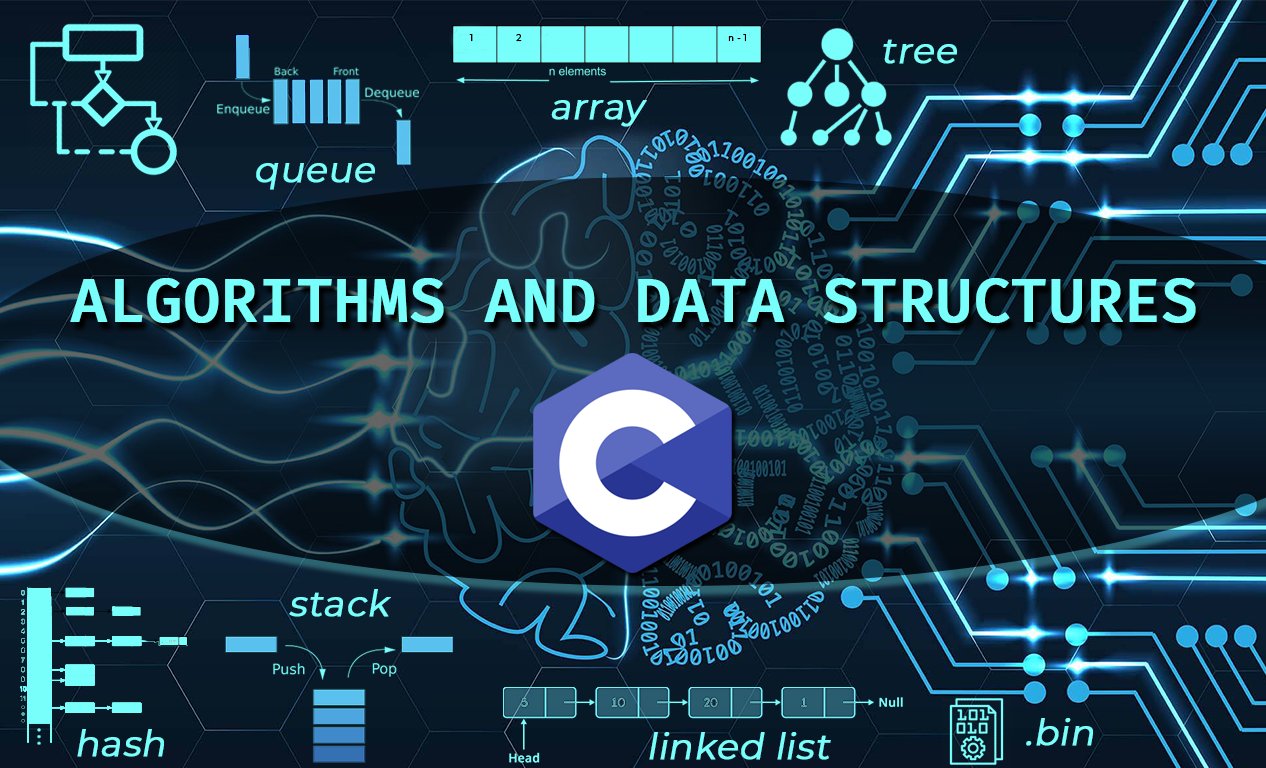


Practical File



**Sachin Rajbhar**

**COURSE: BCA**

**ROLL NO:41221139**

**DATA & FILE STRUCTURE USING C**

**FACULTY**

**Mrs. DIVYA RANA**

Index

1. Write a program for Traversal of array.
2. Write a program for Insertion in array and deletion of array.
3. Write a program for Addition and multiplication of 2d matrix
4. Write a program for Linear search
5. Write a program for Binary search
6. Write a program for Bubble sort
7. Write a program for Selection sort
8. Write a program for Insertion sort
9. Write a program for Implementation of linked list.
10. Write a program for Specified position insertion in linked list
11. Write a program for Deletion at specific position in linked list
12. Write a program for Push and pop in stack
13. Write a program for Insertion in queue.

**PRACTICAL-1**

**AIM:** Write a program for Transversal of array.

**PROGRAM:**

#include<stdio.h>

int main(){

    int arr[50];

    int i,size;

    printf("Enter the size of array:");

    scanf("%d",&size);

    printf("Enter the elements of array:\n");

    for(i=0;i<size;i++){

        scanf("%d",&arr[i]);

    }

    printf("Entered elements are:\n");

    for(i=0;i<size;i++){

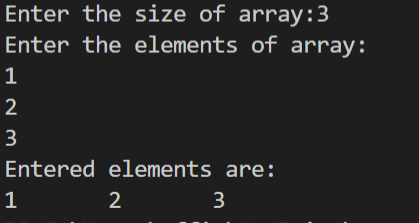
        printf("%d\t",arr[i]);

    }

return 0;

}

**OUTPUT:**



**PRACTICAL-2**

**AIM:** Write a program for Insertion in array and Deletion of array.

**PROGRAM:**

#include<stdio.h>

int main(){

    int a[50];

    int i,size,num,pos;

    printf("Enter the size of array:");

    scanf("%d",&size);

    printf("Enter the elements of array:\n");

    for(i=0;i<size;i++){

        scanf("%d",&a[i]);

    }

    printf("Enter the position where you want to insert the element:");

    scanf("%d",&pos);

     printf("Enter the element at that position:");

    scanf("%d",&num);

    if(pos<=0||pos>=size+1){

        printf("Invalid position");}

        else{

            for(i=size-1;i>=pos-1;i--){

                a[i+1]=a[i];

            }

            a[pos-1]=num;

            size++;

        }

    printf("Array after inserting the new element:\n");

    for(i=0;i<size;i++){

        printf("%d\t",a[i]);

}

            for(i=0;i<size-1;i++){

                a[i]=a[i+1];

            }

            size--;

    printf("\nArray after deleting the element:\n");

    for(i=0;i<size;i++){

        printf("%d\t",a[i]);

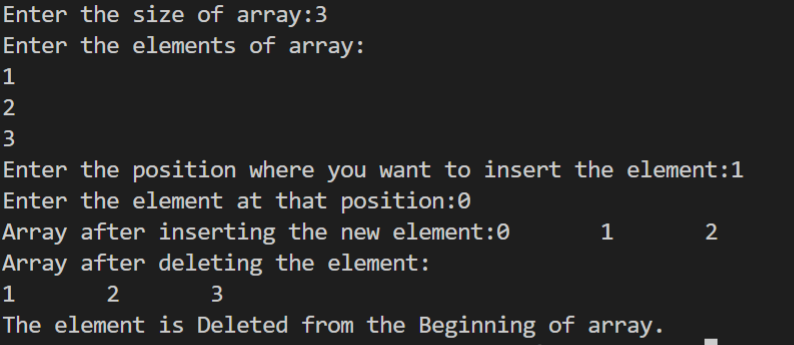
    }

    printf("\nThe element is Deleted from the Beginning of array.");

return 0;

}

**OUTPUT:**



**PRACTICAL-3**

**AIM:** Write a program for Multiplication & Addition of 2 matrices.

**PROGRAM:**

#include <stdio.h>

int main()

{

    int i,j,k,m,n,p,q;

int a[10][10],b[10][10],res[10][10],sum[10][10];

printf("Enter rows and columns of first matrix\n");

scanf("%d%d",&m,&n);

printf("Enter elements of first matrix\n");

for(i=0;i<m;i++){

    for(j=0;j<n;j++){

        scanf("%d",&a[i][j]);

    }

}

printf("Enter rows and columns of second matrix\n");

scanf("%d%d",&p,&q);

printf("Enter elements of second matrix\n");

for(i=0;i<p;i++){

    for(j=0;j<q;j++){

        scanf("%d",&b[i][j]);

    }

}

if(m==p){

for(i=0;i<m;i++){

    for(j=0;j<q;j++){

            int sum=0;

    for(k=0;k<m;k++){

           sum = sum + a[i][k]\*b[k][j];

    }

    res[i][j]=sum;

    }

}

printf("Multiplication of matrices :\n");

    for(i=0;i<m;i++){

    for(j=0;j<q;j++){

           printf("%d ",res[i][j]);

}

    printf("\n");

}

}

else{

    printf("\nInvalid Matrix\n");

    printf("Colum of 1st matrix is not equal to the row of 2nd martrix");

}

 for(i=0;i<m;i++){

    for(j=0;j<n;j++){

        sum[i][j]=a[i][j] + b[i][j];

    }

 }

 printf("\nSum of two matrices:\n");

 for(i=0;i<m;i++){

    for(j=0;j<n;j++){

        printf("%d ",sum[i][j]);

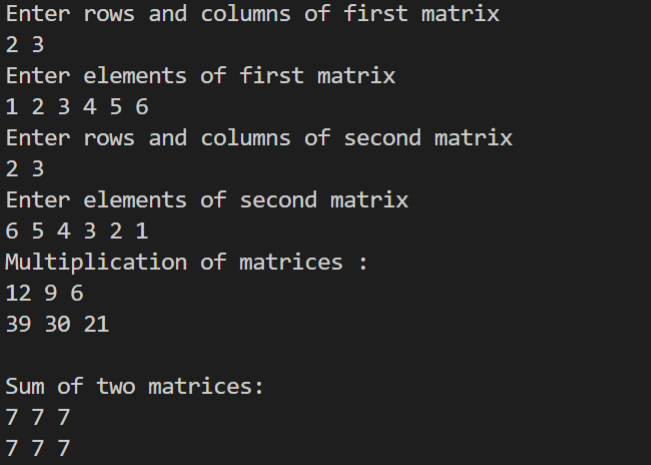
    }

    printf("\n");

 }

}

**OUTPUT:**



**PRACTICAL-4**

**AIM:** Write a program for Linear Search.

**PROGRAM:**

#include<stdio.h>

int main(){

        int i;

        int data;

        int size;

        int arr[100];

    printf("Enter the size of array:");

    scanf("%d",&size);

        printf("Enter the elements of array:");

    for(i=0;i<size;i++){

    scanf("%d",&arr[i]);}

    printf("Enter data to search for:");

    scanf("%d",&data);

        for(i=0;i<size;i++)

{

   if( arr[i]==data){

    printf("Element found at index %d:",i);

    printf("\n Found element is %d:",data);

     break;

    }

}

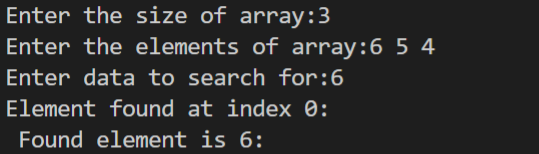
 if(i==size)

    printf("Element not found");

return 0;

}

**OUTPUT:**



**PRACTICAL-5**

**AIM:** Write a program for Binary Search.

**PROGRAM:**

#include <stdio.h>

int binarySearch(int a[], int n, int data)

{

    int l = 0, r = n - 1, mid;

    while (l <= r)

    {

        mid = (l + r) / 2;

        if (data == a[mid])

        {

            return mid;

        }

        else if (data < a[mid])

        {

            r = mid - 1;

        }

        else

        {

            l = mid + 1;

        }

    }

    return -1;

}

int main()

{

    int a[50];

    int i, n, data;

    printf("Enter the size of array: ");

    scanf("%d", &n);

    printf("Enter the elements: \n");

    for (i = 0; i < n; i++)

    {

        scanf("%d", &a[i]);

    }

    printf("Enter the data you want to search \n");

    scanf("%d", &data);

    int res = binarySearch(a, n, data);

    if (res >= 0)

        printf("%d is found at index %d",data,res);

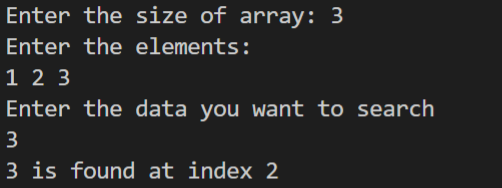
    else

        printf("Element not found!");

    return 0;

}

**OUTPUT:**



**PRACTICAL-6**

**AIM:** Write a program for Bubble Sort.

**PROGRAM:**

#include<stdio.h>

//bubble sort

int main(){

int i,j,temp,size,arr[100];

printf("Enter the size of array: ");

scanf("%d",&size);

printf("Enter the elements of array: ");

for(i=0;i<size;i++)

    scanf("%d",&arr[i]);

    printf("Array before implementing bubble sort\n");

 for(i=0;i<size;i++)

    printf("%d\t",arr[i]);

for(i=0;i<size-1;i++){

   int flag=0;

    for(j=0;j<size-1-i;j++){

    if(arr[j]>arr[j+1]){

        temp=arr[j];

        arr[j]=arr[j+1];

        arr[j+1]=temp;

        flag=1;

    }

}

if(flag==0)

break;

}

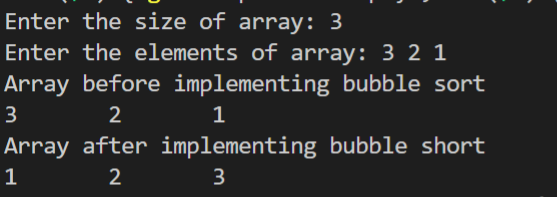
printf("\nArray after implementing bubble short\n");

for(j=0;j<size;j++)

printf("%d\t",arr[j]);

}

**OUTPUT:**



**PRACTICAL-7**

**AIM:** Write a program for Selection Sort.

**PROGRAM:**

#include<stdio.h>

    void selectionsort(int arr[], int size);

    void swap(int \*a,int \*b);

   void selectionSort(int arr[], int size)

{

    int i, j;

    for (i = 0 ;  i < size;i++)

    {

        for (j = i ; j < size; j++)

        {

            if (arr[i] > arr[j])

                swap(&arr[i], &arr[j]);

        }

    }

}

void swap(int \*a, int \*b)

{

    int temp;

    temp = \*a;

    \*a = \*b;

    \*b = temp;

}

 int main(){

    int i,j,size;

    int arr[50];

    printf("Enter the size:");

    scanf("%d",&size);

    printf("Enter the elements of array\n");

    for(i=0;i<size ;i++)

    scanf("%d",&arr[i]);

    printf("Array before selection sort\n");

         for(i=0;i<size;i++)

    printf("%d ",arr[i]);

    selectionSort(arr, size);

       printf("\nArray after selection sort \n");

        for(i=0;i<size;i++){

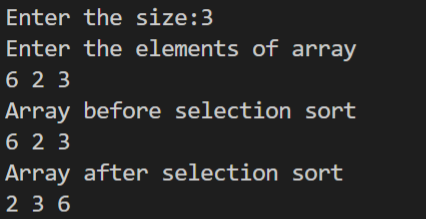
            printf("%d ",arr[i]);

        }

    return 0;

}

**OUTPUT:**



**PRACTICAL-8**

**AIM:** Write a program for Insertion sort.

**PROGRAM:**

#include <stdio.h>

int main()

{

    int n, i, j, temp;

    int arr[50];

    printf("Enter number of elements: ");

    scanf("%d", &n);

    printf("Enter the elements\n", n);

    for (i = 0; i < n; i++)

    {

        scanf("%d", &arr[i]);

    }

    for (i = 1 ; i <= n - 1; i++)

    {

        j = i;

            while ( j > 0 && arr[j-1] > arr[j])

            {

                temp     = arr[j];

                arr[j]   = arr[j-1];

                arr[j-1] = temp;

                j--;

            }

    }

    printf("Array after insertion sort:\n");

    for (i = 0; i <= n - 1; i++)

    {

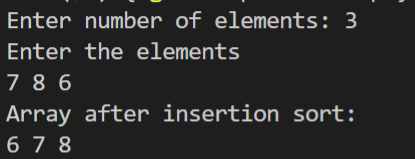
        printf("%d ", arr[i]);

    }

    return 0;

}

**OUTPUT:**



**PRACTICAL-9**

**AIM:** Write a program for Implementation of Linked List.

**PROGRAM:**

#include <stdio.h>

#include <stdlib.h>

struct node{

    int data;

    struct node \*next;

};

void display(struct node \*ptr){

    while(ptr!=NULL){

    printf("Elements of Linked List :%d\n", ptr->data);

    ptr = ptr->next;

}

}

int main(){

      // Initialize nodes

       struct node \*head;

       struct node \*first;

       struct node \*second;

       struct node \*third;

       struct node \*fourth;

       struct node \*fifth;

       // Allocate memory

     first=(struct node \*)malloc(sizeof(struct node));

     second=(struct node \*)malloc(sizeof(struct node));

     third=(struct node \*)malloc(sizeof(struct node));

     fourth=(struct node \*)malloc(sizeof(struct node));

     fifth=(struct node \*)malloc(sizeof(struct node));

       // Assign value values

      first->data=2;

      second ->data=11;

      third->data=22;

      fourth->data=90;

      fifth->data=34;

      //link first and second node

      first->next=second;

      //link second and third node

      second->next=third;

      third->next=fourth;

      fourth->next=fifth;

      fifth->next=NULL;

      //printing values

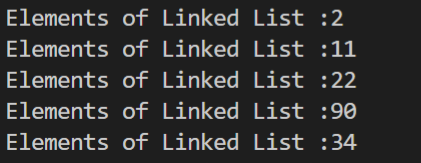
      head = first;

       display(first);

 return 0;

}

**OUTPUT:**



**PRACTICAL-10**

**AIM:** Write a program for Specific Position Insertion in linked list.

**PROGRAM:**

#include <stdio.h>

#include <stdlib.h>

struct node{

    int data;

    struct node \*next;

};

void display\_ll(struct node \*ptr){

    while(ptr!=NULL){

    printf("Elements :%d\n", ptr->data);

    ptr = ptr->next;

}

}

struct node \* insert\_at\_specific(struct node \*head, int data,int index){

     struct node \*ptr=(struct node \*)malloc(sizeof(struct node));

     struct node \*p = head;

       int i=0;

       while(i!=index-1){

        p=p->next;

        i++;

       }

       ptr->data=data;

       ptr->next=p->next;

       p->next=ptr;

return head;

}

int main(){

      // Initialize nodes

        struct node \*head,\*first,\*second,\*third;

       // Allocate memory

     first=(struct node \*)malloc(sizeof(struct node));

     second=(struct node \*)malloc(sizeof(struct node));

     third=(struct node \*)malloc(sizeof(struct node));

      first->data=2;

      second ->data=11;

      third->data=22;

      first->next=second;

      second->next=third;

      third->next=NULL;

      //printing values

             head=first;

       printf("Linked List before insertion \n");

       display\_ll(head);

       printf("Linked List after insertion\n");

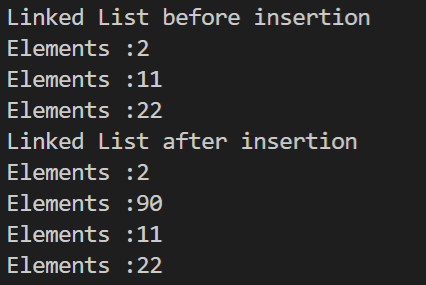
       head = insert\_at\_specific(head,90,1);

        display\_ll(head);

 return 0;

}

**OUTPUT:**



**PRACTICAL-11**

**AIM:** Write a program for Deletion at Specific Position in linked list.

**PROGRAM:**

#include <stdio.h>

#include <stdlib.h>

struct node{

    int data;

    struct node \*next;

};

void specific(struct node \*ptr){

    while(ptr!=NULL){

    printf("Elements :%d\n", ptr->data);

    ptr = ptr->next;

}

}

struct node \* delete(struct node \*head,int index){

struct node \*p=head;

struct node \*q=head->next;

for(int i=0;i<index-1;i++){

    p=p->next;

    q=q->next;

}

p->next=q->next;

free(q);

return head;

}

int main(){

      // Initialize nodes

        struct node \*head;

       struct node \*first;

       struct node \*second;

       struct node \*third;

       // Allocate memory

     first=(struct node \*)malloc(sizeof(struct node));

     second=(struct node \*)malloc(sizeof(struct node));

     third=(struct node \*)malloc(sizeof(struct node));

       // Assign value values

      first->data=4;

      second ->data=3;

      third->data=8;

      first->next=second;

      second->next=third;

      third->next=NULL;

      //printing values

             head=first;

       printf("Linked List before deletion\n");

      specific(head);

        head = delete(head,1);

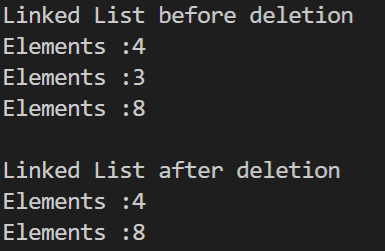
       printf("\nLinked List after deletion\n");

        specific(head);

 return 0;

}

**OUTPUT:**



**PRACTICAL-12**

**Aim:** Write a program for Push and Pop in Stack.

**PROGRAM:**

#include <stdio.h>

#define N 100

int stack[N];

int i, top = -1;

int IsFull(){

    if(top == N- 1){

        printf("\n Stack is Full");

        return 1;

    }

    return 0;

}

int IsEmpty(){

    if(top == -1){

        printf("\nStack is Empty");

        return 1;

    }

    return 0;

}

void push(int x){

    if(top == N- 1){

        printf("\n Stack is Full");}

        else{

        top++;

        stack[top] = x;

}

}

void pop(){

    int value;

    if(top == -1){

        printf("\nStack is Empty");}

        else{

        value = stack[top];

        top--;

        printf("\nDeleted element from stack is :%d ", value);

}

}

void peek(){

       if(top == -1){

        printf("\nStack is Empty");}

        else

        printf("\nThe top most element is :%d ", stack[top]);

}

void display(){

    int i;

     if(top == -1){

        printf("\nStack is Empty");}

        else{

    printf("\nElements are: ");

    for (i = top; i >= 0; i--)

    {

        printf("%d ", stack[i]);

    }

}

}

int main()

{

        push(12);

        push(45);

        push(67);

        display();

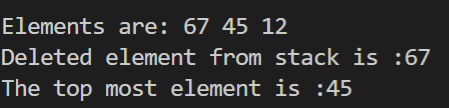
        pop();

        peek();

    return 0;

}

**OUTPUT:**



**PRACTICAL-13**

**AIM:** Write a program for Insertion & Deletion in Queue.

**PROGRAM:**

#include <stdio.h>

#define N 100

int queue[N];

int front= -1;

int rear=-1;

int IsFull(){

    if(rear==N-1){

    printf("\nQueue is Full");

    return 1;

    }

    return 0;

}

int IsEmpty(){

    if(front==-1 && rear==-1){

    printf("\nQueue is Empty");

    return 1;

    }

    return 0;

}

void Enqueue(int val){

    IsFull();

    if(front==-1 && rear==-1){

    front=rear=0;

    queue[rear]=val;

    }

    else{

        rear++;

        queue[rear]=val;

    }

}

void Dequeue(){

    IsEmpty();

    if(front==rear){

    front=rear=-1;

    }

    else{

        printf("\nThe deleted element is: %d",queue[front]);

        front++;

    }

}

void display(){

       IsEmpty();

    printf("Elements of Queue are: ");

    for(int i=front;i<rear+1;i++){

        printf("%d ",queue[i]);

    }

}

void peek(){

    IsEmpty();

    printf("\nFirst elements of queue is: %d ",queue[front]);

}

void size(){

    printf("\nThe size of queue is: %d",sizeof(queue[rear]));

}

 int main(){

        Enqueue(12); Enqueue(34);Enqueue(23);

        display();size();

        Dequeue();Dequeue();

        IsFull();IsEmpty();peek();

}

**OUTPUT:**

