){
        int i =1;
```

```
        while(i != 0){
                nn = (struct node *)malloc(sizeof(struct node));
                printf("Enter the node data :");
                scanf("%d",&nn->data);
                if(head == NULL){
                        head = nn;
                        nn->previous = NULL;
                        nn->next = head;
                }
                else{
                        ptr = head;
                        while(ptr->next !=head){
                                ptr = ptr->next;
                        }
                        nn->previous = ptr;
                        ptr->next = nn;
                        nn->next = head;
                        head->previous = nn;
                }
                printf("Enter 0  to exit :");
                scanf("%d",&i);
        }
        return 0;
}
// function to display the node element's
int display(){
        ptr = head;
        int i = 0;
        while(ptr->next!=head){
                printf("\nThe data at node %d is %d ",i,ptr->data);
                ptr = ptr->next;
                i++;
        }
        printf("\nThe data at node %d is %d ",i,ptr->data);
        return 0 ;
}
//function for inserting the element at required position
int insertion(){
        int x=0,y = 0,k = 1;
        ptr = head;
                nn = (struct node *)malloc(sizeof(struct node));
                printf("Enter the node data :");
                scanf("%d",&nn->data);
        while(ptr->next!=head){
                ptr= ptr->next;
                y++;
        }
        y++;
        int n;
        while(x==0){
                printf("\nEnter the position where ");
                printf(" you want to insert the data :");
                scanf("%d",&n);
                if(n<=y+1 && n>0){
                        x++;
                }
                else{
```

```
                x =0;
                printf("Invalid position");
            }
        }
        if(n==1){
            nn->next = head;
            nn->previous = head->previous;
            head->previous = nn;
            ptr = head;
            while(ptr->next != head){
                ptr = ptr->next;
            }
            head = nn;
            ptr->next=head;

        }else if(n==y+1){
            ptr = head;
            while(ptr->next != head){
                ptr = ptr->next;
            }
            nn->next = head;
            nn->previous = ptr;
            ptr->next = nn;
            head->previous = nn;
        }else{
            ptr = head;
            for(k=1;k<n;k++){
                qtr = ptr;
                ptr = ptr->next;
            }
            nn->next = ptr;
            nn->previous = qtr;
            ptr->previous = nn;
            qtr->next = nn;
        }
        return 0;
}
// function to delete the element from the list
int deletion(){
        int x=0,y = 0,k = 1;
        ptr = head;
        while(ptr->next!=head){
            ptr= ptr->next;
            y++;
        }
        y++;
        int n;
        while(x==0){
            printf("\nEnter the position where ");
            printf("you want to delete the data :");
            scanf("%d",&n);
            if(n<=y && n>0){
                x++;
            }
            else{
                printf("Invalid position");
            }
```

```
        }
        if(n==1){
                ptr = head;
                while(ptr->next!=head){
                        ptr=ptr->next;
                }
                ptr->next = head->next;
                head->next->previous = ptr;
                ptr = head;
                head = head->next;
                free(ptr);
        }else if(n==y){
                ptr = head;
                while(ptr->next != head){
                        qtr = ptr;
                        ptr = ptr->next;
                }
                qtr->next = head;
                head->previous = qtr;
                free(ptr);
        }else{
                ptr = head;
                for(k=1;k<n;k++){
                        ptr = ptr->next;
                }
                ptr->previous->next = ptr->next;
                ptr->next->previous = ptr->previous;
                free(ptr);
        }
        return 0;
}
//function to get the element at required position
int searching(){
        int n,i;
        printf("Enter the position of the element :");
        scanf("%d",&n);
        ptr = head;
        for(i=1;i<n;i++){
                ptr=ptr->next;
        }
        printf("The element at position %d is %d",n,ptr->data);
        ptr = head;
        return 0;
}
```

Output:



```
C:\Users\RISHI\Desktop\ZIP\N190750_A2\N190750_A2_P4.c.exe

1 create list 2 insertation 3 deletion 4 searching 5 exit

Choose your option :1
Enter the node data :2
Enter 0  to exit :1
Enter the node data :7
Enter 0  to exit :0

The data at node 0 is 2
The data at node 1 is 7
1 create list 2 insertation 3 deletion 4 searching 5 exit

Choose your option :2
Enter the node data :8

Enter the position where  you want to insert the data :1

The data at node 0 is 8
The data at node 1 is 2
The data at node 2 is 7
1 create list 2 insertation 3 deletion 4 searching 5 exit

Choose your option :3

Enter the position where you want to delete the data :1

The data at node 0 is 2
The data at node 1 is 7
1 create list 2 insertation 3 deletion 4 searching 5 exit

Choose your option :5

--------------------------------
Process exited after 42.68 seconds with return value 1
Press any key to continue . . .
```

```
/*                      Author Name : N.Mariya Babu
                                Id  No. : N190750
        Assignment Number : 2
        programme Number : 5
  programme Description : C Programme to store polynomial in a linked list and apply add and
subtract operations on 2 polynomials
                                Date : 6/28/2022 (MM/DD/YYYY)
                        certification : I hereby certify that this work is my own and none of it is the
work of any other person      */
//Header Files
#include<stdio.h>
```

```
#include<stdlib.h>
//Function declaration
int read(void);
int display(void);
int add(void);
int sub(void);
//Structure definition in the global declaration
struct poly{
	int coff1;
	int coff2;
	struct poly *next;
}*head=NULL,*ptr,*qtr,*nn;
//Main function
int main(){
	int n,i = 1;
	while(i!=0){
		printf("\n1 read polynomial 2 add two polynomials ");
		printf("3 subtract two polynomials 4 exit");
		printf("\nChoose your option :");
		scanf("%d",&n);
		switch(n){
			case 1:
				read();
				display();
				break;
			case 2:
				add();
				break;
			case 3:
				sub();
				break;
			case 4:
				exit(1);
			default:
				printf("Invalid syntax!...");
		}
	}
}
//Function to read the coefficient of the polynomials
int read(){
	int i=1,j=0;
	while(i!=0){
		nn = (struct poly *)malloc(sizeof(struct poly));
		printf("Enter  the fun1 x power %d coefficient :",j);
		scanf("%d",&nn->coffs1);
		printf("Enter  the fun2 x power %d coefficient :",j);
		scanf("%d",&nn->coff2);
		j++;
		if(head==NULL){
			head = nn;
			nn->next = NULL;
		}else{
			ptr = head;
			while(ptr->next!=NULL){
				ptr = ptr->next;
			}
			ptr->next = nn;
```

```
                        nn->next = NULL;
                }
                printf("0 to exit press any number for new coffercient");
                scanf("%d",&i);
        }
}
//Function to display the polynomial
int display(){
        ptr= head;
        int i = 0;
        printf("\nFirst polynomial :\n");
        while(ptr!=NULL){
                printf("+(%d)x^ %d ",ptr->coff1,i);
                ptr = ptr->next;
                i++;
        }
         i= 0 ;
        ptr = head;
        printf("\nSecond polynomial :\n");
        while(ptr!=NULL){
                printf("+(%d)x^ %d ",ptr->coff2,i);
                ptr = ptr->next;
                i++;
        }
}
//Function to add the polynomials
int add(){
        ptr = head;
        int i = 0 ;
        printf("\n The addition of the two polynomials is :\n'");
        while(ptr!=NULL){
                printf("+(%d)x^ %d ",ptr->coff1+ptr->coff2,i);
                ptr = ptr->next;
                i++;
        }
        return 0;
}
//Function to subtract the polynomials
int sub(){
        ptr = head;
        int i = 0 ;
        printf("\n The addition of the two polynomials is :\n'");
        while(ptr!=NULL){
                printf("+(%d)x^ %d ",ptr->coff1-ptr->coff2,i);
                ptr = ptr->next;
                i++;
        }
        return 0;
}
```

Output:

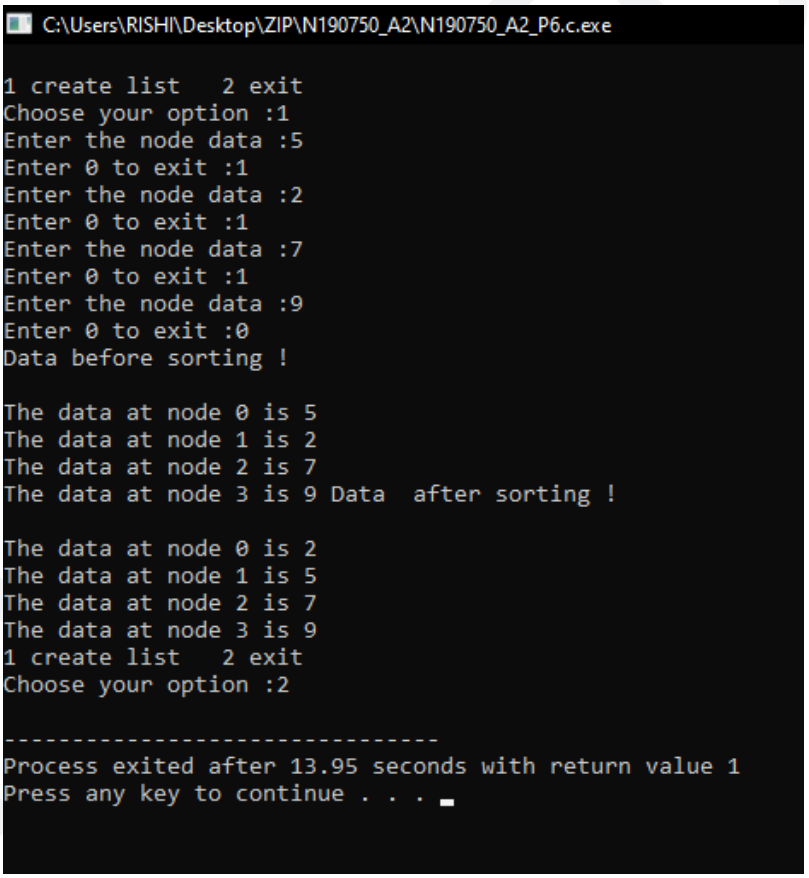
```
C:\Users\RISHI\Desktop\ZIP\N190750_A2\N190750_A2_P5.exe

1 read polinomiyal 2 add two polinomiyals 3 subtract two polinomiyals 4 exit
Choose your option :1
Enter  the fun1 x power 0 cofficeint :5
Enter  the fun2 x power 0 cofficeint :3
0 to exit press any number for new coffercient1
Enter  the fun1 x power 1 cofficeint :2
Enter  the fun2 x power 1 cofficeint :4
0 to exit press any number for new coffercient1
Enter  the fun1 x power 2 cofficeint :8
Enter  the fun2 x power 2 cofficeint :5
0 to exit press any number for new coffercient0

First polinomiyal :
+(5)x^ 0 +(2)x^ 1 +(8)x^ 2
Second polinomial :
+(3)x^ 0 +(4)x^ 1 +(5)x^ 2
1 read polinomiyal 2 add two polinomiyals 3 subtract two polinomiyals 4 exit
Choose your option :2

 The addtion of the two polinomiyal's is :
'+(8)x^ 0 +(6)x^ 1 +(13)x^ 2
1 read polinomiyal 2 add two polinomiyals 3 subtract two polinomiyals 4 exit
Choose your option :3

 The addtion of the two polinomiyal's is :
'+(2)x^ 0 +(-2)x^ 1 +(3)x^ 2
1 read polinomiyal 2 add two polinomiyals 3 subtract two polinomiyals 4 exit
Choose your option :4

--------------------------------
Process exited after 54.83 seconds with return value 1
Press any key to continue . . .
```

```
/*
                    Author Name : N.Mariya Babu
                Id  No. : N190750
        Assignment Number : 2
    programme Number : 6
 programme Description : C Programme to sort the numbers given in a LL
                    Date : 6/28/2022 (MM/DD/YYYY)
            certification : I hereby certify that this work is my own and none of it is the work of any other
person
*/
//Header Files
#include<stdio.h>
#include<stdlib.h>
// function declaration
int read(void);
int display(void);
int sorting(void);
//structure creation
struct node{
        int data;
        struct node *next;
}*head= NULL,*p,*qtr,*r,*nn,*ptr;
// main function
int main(){
        while(1){
```

```
		int n;
		printf("\n1 create list   2 exit");
		printf("\nChoose your option :");
		scanf("%d",&n);
		if(n==1){
			read();
			printf("Data before sorting !\n");
			display();
			printf("Data  after sorting !\n");
			sorting();
			display();
		}
		else if(n==2){
			exit(1);
		}
		else{
			printf("invalid..");
		}
	}
}
// funcion to read the node element's
int read(){
	int i =1;
	while(i != 0){
		nn = (struct node *)malloc(sizeof(struct node));
		printf("Enter the node data :");
		scanf("%d",&nn->data);
		if(head == NULL){
			head = nn;
			nn->next = NULL;
		}
		else{
			ptr = head;
			while(ptr->next !=NULL){
				ptr = ptr->next;
			}
			ptr->next = nn;
			nn->next = NULL;
		}
		printf("Enter 0 to exit :");
		scanf("%d",&i);
	}
	return 0;
}
// function to display the node element's
int display(){
	ptr = head;
	int i = 0;
	while(ptr!=NULL){
		printf("\nThe data at node %d is %d ",i,ptr->data);
		ptr = ptr->next;
		i++;
	}
	return 0 ;
}
//Function to sort the element's
int sorting(){
```

```
        int temp = 0;
        ptr =head;
        while(ptr!=NULL){
                qtr =  ptr->next;
                while(qtr!=NULL){
                        if(ptr->data > qtr->data){
                                temp = ptr->data;
                                ptr->data = qtr->data;
                                qtr->data = temp;
                        }
                        qtr = qtr->next;
                }
                ptr= ptr->next;
        }
        return 0;
}
```

Output:



```
C:\Users\RISHI\Desktop\ZIP\N190750_A2\N190750_A2_P6.c.exe

1 create list   2 exit
Choose your option :1
Enter the node data :5
Enter 0 to exit :1
Enter the node data :2
Enter 0 to exit :1
Enter the node data :7
Enter 0 to exit :1
Enter the node data :9
Enter 0 to exit :0
Data before sorting !

The data at node 0 is 5
The data at node 1 is 2
The data at node 2 is 7
The data at node 3 is 9 Data  after sorting !

The data at node 0 is 2
The data at node 1 is 5
The data at node 2 is 7
The data at node 3 is 9
1 create list   2 exit
Choose your option :2

--------------------------------
Process exited after 13.95 seconds with return value 1
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                        Id  No. : N190750
     Assignment Number : 2
    programme Number : 7
 programme Description : C Programme to create LL which stores details of students
                                        and print the names of the students who got first class
                        Date : 6/28/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the
work of any other person
*/
//Header Files
#include<stdio.h>
#include<stdlib.h>
//Function declaration
```

```
int read(void);
int display(void);
int rank(void);
//structure declaration
struct student{
	char name[50];
	int marks;
	struct student *next;
}*head=NULL,*nn,*ptr,*qtr;
//main function
int main(){
	printf("Hello World!\n");
	read();
	display();
	rank();
}
//Function to read the student details
int read(){
	int i=1;
	printf("Enter the student details :\n");
	while(i==1){
		nn = (struct student *)malloc(sizeof(struct student));
		printf("Enter the student name :");
		scanf("%s",nn->name);
		printf("Enter the student marks :");
		scanf("%d",&nn->marks);
		if(head==NULL){
			head = nn;
			nn->next = NULL;
		}else{
			ptr = head;
			while(ptr->next!=NULL){
				ptr = ptr->next;
			}
			ptr->next = nn;
			nn->next = NULL;
		}
		printf("Press 0 to exit :");
		scanf("%d",&i);
	}
	return 0;
}
// Function to display the student details
int display(){
	int i = 0;
	ptr = head;
	while(ptr!=NULL){
		printf("The student %d name is %s ",i,ptr->name);
		printf("and marks are %d \n",ptr->marks);
		ptr = ptr->next;
		i++;
	}
	return 0;
}
//Function to display the rank
int rank(){
	int fc;
```

```
        printf("Enter the required mark to get first class :");
        scanf("%d",&fc);
        ptr = head;
        while(ptr!=NULL){
                if(ptr->marks>=fc){
                        printf("%s got the first class \n",ptr->name);
                }
                ptr = ptr->next;
        }
        return 0;
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A2\N190750_A2_P7.c.exe
Hello World!
Enter the student detalis :
Enter the student name :Moji
Enter the student marks :91
Press 0 to exit :1
Enter the student name :Mahi
Enter the student marks :86
Press 0 to exit :1
Enter the student name :Vamsi
Enter the student marks :78
Press 0 to exit :1
Enter the student name :Narendra
Enter the student marks :94
Press 0 to exit :0
The student 0 name is Moji and marks are 91
The student 1 name is Mahi and marks are 86
The student 2 name is Vamsi and marks are 78
The student 3 name is Narendra and marks are 94
Enter the required mark to get first class :91
Moji got the first class
Narendra got the first class

--------------------------------
Process exited after 78.14 seconds with return value 0
Press any key to continue . . .
```

LAB-3

```
/*              Author Name : N.Mariya Babu
                        Id  No. : N190750
        Assignment Number : 3
        programme Number : 1
  programme Description : C Programme to perform push,pop and peek operations on a stack
using arrays
                        Date : 7/12/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work
of any other person     */
//Header file's section
#include<stdio.h>
#include<stdlib.h>
//Global declaration
#define N 20
int Stack[N];
int top = -1;
```

```
//Function declaration section
int push(void);
int pop(void);
int peek(void);
int display(void);
//Main function
int main(){
	while(1){
		int n;
		printf("\n1 push 2 pop  3peek element 4 exit\n");
		printf("Enter your option :");
		scanf("%d",&n);
		switch(n){
			case 1:
				push();
				break;
			case 2:
				pop();
				break;
			case 3:
				peek();
				break;
			case 4:
				exit(1);
				break;
			default:
				printf("Invalid Syntax");
		}
		display();
	}
}
//Function to read the element's
int push(){
	int i=1;
	while(i !=0){
		if(top==N-1){
			printf("Over Flow");
		}else{
			int n;
			top++;
			printf("Enter the element :");
			scanf("%d",&n);
			Stack[top] = n;
		}
		printf("Press any key to continue 0 to exit");
		scanf("%d",&i);
	}
	return 0;
}
//Function to pop the top element
int pop(){
	if(top==-1){
		printf("Under Flow");
	}else{
		int item;
		item = Stack[top];
		printf("The popped item is %d\n",item);
```

```
                top--;
        }
        return 0;
}
//Function to display the peek element
int peek(){
        printf("The peek element is %d\n",Stack[top]);
}
//Function to display the element's
int display(){
        int i;
        if(top== -1){
                printf("Under Flow\n");
        }else{
                for(i=0;i<=top;i++){
                        printf("%d ",Stack[i]);
                }
                printf("\n");
        }
        return 0;
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A3\N190750_A3_P1.exe

1push 2pop 3peek element 4exit
Enter your option :1
Enter the element :5
Press any key to continue 0 to exit1
Enter the element :5
Press any key to continue 0 to exit1
Enter the element :9
Press any key to continue 0 to exit1
Enter the element :6
Press any key to continue 0 to exit0
5 5 9 6

1push 2pop 3peek element 4exit
Enter your option :2
The poped item is 6
5 5 9

1push 2pop 3peek element 4exit
Enter your option :3
The peek element is 9
5 5 9

1push 2pop 3peek element 4exit
Enter your option :4

--------------------------------
Process exited after 43.18 seconds with return value 1
Press any key to continue . . .
```

/* **Author Name : N.Mariya Babu**
Id No. : N190750
Assignment Number : 3
programme Number : 2

programme Description : C Programme to implement stack using linked list
Date : 7/12/2022 (MM/DD/YYYY)
certification : I hereby certify that this work is my own and none of it is the work of any other person */

```
//Header file's section
#include<stdio.h>
#include<stdlib.h>
//Structure Definition section
struct stack{
	int data;
	struct stack *next;
}*head=NULL,*ptr,*qtr,*nn,*top,*temp=NULL;
//Function declaration section
int push(void);
int pop(void);
int peek(void);
int display(void);
//Main function
int main(){
	while(1){
		int n;
		printf("\n1.push 2.pop 3.peek element 4.exit\n");
		printf("Enter your option :");
		scanf("%d",&n);
		switch(n){
			case 1:
				push();
				break;
			case 2:
				pop();
				break;
			case 3:
				peek();
				break;
			case 4:
				exit(1);
				break;
			default:
				printf("Invalid Syntax");
		}
		display();
	}
	return 0;
}
//Function to read the element's
int push(){
	int i=1;
	while(i){
		nn = (struct stack *)malloc(sizeof(struct stack));
		printf("Enter the data :");
		scanf("%d",&nn->data);
		nn->next = NULL;
		//top = nn;
		if(head==NULL){
			head = nn;
			top = nn;
		}else{
```

```
                        ptr = top;
                        top = nn;
                        top->next = ptr;
                }
                printf("Press 0 to exit : ");
                scanf("%d",&i);
        }
        return 0;
}
//Function to pop the top element
int pop(){
        int item;
        item = top->data;
        temp = top;
        top = top->next;
        free(temp);
        printf("The popped item is %d\n",item);
        return item;
}
//Function to display the peek element
int peek(){
        printf("The peek element is %d\n ",top->data);
        return top->data;
}
//Function to display the elements in the stack
int display(){
        temp = top;
        while(temp!=NULL){
                printf("%d\t",temp->data);
                temp = temp->next;
        }
        return 0;
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A3\N190750_A3_P2.exe

1.push 2.pop 3.peek element 4.exit
Enter your option :1
Enter the data :8
Press 0 to exit : 1
Enter the data :3
Press 0 to exit : 1
Enter the data :7
Press 0 to exit : 0
7       3       8
1.push 2.pop 3.peek element 4.exit
Enter your option :2
The poped item is 7
3       8
1.push 2.pop 3.peek element 4.exit
Enter your option :3
The peek element is 3
 3      8
1.push 2.pop 3.peek element 4.exit
Enter your option :4

--------------------------------
Process exited after 28.05 seconds with return value 1
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                        Id  No. : N190750
     Assignment Number : 3
     programme Number : 3
 programme Description : C Programme to reverse the contents of stack
                        Date : 7/12/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work
of any other person          */
//Header Files
#include<stdio.h>
#include<stdlib.h>
#define N 20
//Stack definition
int stack[N];
int top = -1;
//Function to push the element's into the stack
int push(int x){
        if(top==N-1){
                return 0;
        }
        stack[++top] = x;
}
//Function to pop the element from the stack
int pop(){
        if(top==-1){
                return 0;
        }
        return stack[top--];
}
//Function to display the stack content
int display(){
        int i;
        for(i=top;i>=0;i--){
                printf("%d  ",stack[i]);
        }
        return 0;
}
//Function to display the peek element
int peek(){
        printf("\n The peek element : %d\n",stack[top]);
}
//Queue definition
int queue[N];
int front=-1,rare=-1;
//Function to insert the data into the queue
int enqueue(int x){
        if(front==-1 && rare==-1){
                front=rare=0;
                queue[rare] = x;
        }
        else if((rare+1)%N==front){
                return 0;
        }
        else{
                rare = (rare+1)%N;
                queue[rare] = x;
        }
```

```
}
//Function to delete the element from the stack
int dequeue(){
	int item;
	if(front==-1 && rare==-1){
		return -1;
	}
	else if(front==rare){
		item = queue[front];
		front=rare=-1;
		return item;
	}
	else{
		item = queue[front];
		front = (front+1)%N;
		return item;
	}
}
//Function to reverse the stack
int stack_reversing(){
	while(top>=0){
		int x  = pop();
		enqueue(x);
	}
	while(rare!=-1){
		push(dequeue());
	}
}
//Main Function
int main(){
	int data;
	int opt;
	while(1){
		printf("\n1.push 2.pop 3.reverse stack");
		printf(" 4.peek 5.exit\n");
		printf("Choose your option :");
		scanf("%d",&opt);
		if(opt==1){
			printf("Enter the data :");
			scanf("%d",&data);
			push(data);
		}else if(opt==2){
			pop();
		}else if(opt==3){
			stack_reversing();

		}else if(opt==4){
			peek();
		}else if(opt==5){
			exit(1);
		}
		display();
	}
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A3\N190750_A3_P3.exe

1.push 2.pop 3.reverse stack 4.peek 5.exit
Choose your option :1
Enter the data :2
2
1.push 2.pop 3.reverse stack 4.peek 5.exit
Choose your option :1
Enter the data :6
6  2
1.push 2.pop 3.reverse stack 4.peek 5.exit
Choose your option :1
Enter the data :9
9  6  2
1.push 2.pop 3.reverse stack 4.peek 5.exit
Choose your option :2
6  2
1.push 2.pop 3.reverse stack 4.peek 5.exit
Choose your option :3
2  6
1.push 2.pop 3.reverse stack 4.peek 5.exit
Choose your option :4

 The peek element : 2
2  6
1.push 2.pop 3.reverse stack 4.peek 5.exit
Choose your option :5

--------------------------------
Process exited after 33.82 seconds with return value 1
Press any key to continue . . .
```

```
/*          Author Name : N.Mariya Babu
                  Id  No. : N190750
    Assignment Number : 3
    programme Number : 4
 programme Description : C Programme to check the nesting of parentheses
                  Date : 7/12/2022 (MM/DD/YYYY)
            certification : I hereby certify that this work is my own and none of it is the work
of any other person      */
//Header Files
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
//Macros
#define MAX 30
//Stack Definition
int stack[MAX];
int top=-1;
//Function definition
void push(char);
char pop();
int match(char a,char b);
int check(char []);

int main()
{
   char exp[MAX];
```

```
   int valid;
   printf("Enter an algebraic expression : ");
   gets(exp);
   valid=check(exp);
   if(valid==1)
      printf("Valid expression\n");
   else
      printf("Invalid expression\n");
           return 0;

}
int check(char exp[] ){
   int i;
   char temp;
   for(i=0;i<strlen(exp);i++){
      if(exp[i]=='(' || exp[i]=='{' || exp[i]=='[')
        push(exp[i]);
      if(exp[i]==')' || exp[i]=='}' || exp[i]==']')
         if(top==-1){    /*stack empty*/
            printf("Right parentheses are more \n");
            return 0;
         }
         else{
            temp=pop();
            if(!match(temp, exp[i])){
               printf("Mismatched parentheses are : ");
               printf("%c and %c\n",temp,exp[i]);
               return 0;
            }
         }
   }
   if(top==-1){
            /*stack empty*/
      printf("Balanced Parentheses\n");
      return 1;
   }
   else{
      printf("Left parentheses are more \n");
      return 0;
   }
}/*End of main()*/
int match(char a,char b){
   if(a=='[' && b==']')
      return 1;
   if(a=='{' && b=='}')
      return 1;
   if(a=='(' && b==')')
      return 1;
   return 0;
}/*End of match()*/

void push(char item){
   if(top==(MAX-1)){
      printf("Stack Overflow\n");
      return;
   }
   top=top+1;
```

```
   stack[top]=item;
}/*End of push()*/

char pop(){
   if(top==-1){
      printf("Stack Underflow\n");
      exit(1);
   }
   return(stack[top--]);
}/*End of pop()*/
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A3\N190750_A3_P4.exe
Enter an algebraic expression : (a+b){c-d}([jd*ds]){}()
Balanced Parentheses
Valid expression

--------------------------------
Process exited after 39.45 seconds with return value 0
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                        Id  No. : N190750
     Assignment Number : 3
     programme Number : 5_1
 programme Description : C Programme to convert infix to postfix
                        Date : 7/12/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work
of any other person        */
//Header Files section
#include<stdio.h>
#include<ctype.h>
//Definition section
# define N 20
//stack declaration
char stack[N];
int top = -1;
//Function to push the element
int push(char op){
        stack[++top] = op;
}
//Function to pop the element
char pop(){
        if(top==-1){
                return -1;
        }
        return stack[top--];
}
//Functio for presidence
int prec(char op){
        if(op=='('){
                return 0;
        }
        if(op=='+'||op=='-'){
                return 1;
```

```
        }
        if(op=='*'||op=='/'){
                return 2;
        }
        if(op=='^'){
                return 3;
        }
        return 0;
}
//Function to to perform infix to postfix
char* Infix_To_Postfix(char *exp){
        int k=0;
        static char postfix[50];
        char *e;
        e = exp;
        while(*e!=NULL){
                if(isalnum(*e)){
                        postfix[k++] = *e;
                }
                else if(*e=='('){
                        push(*e);
                }
                else if(*e==')'){
                        while(stack[top]!='('){
                                postfix[k++] = pop();
                        }
                        pop();
                }
                else{
                         while(prec(stack[top]) >= prec(*e))
         postfix[k++] = pop();
       push(*e);
                }
                e++;
        }
        while(top!=-1){
                postfix[k++] = pop();
        }
        printf("The postfix exp is %s ",postfix);
        return postfix;
}
//Mani function
int main(){     char exp[50],*postfix;
        printf("Enter the infix expression :");
        scanf("%s",exp);
        postfix = Infix_To_Postfix(exp);
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A3\N190750_A3_P5_1.exe
Enter the infix expression :2+4-7(9*7)/4-3
The postfix exp is 24+797*4/-3-
--------------------------------
Process exited after 22.95 seconds with return value 0
Press any key to continue . . .
```

```
/*    Author Name : N.Mariya Babu
                Id  No. : N190750
     Assignment Number : 3
    programme Number : 5_2
 programme Description : C Programme to convert infix to prefix
                Date : 7/12/2022 (MM/DD/YYYY)
          certification : I hereby certify that this work is my own and none of it is the work of any
other person
*/
//Header Files section
#include<stdio.h>
#include<ctype.h>
#include<string.h>
//Definition section
# define N 20
//stack declaration
char stack[N];
int top = -1;
//Function to push the element
int push(char op){
        stack[++top] = op;
}
//Function to pop the element
char pop(){
        if(top==-1){
                return -1;
        }
        return stack[top--];
}
//Functio for presidence
int prec(char op){
        if(op==')'){
                return 0;
        }
        if(op=='+'||op=='-'){
                return 1;
        }
        if(op=='*'||op=='/'){
                return 2;
        }
        if(op=='^'){
                return 3;
        }
        return 0;
}
//Function to to perform infix to postfix
char* Infix_To_Prefix(char *exp){
        int k=0;
        static char postfix[50];
        char *e;
        e = exp;
        while(*e!=NULL){
                if(isalnum(*e)){
                        postfix[k++] = *e;
                }
                else if(*e==')'){
                        push(*e);
```

```
                }
                else if(*e=='('){
                        while(stack[top]!=')'){
                                postfix[k++] = pop();
                        }
                        pop();
                }
                else{
                         while(prec(stack[top]) > prec(*e))
        postfix[k++] = pop();
      push(*e);
                }
                e++;
        }
        while(top!=-1){
                postfix[k++] = pop();
        }
        //printf("\nThe postfix exp is %s ",postfix);
        return postfix;
}
//Mani function
int main(){
        char exp[50],*prefix;
        printf("Enter the infix expression :");
        gets(exp);
        prefix = Infix_To_Prefix(strrev(exp));
        prefix = strrev(prefix);
        printf("\nThe prefix expression is : \n%s",prefix);
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A3\N190750_A3_P5_2.exe
Enter the infix expression :3+9-0*7(9-0)+4/94

The prefix expression is :
+-+39*07-90/494
--------------------------------
Process exited after 3.686 seconds with return value 0
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                        Id  No. : N190750
      Assignment Number : 3
      programme Number : 6
 programme Description : C Programme to evaluate a postfix and prefix expressions
                        Date : 7/12/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work of any
other person
*/
//Header Files
#include<stdio.h>
#include<ctype.h>
#include<string.h>
#define n 50
//Stack definition
int stack[n];
int top = -1,b;
```

```
//Function to push the element's into the stack
int push(int x){
	stack[++top] = x;
}
//Function to pop the element's from the stack
int pop(){
	if(top==-1){
		return -1;
	}
	return stack[top--];
}
//Function to evaluate the postfix expression
int Postfix_Evaluation(char *exp){
	int num,n1,n2,n3;
	char *e;
	e = exp;
	while(*e!=NULL){
		if(isdigit(*e)){
			num = *e-48;
		//	printf("%d",num);
			push(num);
		}
		else{
			if(b==2){
			n2 = pop();
			n1 = pop();
			}
			else{
			n1 = pop();
			n2 = pop();
			}
			switch(*e){
				case '+':
					n3 = n1+n2;
					break;
				case '-':
					n3 = n2 - n1;
					break;
				case '*':
					n3 = n2 * n1;
					break;
				case '/':
					n3 = n2/n1;
					break;
				case '^':
					n3 = n1^n2;
					break;
			}
			push(n3);
		}
		e++;
	}
	int x = pop();
	return x;
}
//Main Function
int main(){
```

```
        char exp[50];
        int result=1;
        printf("Enter 1.postfix expression 2.prefix expression\n");
        scanf("%d",&b);
        if(b==1){
                printf("Enter the postfix expression :");
                scanf("%s",exp);
                result = Postfix_Evaluation(exp);
        }
        else if(b==2){
                printf("Enter the prefix expression :");
                scanf("%s",exp);
                result = Postfix_Evaluation(strrev(exp));
        }
        printf("The result of the expression is %d",result);
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A3\N190750_A3_P6.exe
Enter 1.postfix expression 2.prefix expression
2
Enter the prefix expression :-+-+39*92/4567
The result of the expression is -12
--------------------------------
Process exited after 6.435 seconds with return value 0
Press any key to continue . . .
```

LAB-4

```
/*                      Author Name : N.Mariya Babu
                              Id  No. : N190750
        Assignment Number : 4
        programme Number : 1
        programme Description : C Programme to implement queue using linked list
                              Date : 7/18/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work of any
        other person        */
        #include<stdio.h>
        #include<stdlib.h>
        //Structure definition
        struct queue{
                int data;
                struct queue *next;
        }*head=NULL,*tail=NULL,*nn,*ptr;
        //Enqueue Operation
        int enqueue(){
                nn = (struct queue*) malloc (sizeof(struct queue));
                printf("Enter the node data :");
                scanf("%d",&nn->data);
                if(head==NULL && tail==NULL){
                        head=tail=nn;
```

```
		nn->next = NULL;
	}
	else{
		tail->next=nn;
		tail=nn;
		nn->next = NULL;
	}
}
//Dequeue Function
int dequeue(){
	int item = head->data;
	if(head==tail){
		head=tail=NULL;
		return item;
	}
	else{
		ptr = head;
		head=head->next;
		return item;
	}

}
//Display Function
int display(){
	ptr = head;
	while(ptr!=NULL){
		printf("%d\t",ptr->data);
		ptr = ptr->next;
	}
	return 0;
}
//Main Function
int main(){
	int opt;
	printf("Queue implementation using linked list");
	while(1){
		printf("\n1.enqueue 2.dequeue 3.exit ");
		printf("\nEnter your option :");
		scanf("%d",&opt);
		switch(opt){
			case 1:
				enqueue();
				break;
			case 2:
				dequeue();
				break;
			case 3:
				exit(1);
			default:
				printf("Invalid Option....");
		}
		display();
	}
	return 0;
}
```

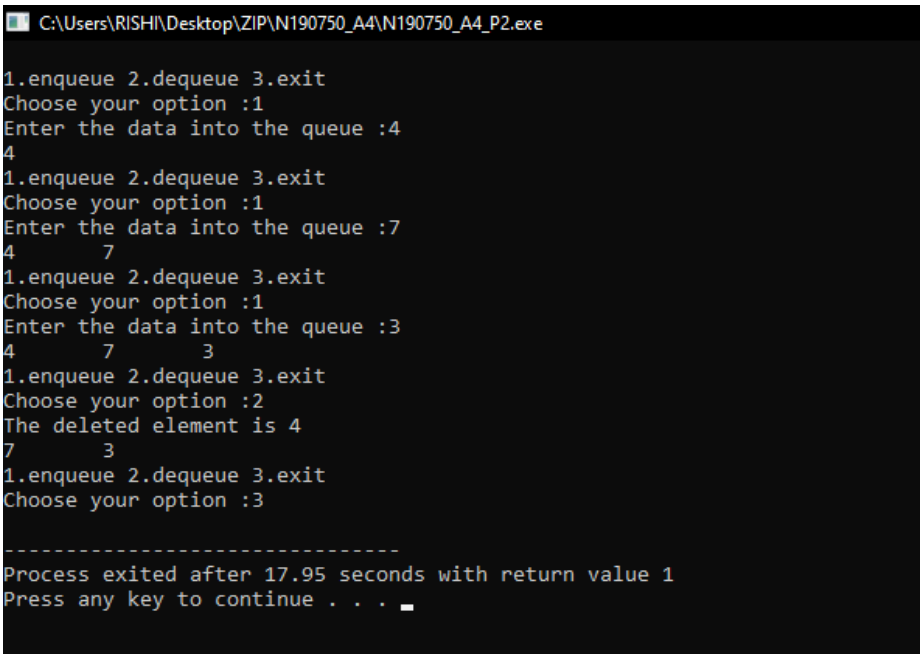
Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A4\N190750_A4_P1.exe
Queue implimintation using linked list
1.enqueue 2.dqueue 3.exit
Enter your option :1
Enter the node data :3
3
1.enqueue 2.dqueue 3.exit
Enter your option :1
Enter the node data :6
3       6
1.enqueue 2.dqueue 3.exit
Enter your option :1
Enter the node data :8
3       6       8
1.enqueue 2.dqueue 3.exit
Enter your option :2
6       8
1.enqueue 2.dqueue 3.exit
Enter your option :3

--------------------------------
Process exited after 21.1 seconds with return value 1
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                        Id  No. : N190750
      Assignment Number : 4
      programme Number : 2
 programme Description : C Programme to implement queue using array
                        Date : 7/18/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work
of any other person        */
#include<stdio.h>
#include<stdlib.h>
#define N 20
//Queue definition
int queue[N];
int rare=-1,front=-1;
//Enqueue Function
int enqueue(int x){
        if(front==-1 && rare==-1){
                front=rare=0;
                queue[rare] = x;
        }
        else if(rare==N-1){
                printf("Overflow..");
        }
        else{
                queue[++rare] = x;
        }
        return 0;
}
//Dequeue Function
int dequeue(){
        int item = queue[front];
        if(front==-1 && rare==-1){
                printf("UnderFlow..");
```

```
		return -1;
	}
	else if(front==rare){
		front=rare=-1;
		return item;
	}else{
		front++;
		return item;
	}

}
//Display Function
int display(){
	int i;
	for(i=front;i<=rare;i++){
		printf("%d\t",queue[i]);
	}
	return 0;
}
//Main Function
int main(){
	int opt,data;
	while(1){
		printf("\n1.enqueue 2.dequeue 3.exit \n");
		printf("Choose your option :");
		scanf("%d",&opt);
		if(opt==1){
			printf("Enter the data into the queue :");
			scanf("%d",&data);
			enqueue(data);
		}
		else if(opt==2){
			printf("The deleted element is %d \n",dequeue());
		}
		else if(opt==3){
			exit(1);
		}
		else{
			printf("Invalid...");
		}
		display();
	}
}
```

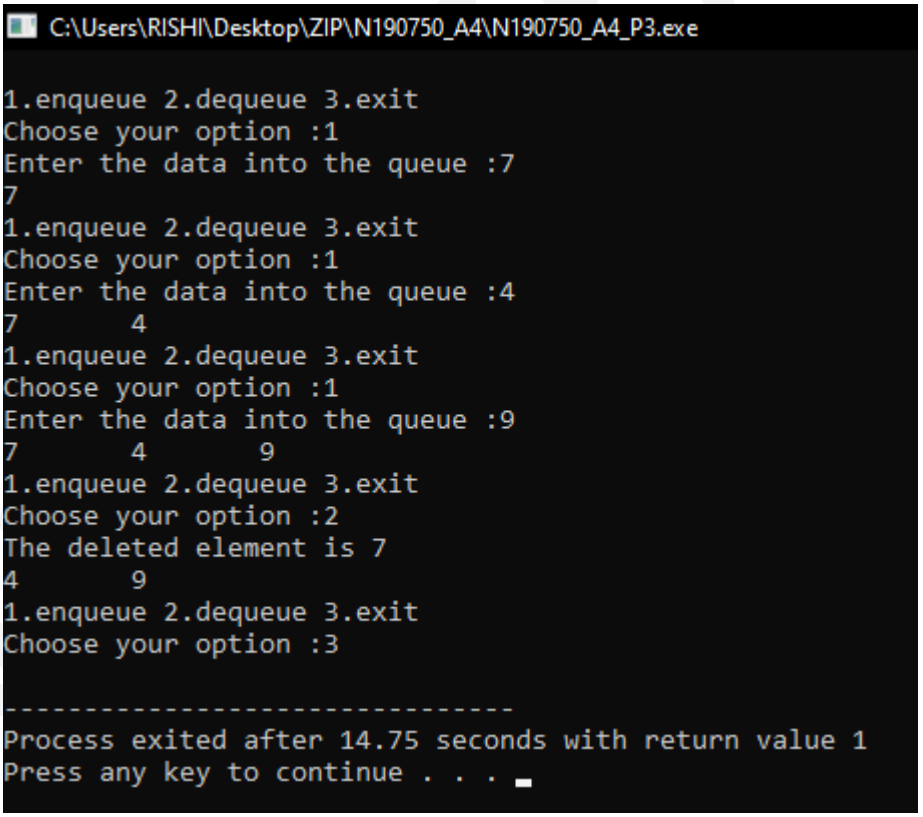
Output:



```
/*              Author Name : N.Mariya Babu
                      Id  No. : N190750
      Assignment Number : 4
      programme Number : 3
 programme Description : C Programme to implement queue using circular array
                        Date : 7/18/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work of any
other person      */
#include<stdio.h>
#include<stdlib.h>
#define N 20
//Queue definition
int queue[N];
int rare=-1,front=-1;
//Enqueue Function
int enqueue(int x){
        if(front==-1 && rare==-1){
                front=rare=0;
                queue[rare] = x;
        }
        else if((rare+1)%N==front){
                printf("Overflow..");
        }
        else{
                rare = (rare+1)%N;
                queue[rare] = x;
        }
        return 0;
}
//Dequeue Function
int dequeue(){
        int item = queue[front];
        if(front==-1 && rare==-1){
                printf("UnderFlow..");
                return -1;
        }
        else if(front==rare){
                front=rare=-1;
                return item;
        }else{
                front = (front+1)%N;
                return item;
        }

}
//Display Function
int display(){
        int i=front;
        while(i!=(rare+1)){
                printf("%d\t",queue[i]);
                i = (i+1)%N;
        }
        return 0;
}
//Main Function
int main(){
        int opt,data;
```

```
        while(1){
                printf("\n1.enqueue 2.dequeue 3.exit\n");
                printf("Choose your option :");
                scanf("%d",&opt);
                if(opt==1){
                        printf("Enter the data into the queue :");
                        scanf("%d",&data);
                        enqueue(data);
                }
                else if(opt==2){
                        printf("The deleted element is %d \n",dequeue());
                }
                else if(opt==3){
                        exit(1);
                }
                else{
                        printf("Invalid...");
                }
                display();
        }
}
```

Output:




```
/*              Author Name : N.Mariya Babu
                        Id  No. : N190750
      Assignment Number : 4
      programme Number : 4
  programme Description : C Programme to implement queue which permits insertion and deletion at both the ends
                        Date : 7/18/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work of any other person      */
//Header Files
#include<stdio.h>
#include<stdlib.h>
```

```
#define N 5
//Queue definition
int queue[N];
int front=-1,rare=-1;
//Enqueue from the front of the queue
int EnqueueFront(int x = 10){
	if(front==-1 && rare==-1){
		front=rare=0;
		queue[front] = x;
	}
	else if(front==0){
		if(rare==N-1){
			printf("OverFlow");
		}
		else{
			front=N-1;
			queue[front] = x;
		}
	}
	else {
		if(((front-1)%N)==rare){
			printf("OverFlow");
		}
		else{
			front = (front-1)%N;
			queue[front] = x;
		}
	}
}
//Enqueue from the rare of the queue
int EnqueueRare(int x = 5){
	if(front==-1 && rare==-1){
		rare = front = 0;
		queue[rare] = x;
	}
	else if((rare+1)%N==front){
		printf("Overflow..");
	}
	else{
		rare = (rare+1)%N;
		queue[rare] = x;
	}
	return 0;
}
//Dequeue the element from the front
int DequeueFront(){
	int item;
	item = queue[front];
	if(front==-1 && rare==-1){
		printf("Under Flow...");
	}
	else if(front==rare){
		printf("\nThe Dequeued item is : %d  ",item);
		front=rare=-1;
		return item;
	}
	else{
```

```
		front = (front+1)%N;
		printf("\nThe Dequeued item is %d : \n",item);
		return item;
	}
}
//Dequeue the element from the rare of the queue
int DequeueRare(){
	int item;
	item = queue[rare];
	if(front==-1 && rare==-1){
		printf("UnderFlow...");
	}
	else if(front==rare){
		front=rare=-1;
		printf("\nThe Dequeued item is %d : \n",item);
		return item;

	}
	else if(rare==0){
		rare = N-1;
		printf("\nThe Dequeued item is %d : \n",item);
		return item;
	}
	else{
		rare = (rare-1)%N;
		printf("\nThe Dequeued item is %d : \n",item);
		return item;
	}

}
//Display the content of the queue
int display(){
	int i;
	i = front;
	while(i!=rare){
		printf("%d\t",queue[i]);
		i = (i+1)%N;
	}
	printf("%d\t",queue[i]);
	return 0;
}
//Main Function
int main(){
	int opt,data;
	printf("Double Ended queue data structure ");
	while(1){
		printf("\n1.EnqueueFront 2.EnqueueRare ");
		printf(" 3.DequeueFront 4.DequeueRare 5.exit \n");
		printf("Enter your option :");
		scanf("%d",&opt);
		if(opt==1){
			printf("Enter data to insert from  front of the queue :");
			scanf("%d",&data);
			EnqueueFront(data);
		}
		else if(opt==2){
			printf("Enter data to insert from rare of the queue :");
```

```
                scanf("%d",&data);
                EnqueueRare(data);
            }
            else{
                switch(opt){
                    case 3:
                        DequeueFront();
                        break;
                    case 4:
                        DequeueRare();
                        break;
                    case 5:
                        exit(1);
                    default:
                        printf("Invalid option...");

                }
            }
            display();
        }
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A4\N190750_A4_P4.exe
Double Ended queue data structure
1.EnqueueFront 2.EnqueueRare  3.DequeueFront 4.DequeueRare 5.exit
Enter your option :1
Enter data to insert from  front of the queue :4
4
1.EnqueueFront 2.EnqueueRare  3.DequeueFront 4.DequeueRare 5.exit
Enter your option :2
Enter data to insert from rare of the queue :7
4       7
1.EnqueueFront 2.EnqueueRare  3.DequeueFront 4.DequeueRare 5.exit
Enter your option :3

The Dequeued item is 4 :
7
1.EnqueueFront 2.EnqueueRare  3.DequeueFront 4.DequeueRare 5.exit
Enter your option :4

The Dequeued item is 7 :
0
1.EnqueueFront 2.EnqueueRare  3.DequeueFront 4.DequeueRare 5.exit
Enter your option :5

--------------------------------
Process exited after 17.67 seconds with return value 1
Press any key to continue . . .
```

LAB-5

```
/*    Author Name : N.Mariya Babu
                Id  No. : N190750
    Assignment Number : 5
    programme Number : 1
 programme Description : C Programme to find an Element using LinearSearch
                Date : 7/19/2022 (MM/DD/YYYY)
            certification : I hereby certify that this work is my own and none of it is the work of any
other person        */
//Header Files
#include<stdio.h>
//Function definition section
int read(int*,int);
int LinearSearch(int*,int,int);
//Main function
int main(){
        int n,key;
        printf("LinearSearch implementation....\n");
        printf("Enter the size of the array :");
        scanf("%d",&n);
        int arr[n];
        read(arr,n);
        printf("Enter the search element :");
        scanf("%d",&key);
        LinearSearch(arr,key,n);
}
//Read Function
int read(int arr[],int n){
        for(int i=0;i<n;i++){
                printf("a[%d] :",i);
                scanf("%d",&arr[i]);
        }
}
//LinearSearch Function
int LinearSearch(int arr[],int key,int n){
        int i,flag=0;
        for(i=0;i<n;i++){
                if(arr[i]==key){
                        printf("Search key is found at the index : %d",i);
                        flag++;
                        return i;
                }
        }
        if(flag==0){
                printf("Element not present...");
                return -1;
        }
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A5\N190750_A5_P1.exe
LinearSearch implementation....
Enter the size of the array :5
a[0] :1
a[1] :5
a[2] :2
a[3] :9
a[4] :7
Enter the search element :2
Seach key is found at the index : 2
--------------------------------
Process exited after 17.8 seconds with return value 0
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                      Id  No. : N190750
      Assignment Number : 5
      programme Number : 2
 programme Description : C Programme to find an Element using Binary Search without recursion
                      Date : 7/19/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work of any
other person     */
//Header Files
#include<stdio.h>
//Function  definition section
int read(int*,int);
int BinarySearch(int*,int,int);
//Main function
int main(){
        int n,key;
        printf("...BinarySearch without using the recursion \n");
        printf("Enter the size of the array :");
        scanf("%d",&n);
        int arr[n];
        read(arr,n);
        printf("Enter the searching element :");
        scanf("%d",&key);
        BinarySearch(arr,key,n);
}
//Read Function
int read(int arr[],int n){
        int i;
        printf("..Enter Elements  in sorted order..");
        for(i=0;i<n;i++){
                printf("arr[%d] :",i);
                scanf("%d",&arr[i]);
        }
}
//Binary Search
int BinarySearch(int arr[],int key,int n){
        int start=0,end=n;
        int flag = 0;
        while(start<=end){
                int mid = (start+end)/2;
                if(arr[mid]==key){
                        printf("Search element is found at the index : %d",mid);
                        flag++;
                        return mid;
                }
                else if(arr[mid]<key){
                        start = mid+1;
                }
                else{
                        end = mid-1;
                }
        }
        if(flag==0){
                        printf("Search element is not found....");
        }
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A5\N190750_A5_P2.exe
...BinarySearch without using the recursion
Enter the size of the array :5
..Enter Element's  in sorted order..arr[0] :1
arr[1] :3
arr[2] :5
arr[3] :7
arr[4] :9
Enter the searching element :7
Search element is found at the index : 3
--------------------------------
Process exited after 22.5 seconds with return value 0
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                      Id  No. : N190750
        Assignment Number : 5
        programme Number : 3
 programme Description : C Programme to implement the BinarySearch using the Recursion
                      Date : 7/19/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work of any
other person     */
#include<stdio.h>
//Function definition section
int read(int*,int);
int BinarySearch(int*,int,int,int);
//Main function
int main(){
        int n,key;
        printf("...BinarySearch with using the recursion...\n");
        printf("Enter the size of the array :");
        scanf("%d",&n);
        int arr[n];
        read(arr,n);
        printf("Enter the search element :");
        scanf("%d",&key);
        BinarySearch(arr,0,n,key);
}
//Function to read the element's from the array
int read(int arr[],int n){
        int i;
        printf("..Enter elements in sorted order..");
        for(i=0;i<n;i++){
                printf("arr[%d] :",i);
                scanf("%d",&arr[i]);
        }
}
//Function to search the given element
int BinarySearch(int arr[],int start,int end,int key){
        int mid = (start+end)/2;
        if(start<=end){
                if(arr[mid]==key){
                        printf("The search key is found at the index : %d",mid);
                        return mid;
                }
                else if(arr[mid]<key){
```

```
                BinarySearch(arr,mid+1,end,key);
        }
        else{
                BinarySearch(arr,start,mid-1,key);
        }
    }
    else{
        printf("Element was not found...");
        return 0;
    }

}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A5\N190750_A5_P3.exe
...BinarySearch with using the recursion...
Enter the size of the array :6
..Enter element's in sorted order..arr[0] :1
arr[1] :3
arr[2] :6
arr[3] :8
arr[4] :14
arr[5] :17
Enter the search element :8
The search key is found at the index : 3
--------------------------------
Process exited after 18.11 seconds with return value 0
Press any key to continue . . .
```

LAB-6

```
/*
                Author Name : N.Mariya Babu
                        Id  No. : N190750
        Assignment Number : 6
  programme Description : 1
  programme Description : C Programme to implement Selection Sort
                        Date : 8/2/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work
of any other person
*/
//Header file section
#include<stdio.h>
//Function definition section
int read(int*,int);
int display(int*,int);
int SelectionSort(int*,int);
//Main function
int main(){
        printf(".....Selection Sort.....\n");
        int n;
        printf("Enter the size of the array :");
        scanf("%d",&n);
        int arr[n];
        read(arr,n);
        printf("Array before sorting \n");
```

```
        display(arr,n);
        printf("Array after sorting \n");
        SelectionSort(arr,n);
        display(arr,n);
}
//Function to read the array element's
int read(int arr[],int n){
        int i;
        printf("Enter the array element's \n");
        for(i=0;i<n;i++){
                printf("arr[%d] :",i);
                scanf("%d",&arr[i]);
        }
        return 0;
}
//Function to display the array element's
int display(int arr[],int n){
        int i;
        for(i=0;i<n;i++){
                printf("%d\t",arr[i]);
        }
        printf("\n");
}
//Function to implement the selection sort
int SelectionSort(int arr[],int n){
        int i,j,min_index,temp;
        for(i=0;i<n;i++){
                min_index = i;
                for(j=i+1;j<n;j++){
                        if(arr[min_index]>arr[j]){
                                min_index = j;
                        }
                }
                temp = arr[i];
                arr[i] = arr[min_index];
                arr[min_index] = temp;
        }

}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A5\N190750_A5_P1.exe
LinearSearch implementation....
Enter the size of the array :5
a[0] :1
a[1] :5
a[2] :2
a[3] :9
a[4] :7
Enter the search element :2
Seach key is found at the index : 2
--------------------------------
Process exited after 17.8 seconds with return value 0
Press any key to continue . . .
```

/* **Author Name : N.Mariya Babu**
Id No. : N190750
Assignment Number : 6
programme Description : 2

programme Description : C Programme to implement Merge Sort
Date : 8/2/2022 (MM/DD/YYYY)
certification : I hereby certify that this work is my own and none of it is the work of any other person */

```
//Headfiles section
#include<stdio.h>
//Function definition section
int read(int*,int);
int display(int*,int);
int MergeSort(int*,int,int);
int merge(int*,int,int,int);
//Main Function
int main(){
	printf(" MergeSort Technique \n");
	int n;
	printf("Enter the array size :");
	scanf("%d",&n);
	int arr[n];
	read(arr,n);
	printf("\nArray before sorting \n");
	display(arr,n);
	printf("\nArray after sorting \n");
	MergeSort(arr,0,n-1);
	display(arr,n);
}
//Read function
int read(int arr[],int n){
	int i;
	printf("\nEnter array element's :\n");
	for(i=0;i<n;i++){
		printf("arr[%d] : ",i);
		scanf("%d",&arr[i]);
	}
	return 0;
}
//Display function
int display(int arr[],int n){
	int i;
	for(i=0;i<n;i++){
		printf("%d\t",arr[i]);
	}
	printf("\n");
	return 0;
}
//MergeSort function
int MergeSort(int arr[],int s,int e){
	int mid = (s+e)/2;
	if(s<e){
		MergeSort(arr,s,mid);
		MergeSort(arr,mid+1,e);
		merge(arr,s,mid,e);
	}
	return 1;
}
//Merging the element's of the array
int merge(int arr[],int s,int mid,int e){
	int b[e];
```

```
        int i=s,j=mid+1,k=s;
        while(i<=mid && j<=e){
                if(arr[i]<arr[j]){
                        b[k++] = arr[i++];
                }
                else{
                        b[k++] = arr[j++];
                }
        }
        if(i>mid){
                while(j<=e){
                        b[k++] = arr[j++];
                }

        }
        else{
                        while(i<=mid){
                                b[k++] = arr[i++];
                        }
                }
        for(i=s;i<=e;i++){
                arr[i] = b[i];
        }
        return 0;
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A5\N190750_A5_P2.exe
...BinarySearch without using the recursion
Enter the size of the array :5
..Enter Element's  in sorted order..arr[0] :1
arr[1] :3
arr[2] :5
arr[3] :7
arr[4] :9
Enter the searching element :7
Search element is found at the index : 3
--------------------------------
Process exited after 22.5 seconds with return value 0
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                      Id  No. : N190750
      Assignment Number : 6
  programme Description : 3
  programme Description : C Programme to implement Quick Sort
                      Date : 8/2/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work of any
other person        */
//Header Files section
#include<stdio.h>
//Function definition section
int read(int*,int);
int display(int*,int);
int swap(int*,int*);
int QuickSort(int*,int,int);
int partition(int*,int,int);
```

```
//Main Function
int main(){
	printf(" QuickSort Technique \n");
	int n;
	printf("Enter the array size :");
	scanf("%d",&n);
	int arr[n];
	read(arr,n);
	printf("\nArray before sorting \n");
	display(arr,n);
	printf("\nArray after sorting \n");
	QuickSort(arr,0,n-1);
	display(arr,n);
}
//Read function
int read(int arr[],int n){
	int i;
	printf("\nEnter array element's :\n");
	for(i=0;i<n;i++){
		printf("arr[%d] : ",i);
		scanf("%d",&arr[i]);
	}
	return 0;
}
//Display function
int display(int arr[],int n){
	int i;
	for(i=0;i<n;i++){
		printf("%d\t",arr[i]);
	}
	return 0;
}
//Quick Sort function
int QuickSort(int arr[],int lb,int ub){
	int loc;
	if(lb<ub){
		loc = partition(arr,lb,ub);
		QuickSort(arr,lb,loc-1);
		QuickSort(arr,loc+1,ub);
	}
}
//Partition function
int partition(int arr[],int lb,int ub){
	int pivot=arr[lb];
	int s = lb;
	int e = ub;
	while(s<e){
		while(pivot>=arr[s]){
			s++;
		}
		while(pivot<arr[e]){
			e--;
		}
		if(s<e){
			swap(&arr[s],&arr[e]);
		}
	}
```

```
        swap(&arr[lb],&arr[e]);
        return e;
}
//Function to swap the given element's
int swap(int *ptr,int *qtr){
        int temp;
        temp = *ptr;
        *ptr = *qtr;
        *qtr = temp;
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A5\N190750_A5_P3.exe
...BinarySearch with using the recursion...
Enter the size of the array :6
..Enter element's in sorted order..arr[0] :1
arr[1] :3
arr[2] :6
arr[3] :8
arr[4] :14
arr[5] :17
Enter the search element :8
The search key is found at the index : 3
--------------------------------
Process exited after 18.11 seconds with return value 0
Press any key to continue . . .
```

LAB-7

```
/*              Author Name : N.Mariya Babu
                     Id  No. : N190750
        Assignment Number : 7(BST)
       programme Number : 1
  programme Description : C Programme to create BST and Insertion,deletion,3 traversal
                     Date : 8/23/2022
                certification : I hereby certify that this work is my own and none of it is the work
of any other person      */
//Description:-BST Operations
//HeaderFiles section
#include<stdio.h>
#include<stdlib.h>
//Structure definition
struct node{
        int data;
        struct node *lst;
        struct node *rst;
};
//Insert the data into the node
struct node * insert(struct node *root){
   int val;
   printf("enter value of node :");
   scanf("%d",&val);
        struct node *nn,*node,*parent;
        nn=(struct node*)malloc(sizeof(struct node));
        nn->data=val;
        nn->lst=0;
```

```
        nn->rst=0;
        if(root==0){
                root=nn;
                root->lst=0;
                root->rst=0;
        }
        else{
                parent=0;
                node=root;
                while(node!=0){
                        parent=node;
                        if(val<node->data)
                            node=node->lst;
                        else
                            node=node->rst;
                }
                if(val<parent->data)
                  parent->lst=nn;
                else
                  parent->rst=nn;
        }
        return root;
}
//Inorder traversal for the BST
void inorder(struct node *root){
      if(root!= NULL){
         inorder(root->lst);
         printf("%d ",root->data);
         inorder(root->rst);
      }
}
//Preorder traversal of the BST
void printPreorder(struct node* node){
   if (node == NULL)
   return;
   printf("%d ", node->data);
   printPreorder(node->lst);
   printPreorder(node->rst);
}
//Postorder traversal of the BST
void printPostorder(struct node* node){
   if (node == NULL)
      return;
   printPostorder(node->lst);
   printPostorder(node->rst);
   printf("%d ", node->data);
}
//Searching in the BST
struct node* search(struct node* node,int key){
   if(node==0 || node->data==key)
            return node;
        else if(key>node->data)
            return search(node->rst,key);
        else if(key<node->data)
        return search(node->lst,key);
   printf("element not found");
}
```

```
//Maximum element in the tree
struct node* max(struct node *node){
        if(node==0 || node->rst==0)
          return node;
        else
          return max(node->rst);
}
//Minimum element in the tree
struct node* min(struct node *node){
        if(node==0 || node->lst==0)
          return node;
        else
          return min(node->lst);
}
//Delete the data from the BST
 struct node *delete(struct node *root,int x){
   struct node *temp;
        if(root==0){
         printf("no data to delete");
                return root;
         }
        if(x<root->data){
                root->lst=delete(root->lst,x);
        }
        else if(x>root->data){
                root->rst=delete(root->rst,x);
        }
        else{
                if(root->lst==0){
                        temp=root->rst;
                        free(root);
                        return temp;
                }
                else if(root->rst==0){
                        temp=root->lst;
                        free(root);
                        return temp;
                }
                else if((root->lst==0 )&& (root->rst==0))
                    free(root);
                    temp=min(root->rst);
                    root->data=temp->data;
                    root->rst=delete(root->rst,temp->data);
                        }
        return root;
 }
 //Main Function
int main(){
        int val,n,i=0,key,x;
        struct node *root=0,*mini,*maxi;
        printf("enter no. of nodes : ");
        scanf("%d",&n);
        while(i<n){
                root=insert(root);
                i++;
        }
        printf("\nInorder Traversal \n");
```

```
        inorder(root);
        printf("\nPreorder traversal \n");
        printPreorder(root);
        printf("\Postorder traversal \n");
        print Postorder(root);
        mini=min(root);
        printf("\nsmallest element is %d\n",mini->data);
        maxi=max(root);
        printf("largest element is %d\n",maxi->data);
        printf("enter element to delete : ");
        scanf("%d",&x);
        delete(root,x);
        printf("\nInorder traversal After deletion \n");
        inorder(root);
        return 0;
    }
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A7\N190750_A7_P1.exe
enter no. of nodes : 4
enter value of node :1
enter value of node :2
enter value of node :5
enter value of node :8

Inorder Traversal
1 2 5 8
Preorder traversal
1 2 5 8
Postorder traversal
8 5 2 1
smallest element is 1
largest element is 8
enter element to delete : 5

Inorder traversal After deletion
1 2 8
--------------------------------
Process exited after 18.66 seconds with return value 0
Press any key to continue . . .
```

```
/*          Author Name : N.Mariya Babu
                 Id  No. : N190750
      Assignment Number : 7(BST)
  programme Description : 2
  programme Description :C Programme to count the number nodes of a BST
                   Date : 8/23/2022
            certification : I hereby certify that this work is my own and none of it is the work of any
other person    */
//
//HeaderFiles section
#include<stdio.h>
#include<stdlib.h>
//Structure Definition
struct node{
        int data;
        struct node *lst;
        struct node *rst;
};
//Function to insert the data into the BST
```

```
struct node * insert(struct node *root){
   int val;
   printf("enter value of node :");
   scanf("%d",&val);
        struct node *newnode,*node,*parent;
        newnode=(struct node*)malloc(sizeof(struct node));
        newnode->data=val;
        newnode->lst=0;
        newnode->rst=0;
        if(root==0){
                root=newnode;
                root->lst=0;
                root->rst=0;
        }
        else{
                parent=0;
                node=root;
                while(node!=0){
                        parent=node;
                        if(val<node->data)
                            node=node->lst;
                        else
                            node=node->rst;
                }
                if(val<parent->data)
                  parent->lst=newnode;
                else
                  parent->rst=newnode;
        }
        return root;
}
//Function to find the total no.of nodes in the BST
int totalnodes(struct node *root){
        if(root==0)
          return 0;
        else
         return(totalnodes(root->lst)+totalnodes(root->rst)+1);
}
//Main Function
int main(){
        int val,n,i=0,choice=1;
        struct node *root=0;
        while(choice){
                root=insert(root);
                i++;
                printf("enter 0 to stop :");
                scanf("%d",&choice);
        }
        printf("total no. of nodes %d",totalnodes(root));
        return 0;
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A7\N190750_A7_P2.exe
enter value of node :4
enter 0 to stop :1
enter value of node :3
enter 0 to stop :1
enter value of node :7
enter 0 to stop :1
enter value of node :6
enter 0 to stop :1
enter value of node :9
enter 0 to stop :0
total no. of nodes 5
--------------------------------
Process exited after 16.82 seconds with return value 0
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                        Id  No. : N190750
      Assignment Number : 7(BST)
 programme Description : 3
 programme Description : C Programme to find the nth node in the inOrder traversal of a BST
                        Date : 8/23/2022
                 certification : I hereby certify that this work is my own and none of it is the work of any
other person        */
//HeaderFiles section
#include<stdio.h>
#include<stdlib.h>
//Structure Definition
struct node{
        int data;
        struct node *lst;
        struct node *rst;
};
//Function to insert the data into the BST
struct node * insert(struct node *root){
   int val;
   printf("enter value of node :");
   scanf("%d",&val);
        struct node *newnode,*node,*parent;
        newnode=(struct node*)malloc(sizeof(struct node));
        newnode->data=val;
        newnode->lst=0;
        newnode->rst=0;
        if(root==0){
                root=newnode;
                root->lst=0;
                root->rst=0;
        }
        else{
                parent=0;
                node=root;
                while(node!=0){
                        parent=node;
                        if(val<node->data)
                           node=node->lst;
                        else
                           node=node->rst;
```

```
                                        }
                if(val<parent->data)
                  parent->lst=newnode;
                else
                  parent->rst=newnode;
        }
        return root;
}
//Function to print the inorder traversal of the BST
void inorder(struct node *root){
    if(root!= NULL){
      inorder(root->lst);
      printf("%d ",root->data);
      inorder(root->rst);
    }
}
//Function to find the nth node in the inOrder
void nthinorder(struct node *root,int pos){
          static int count=0;
    if(root== NULL)
                return;
                if(count<=pos){
                        nthinorder(root->lst,pos);
                        count++;
                        if(count==pos)
                                printf("%d",root->data);
        nthinorder(root->rst,pos);

    }
}
//Main Function
int main(){
        int val,n,i=0,pos;
        struct node *root=0,*mini,*maxi;
        printf("enter no. of nodes");
        scanf("%d",&n);
        while(i<n){
          root=insert(root);
          i++;
        }
                printf("\nInorder traversal of binary tree is \n");
                inorder(root);
          printf("\nenter position:");
          scanf("%d",&pos);
                printf("\n%d th node in Inorder traversal of binary tree is \n",pos);
                nthinorder(root,pos);
                return 0;
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A7\N190750_A7_P3.exe
enter no. of nodes5
enter value of node :1
enter value of node :6
enter value of node :2
enter value of node :8
enter value of node :3

Inorder traversal of binary tree is
1 2 3 6 8
enter position:3

3 th node in Inorder traversal of binary tree is
3
--------------------------------
Process exited after 18.68 seconds with return value 0
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                      Id  No. : N190750
      Assignment Number : 7(BST)
  programme Description : 4
  programme Description : C Programme to find the largest and smallest elements of a BST
                         Date : 8/23/2022
                 certification : I hereby certify that this work is my own and none of it is the work of any
other person          */
//HeaderFiles section
#include<stdio.h>
#include<stdlib.h>
//Structure Definition
struct node{
        int data;
        struct node *lst;
        struct node *rst;
};
//Function to insert the data into the BST
struct node * insert(struct node *root){
   int val;
   printf("enter value of node :");
   scanf("%d",&val);
        struct node *newnode,*node,*parent;
        newnode=(struct node*)malloc(sizeof(struct node));
        newnode->data=val;
        newnode->lst=0;
        newnode->rst=0;
        if(root==0){
                root=newnode;
                root->lst=0;
                root->rst=0;
        }
        else{
                parent=0;
                node=root;
                while(node!=0)
                {
                        parent=node;
                        if(val<node->data)
                            node=node->lst;
                        else
                            node=node->rst;
                }
                if(val<parent->data)
                  parent->lst=newnode;
                else
                  parent->rst=newnode;
                }
        return root;
}
//Function to find the maximum element in the BST
struct node* max(struct node *node){
        if(node==0 || node->rst==0)
          return node;
        else
          return max(node->rst);
}
```

```
//Function to find the minimum element in the BST
struct node* min(struct node *node){
        if(node==0 || node->lst==0)
          return node;
        else
          return min(node->lst);
}
//Main Function
int main(){
 int val,n,i=0,key,x;
 struct node *root=0,*mini,*maxi;
          printf("enter no. of nodes");
          scanf("%d",&n);
 while(i<n){
    root=insert(root);
    i++;
        }
        mini=min(root);
        printf("smallest element is %d\n",mini->data);
        maxi=max(root);
        printf("largest element is %d",maxi->data);
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A7\N190750_A7_P4.exe
enter no. of nodes4
enter value of node :1
enter value of node :5
enter value of node :2
enter value of node :7
smallest element is 1
largest element is 7
--------------------------------
Process exited after 11.64 seconds with return value 20
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                        Id  No. : N190750
      Assignment Number : 7(BST)
 programme Description : 5
 programme Description : C Programme to find all the elements of Nth level
                        Date : 8/23/2022
                certification : I hereby certify that this work is my own and none of it is the work of any
other person      */
//HeaderFiles section
#include <stdio.h>
#include <stdlib.h>
//Structure Definition
struct btnode{
   int value;
   struct btnode *l;
   struct btnode *r;
}*root = NULL, *temp = NULL, *t2, *t1;
//Function Declaration section
void insert();
void inorder(struct btnode *t);
```

```
void create();
void level(struct btnode *t,int plevel,int l);
void search(struct btnode *t);
//Main Function
void main(){
   int i,x,l;
            printf("Enter the level number:");
            scanf("%d",&l);
   printf("Enter the number of nodes:");
   scanf("%d",&x);
   for(i=0;i<x;i++)
   {
            insert();
   }
   level(root,0,l);
}
// Function to insert the data into the BST
void insert(){
   create();
   if (root == NULL)
      root = temp;
   else
      search(root);
}
/* To create a node */
void create(){
   int data;
   printf("Enter data of node to be inserted : ");
   scanf("%d", &data);
   temp = (struct btnode *)malloc(1*sizeof(struct btnode));
   temp->value = data;
   temp->l = temp->r = NULL;
}
//Approative position to insert the node
void search(struct btnode *t){
   if ((temp->value > t->value) && (t->r != NULL))
      search(t->r);
   else if ((temp->value > t->value) && (t->r == NULL))
      t->r = temp;
   else if ((temp->value < t->value) && (t->l != NULL))
      search(t->l);
   else if ((temp->value < t->value) && (t->l == NULL))
      t->l = temp;
}
void level(struct btnode *t,int plevel,int l){
            if(t==NULL){
                        return;
            }
            if(plevel==l){
                        printf("%d ",t->value);
                        return;
            }
            level(t->l,plevel+1, l);
            level(t->r,plevel+1,l);
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A7\N190750_A7_P5.exe
Enter the level number:4
Enter the number of nodes:5
Enter data of node to be inserted : 1
Enter data of node to be inserted : 3
Enter data of node to be inserted : 6
Enter data of node to be inserted : 9
Enter data of node to be inserted : 2

--------------------------------
Process exited after 17.7 seconds with return value 0
Press any key to continue . . .
```

```
/*              Author Name : N.Mariya Babu
                      Id  No. : N190750
      Assignment Number : 7(BST)
  programme Description : 6
  programme Description : C Programme to find nodes which are At max distance from the root node in the BST
                      Date : 8/23/2022
                certification : I hereby certify that this work is my own and none of it is the work of any other person      */
//HeaderFiles section
#include <stdio.h>
#include <stdlib.h>
//Structure definition
struct btnode{
   int value;
   struct btnode *l;
   struct btnode *r;
}*root = NULL, *temp = NULL, *t2, *t1;
//Function definition section
void insert();
void create();
void search(struct btnode *t);
void max(struct btnode *t);
int c=0,m=0,v[50],j=0,a[50],ma;
//main Function
void main(){
   int i,x;;
   printf("Enter the number of nodes:");
   scanf("%d",&x);
   for(i=0;i<x;i++){
          insert();
   }
   m=0;c=0;
   max(root);
   for(i=0;i<j;i++){
          ma=a[0];
          if(ma<a[i])
       ma=a[i];
          }
          printf("Maximum distance node:");
          for(i=0;i<j;i++){
                    if(ma==a[i])
                      printf("%d ,%d ",v[i],ma);
          }
```

```
}
//Insertion
void insert(){
   create();
   if (root == NULL)
      root = temp;
   else
      search(root);
}
//Node Creation
void create(){
   int data;
   printf("Enter data of node to be inserted : ");
   scanf("%d", &data);
   temp = (struct btnode *)malloc(1*sizeof(struct btnode));
   temp->value = data;
   temp->l = temp->r = NULL;
}
//The exact position to  insert the  node
void search(struct btnode *t){
   if ((temp->value > t->value) && (t->r != NULL))
      search(t->r);
   else if ((temp->value > t->value) && (t->r == NULL))
      t->r = temp;
   else if ((temp->value < t->value) && (t->l != NULL))
      search(t->l);
   else if ((temp->value < t->value) && (t->l == NULL))
      t->l = temp;
}
//Max Value in the BST
void max(struct btnode *t){
         if(t->l!=NULL){
                  c++;
                  max(t->l);
         }
         if(m<c)
           m=c;
         if(m==c){
                  a[j]=m;
                  v[j]=t->value;
                  j++;
         }
         if(t->r!=NULL){
                  c++;
                  max(t->r);
         }
         c--;
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A7\N190750_A7_P6.exe
Enter the number of nodes:5
Enter data of node to be inserted : 1
Enter data of node to be inserted : 4
Enter data of node to be inserted : 7
Enter data of node to be inserted : 3
Enter data of node to be inserted : 9
Maximum distance node:9 ,3
--------------------------------
Process exited after 29.74 seconds with return value 4
Press any key to continue . . .
```

LAB-8

```
/*              Author Name : N.Mariya Babu
                      Id  No. : N190750
      Assignment Number : 8
      programme Number : 1
 programme Description : C Programme to implement Heap Sort Algorithm using Max Heap method
                      Date : 6/21/2022 (MM/DD/YYYY)
                certification : I hereby certify that this work is my own and none of it is the work of any other person      */
//Header Files
#include <stdio.h>
int size = 0,i;
//Swap Function
void swap(int *a, int *b){
  int temp = *b;
  *b = *a;
  *a = temp;
}
//Heapify Function
void heapify(int array[], int size, int i){
  if (size == 1){
    printf("Single element in the heap");
  }
  else{
    int largest = i;
    int l = 2 * i + 1;
    int r = 2 * i + 2;
    if (l < size && array[l] > array[largest])
      largest = l;
    if (r < size && array[r] > array[largest])
      largest = r;
    if (largest != i){
      swap(&array[i], &array[largest]);
      heapify(array, size, largest);
    }
  }
}
//Insertion Function
void insert(int array[], int newNum){
  if (size == 0){
    array[0] = newNum;
    size += 1;
  }
  else{
    array[size] = newNum;
    size += 1;
    for ( i = size / 2 - 1; i >= 0; i--){
      heapify(array, size, i);
    }
  }
}
//Function to delete the RootData
void deleteRoot(int array[], int num){
  int i;
  for (i = 0; i < size; i++){
```

```
    if (num == array[i])
      break;
  }

  swap(&array[i], &array[size - 1]);
  size -= 1;
  for (i = size / 2 - 1; i >= 0; i--){
    heapify(array, size, i);
  }
}
//Function to print the array element's
void printArray(int array[], int size){
  for ( i = 0; i < size; ++i)
    printf("%d ", array[i]);
  printf("\n");
}
//Main Function
int main(){
  int array[10];
  insert(array, 3);
  insert(array, 4);
  insert(array, 9);
  insert(array, 5);
  insert(array, 2);
  printf("Max-Heap array: ");
  printArray(array, size);
  deleteRoot(array, 4);
  printf("After deleting an element: ");
  printArray(array, size);
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A8\N190750_A8_P2.exe
Print Heap
-> 1 -> 2 -> 4 -> 3 -> 7 -> 8 -> 5 -> 9 -> 6

--------------------------------
Process exited after 0.09138 seconds with return value 0
Press any key to continue . . .
```

/* **Author Name : N.Mariya Babu**
Id No. : N190750
Assignment Number : 8
programme Number : 2
programme Description : C Programme to implement Heap Sort Algorithm using Min Heap method
Date : 6/21/2022 (MM/DD/YYYY)
certification : I hereby certify that this work is my own and none of it is the work of any other person */

```
//Header Files
#include<stdio.h>
#include<stdlib.h>
int HEAP_SIZE = 20;
//Structure Definition
struct Heap{
   int *arr;
   int count;
   int capacity;
```

```
   int heap_type; // for min heap , 1 for max heap
};
typedef struct Heap Heap;
//Function Declaration
Heap *CreateHeap(int capacity,int heap_type);
void insert(Heap *h, int key);
void print(Heap *h);
void heapify_bottom_top(Heap *h,int index);
void heapify_top_bottom(Heap *h, int parent_node);
//Main Function
int main(){
   int i;
   Heap *heap = CreateHeap(HEAP_SIZE, 0); //Min Heap
   if( heap == NULL ){
      printf("__Memory Issue____\n");
      return -1;
   }
   for(i =9;i>0;i--)
      insert(heap, i);
   print(heap);
   return 0;
}
//Function to create the heap
Heap *CreateHeap(int capacity,int heap_type){
   Heap *h = (Heap * ) malloc(sizeof(Heap));
  //check if memory allocation is fails
   if(h == NULL){
      printf("Memory Error!");
      return 0;
   }
   h->heap_type = heap_type;
   h->count=0;
   h->capacity = capacity;
   h->arr = (int *) malloc(capacity*sizeof(int)); //size in bytes
   //check if allocation succeed
   if ( h->arr == NULL){
      printf("Memory Error!");
      return 0;
   }
   return h;
}
//Insertion Function
void insert(Heap *h, int key){
   if( h->count < h->capacity){
      h->arr[h->count] = key;
      heapify_bottom_top(h, h->count);
      h->count++;
   }
}
void heapify_bottom_top(Heap *h,int index){
   int temp;
   int parent_node = (index-1)/2;

   if(h->arr[parent_node] > h->arr[index]){
      //swap and recursive call
      temp = h->arr[parent_node];
      h->arr[parent_node] = h->arr[index];
```

```
        h->arr[index] = temp;
        heapify_bottom_top(h,parent_node);
    }
}
void heapify_top_bottom(Heap *h, int parent_node){
    int left = parent_node*2+1;
    int right = parent_node*2+2;
    int min;
    int temp;
    if(left >= h->count || left <0)
        left = -1;
    if(right >= h->count || right <0)
        right = -1;
    if(left != -1 && h->arr[left] < h->arr[parent_node])
        min=left;
    else
        min =parent_node;
    if(right != -1 && h->arr[right] < h->arr[min])
        min = right;
    if(min != parent_node){
        temp = h->arr[min];
        h->arr[min] = h->arr[parent_node];
        h->arr[parent_node] = temp;
        heapify_top_bottom(h, min);
    }
}
//print Function
void print(Heap *h){
    int i;
    printf("Print Heap\n");
    for(i=0;i< h->count;i++){
        printf("-> %d ",h->arr[i]);
    }
    printf("\n");
}
```

Output:

```
C:\Users\RISHI\Desktop\ZIP\N190750_A8\N190750_A8_P1.exe
Max-Heap array: 9 5 4 3 2
After deleting an element: 9 5 2 3

--------------------------------
Process exited after 0.08252 seconds with return value 0
Press any key to continue . . .
```


Thank You