**Name: Anadi Sharma**

**SAP ID: 500126798**

**Enrollment Number: R2142232086**

**Batch: 46**

**Q1.) Static and Dynamic Array Declaration.**

**Source Code:-**

**#include <stdio.h>**

**void declaration();**

**void sort(int arr[]);**

**int main()**

**{**

**printf("Do you want To Declare An Array(y/n): ");**

**char c;**

**scanf("%c",&c);**

**if(c=='y'||c=='Y')**

**{**

**declaration();**

**}**

**else**

**{**

**printf("<<<EXITING PROGRAM>>>");**

**}**

**}**

**void declaration()**

**{**

**printf("Enter 1 for Dynamic Declaration of Array\n");**

**printf("Enter 2 for Static Declaration of Array\n");**

**int ch;**

**scanf("%d",&ch);**

**switch(ch)**

**{**

**case 1:**

**printf("You Chose Dynamic..\n");**

**int MAX,n;**

**printf("Enter Array Max Size: ");**

**scanf("%d",&MAX);**

**printf("Enter No. Of Elements you want to Enter: ");**

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

**if(MAX<n)**

**{**

**printf("MAX cannot be smaller than UB");**

**}**

**else**

**{**

**printf("Enter %d Elements:\n",n);**

**int i;**

**int \*ar = (int \*)malloc(MAX\*sizeof(int));**

**if(ar == NULL)**

**{**

**printf("NO Space\n..");**

**}**

**else**

**{**

**for(i=0;i<n;i++)**

**{**

**scanf("%d",&ar[i]);**

**}**

**printf("Array Elements Are:\n");**

**for(i=0;i<n;i++)**

**{**

**printf("%d\n",ar[i]);**

**}**

**}**

**}**

**break;**

**case 2:**

**printf("You Chose Static..\n");**

**int arr[50];**

**printf("Enter Number of Elements you want to enter out of 50 \n”);**

**int lim;**

**scanf("%d",&lim);**

**printf("Enter %d Elements:\n",lim);**

**int i;**

**for(i=0;i<lim;i++)**

**{**

**scanf("%d",&arr[i]);**

**}**

**printf("Array Elements Are:\n");**

**for(i=0;i<lim;i++)**

**{**

**printf("%d\n",arr[i]);**

**}**

**break;**

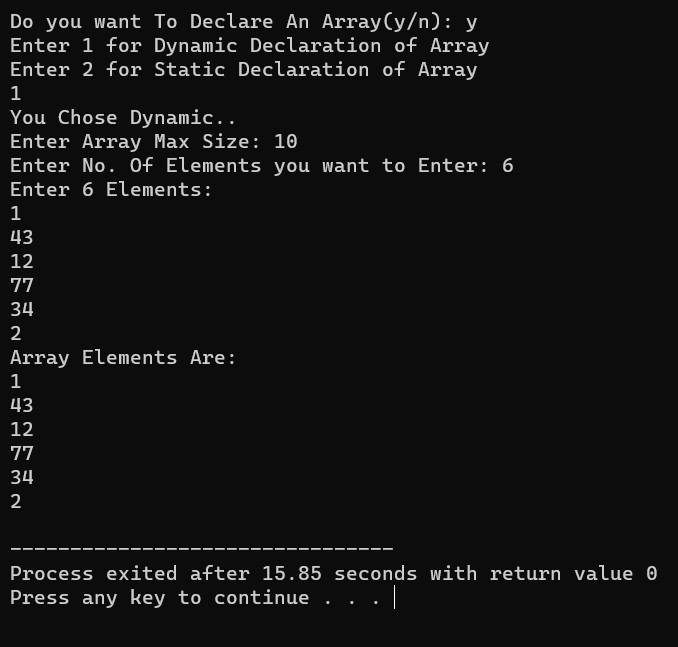
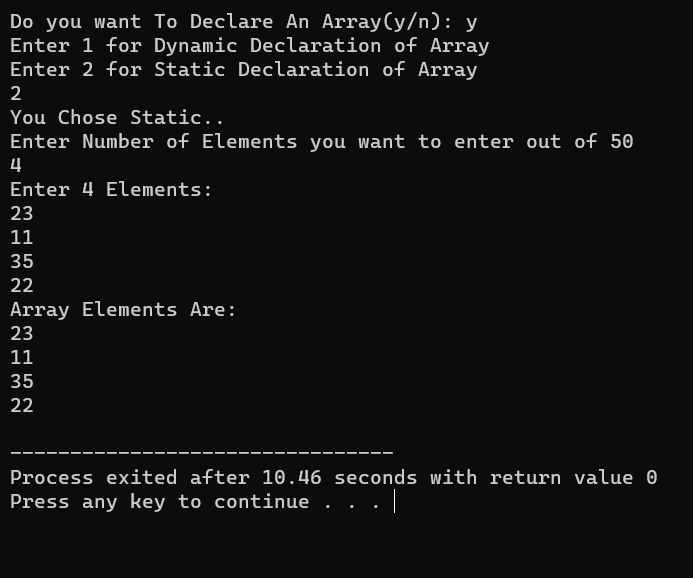
**default:**

**printf("Invalid Choice...");**

**}**

**}**

**Output:-**

**Q2.) Traversing in An Array.**

**Source Code:-**

**#include <stdio.h>**

**int i,n;**

**void display(int arr[]);** //Declaration of the display() funtion

**int main()**

**{**

**printf("Enter Number Of Elements In Array:\n");** //inputs size of the array

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

**int ar[n];**

**printf("Enter %d Elements:\n",n);**

**for(i=0;i<n;i++)**

**{**

**scanf("%d",&ar[i]); //User inputs the elemnts in the array**

**}**

**display(ar);**

**}**

**void display(int arr[])**

**{**

**printf("The Array Elements Are: ");**

**for(i=0;i<n;i++)**

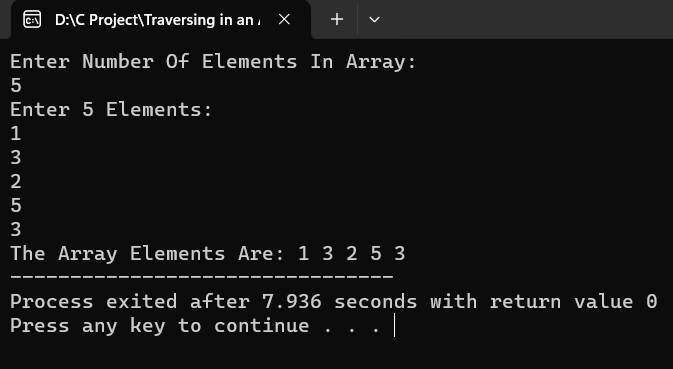
**{**

**printf("%d ",arr[i]); //Prints the value stored in all the indexes of the array**

**}**

**}**

**Output:-**



**Q3.) Linear Search in An Array.**

**Source Code:-**

**#include <stdio.h>**

**int MAX,n,i,j;**

**void fillar(int ar[]);**

**void Lsearch(int ar[]);**

**int main()**

**{**

**printf("Enter Max Size Of Array: ");**

**scanf("%d",&MAX);**

**printf("Enter number of elements in the Array: ");**

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

**int ar[n];**

**fillar(ar);**

**Lsearch(ar);**

**}**

**void fillar(int ar[])**

**{**

**if(n>MAX)**

**{**

**printf("Elements To be Filled in the Array Cannot be Greater than Size of the Array\n");**

**}**

**else**

**{**

**printf("Enter %d Elements:\n",n);**

**for(i=0;i<n;i++)**

**{**

**scanf("%d",&ar[i]);**

**}**

**}**

**}**

**void Lsearch(int ar[])**

**{**

**int num,ind=0;**

**printf("Enter The Number That You Want To Search: ");**

**scanf("%d",&num);**

**for(i=0;i<n;i++)**

**{**

**if(ar[i] == num)**

**{**

**printf("The Array Is: ");**

**for(j=0;j<n;j++)**

**{**

**printf("%d ",ar[j]);**

**}**

**printf("\n");**

**printf("Element Found: %d\n",ar[i]);**

**printf("Found At INDEX: %d",ind);**

**}**

**else**

**{**

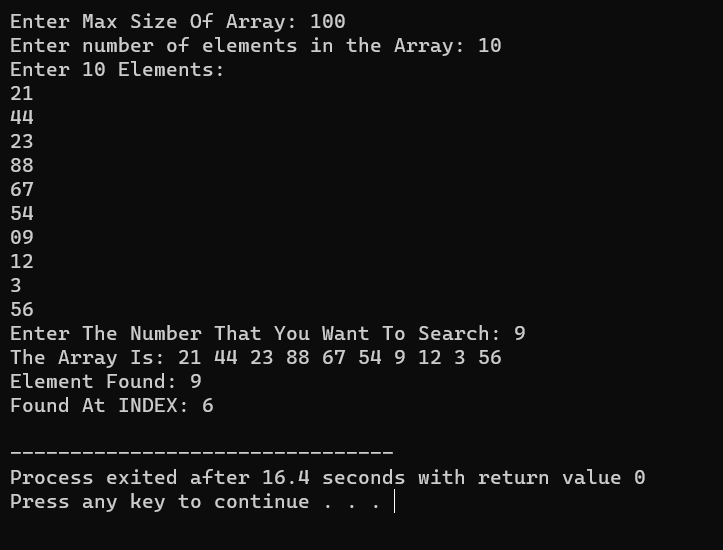
**ind++;**

**}**

**}**

**}**

**Output:-**



**Q4.) Binary Search In An Array.**

**Source Code:-**

**#include <stdio.h>**

**void sort(int arr[],int n); //Declaration of the Function**

**void Bsearch(int arr[],int n,int a,int d); // Decalration of Function which Performs Binary search in the Sorted Array**

**int i,j;**

**int main() //Main function , the program execution starts from here**

**{**

**int n,MAX;**

**printf("Enter Array Size: ");**

**scanf("%d",&MAX);** //Asking number of elements in the Array from the user

**printf("Enter Number Of Elements in Array: ");**

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

**int arr[n];**

**if(MAX>n)**

**{**

**printf("Enter Array Elements:\n");**

**for(i=0;i<n;i++)**

**{**

**scanf("%d",&arr[i]);** //Entering elements inside the array

**}**

**sort(arr,n);**

**}**

**else**

**{**

**printf("Array SIZE cannot be Smaller that Number of Elements You want to Fill(+\_+)\n");**

**}**

**}**

**void sort(int arr[],int n)** //Function for sorting of the array

**{**

**printf("Enter 1 to sort in Ascending Order\nEnter 2 to sort in Descending Order\n");**

**int ch;**

**int a,d;**

**scanf("%d",&ch);**

**switch(ch)**

**{**

**case 1:**

**a=1;**

**int j,tmp;** //tmp is a variable which stores the value of the element and sorts

**printf("The Original Array is: ");**

**for(i=0;i<n;i++)**

**{**

**printf("%d ",arr[i]);** //This Loop prints the original array

**}**

**printf("\n");**

**for(i=0;i<(n-1);i++)**

**{**

**for(j=0;j<(n-1-i);j++)**

**{**

**if(arr[j]>arr[j+1])**

**{**

**tmp=arr[j+1];**

**arr[j+1]=arr[j];** //Sorting is Done here

**arr[j]=tmp;**

**}**

**}**

**}**

**printf("The Array In Ascending Orer Is: ");**

**for(i=0;i<n;i++)**

**{**

**printf("%d ",arr[i]);** //This Loop prints the sorted array

**}**

**printf("\n");**

**break;**

**case 2:**

**d=1;**

**printf("The Original Array is: ");**

**for(i=0;i<n;i++)**

**{**

**printf("%d ",arr[i]);** //This Loop prints the original array

**}**

**printf("\n");**

**for(i=0;i<(n-1);i++)**

**{**

**for(j=0;j<(n-1-i);j++)**

**{**

**if(arr[j]<arr[j+1])**

**{**

**tmp=arr[j+1];**

**arr[j+1]=arr[j];** //Sorting is Done here

**arr[j]=tmp;**

**}**

**}**

**}**

**printf("The Array In Descending Order Is: ");**

**for(i=0;i<n;i++)**

**{**

**printf("%d ",arr[i]);** //This Loop prints the sorted array

**}**

**printf("\n");**

**break;**

**}**

**Bsearch(arr,n,a,d);**

**}**

**void Bsearch(int arr[],int n,int a,int d)**

**{**

**printf("Did you Sort In Ascending Order Or In Descending Order(a/d): ");**

**char ad;**

**scanf(" %c",&ad);**

**int bkey;**

**if(ad=='a' && a==1 ||ad=='A' && a==1)**

**{**

**printf("Enter Element to Search: ");**

**scanf("%d",&bkey);**

**int mid,left=0,right=n;**

**mid=(left+right)/2;**

**while(left<=right)**

**{**

**if(bkey==arr[mid])**

**{**

**printf("Element Found At INDEX: %d\n",mid);**

**break;**

**}**

**else if(bkey>arr[mid])**

**{**

**left=mid+1;**

**}**

**else**

**{**

**right=mid-1;**

**}**

**mid=(left+right)/2;**

**}**

**}**

**else if(ad=='d' && d==1||ad=='D' && d==1 )**

**{**

**printf("Enter Element to Search: ");**

**scanf("%d",&bkey);**

**int mid,left=0,right=n;**

**mid=(left+right)/2;**

**while(left<=right)**

**{**

**if(arr[mid]==bkey)**

**{**

**printf("Element Found At INDEX: %d",mid);**

**break;**

**}**

**else if(arr[mid]<bkey)**

**{**

**right=mid-1;**

**}**

**else**

**{**

**left=mid+1;**

**}**

**mid=(left+right)/2;**

**}**

**}**

**else**

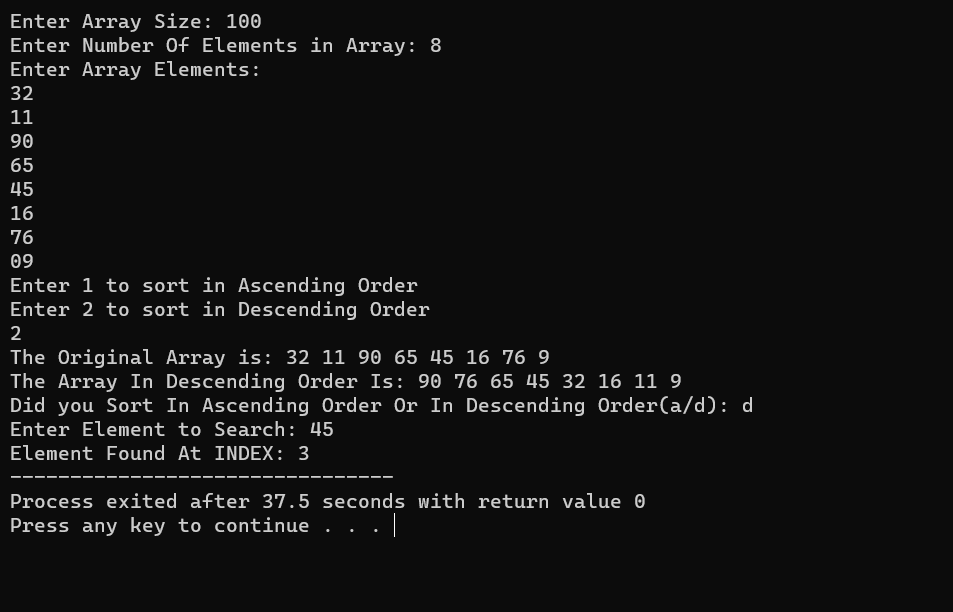
**{**

**printf("Jhooth bolo Baar Baar Jhooth Bolo (T\_T) \n");**

**}**

**}**

**Output:-**



**Q5.) Sorting in An Array.**

**Source Code:-**

**#include <stdio.h>**

**void sort(int arr[],int n);** //Declaration of the Function

**int i,j;**

**int main() //Main function , the program execution starts from here**

**{**

**int n,MAX;**

**printf("Enter Array Size: ");**

**scanf("%d",&MAX);** //Asking number of elements in the Array from the user

**printf("Enter Number Of Elements in Array: ");**

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

**if(MAX>n)**

**{**

**int arr[n];**

**printf("Enter Array Elements:\n");**

**for(i=0;i<n;i++)**

**{**

**scanf("%d",&arr[i]);** //Entering elements inside the array

**}**

**sort(arr,n);**

**}**

**else**

**{**

**printf("Array SIZE cannot be Smaller that Number of Elements You want to Fill(+\_+)\n");**

**}**

**}**

**void sort(int arr[],int n)** //Function for sorting of the array

**{**

**printf("Enter 1 to sort in Ascending Order\nEnter 2 to sort in Descending Order\n");**

**int ch;**

**scanf("%d",&ch);**

**switch(ch)**

**{**

**case 1:**

**int j,tmp;** //tmp is a variable which stores the value of the element and sorts

**printf("The Original Array is: ");**

**for(i=0;i<n;i++)**

**{**

**printf("%d ",arr[i]);** //This Loop prints the original array

**}**

**printf("\n");**

**for(i=0;i<(n-1);i++)**

**{**

**for(j=0;j<(n-1-i);j++)**

**{**

**if(arr[j]>arr[j+1])**

**{**

**tmp=arr[j+1];**

**arr[j+1]=arr[j];** //Sorting is Done here

**arr[j]=tmp;**

**}**

**}**

**}**

**printf("The Array In Ascending Orer Is: ");**

**for(i=0;i<n;i++)**

**{**

**printf("%d ",arr[i]);** //This Loop prints the sorted array

**}**

**break;**

**case 2:**

**printf("The Original Array is: ");**

**for(i=0;i<n;i++)**

**{**

**printf("%d ",arr[i]);** //This Loop prints the original array

**}**

**printf("\n");**

**for(i=0;i<(n-1);i++)**

**{**

**for(j=0;j<(n-1-i);j++)**

**{**

**if(arr[j]<arr[j+1])**

**{**

**tmp=arr[j+1];**

**arr[j+1]=arr[j];** //Sorting is Done here

**arr[j]=tmp;**

**}**

**}**

**}**

**printf("The Array In Descending Order Is: ");**

**for(i=0;i<n;i++)**

**{**

**printf("%d ",arr[i]);** //This Loop prints the sorted array

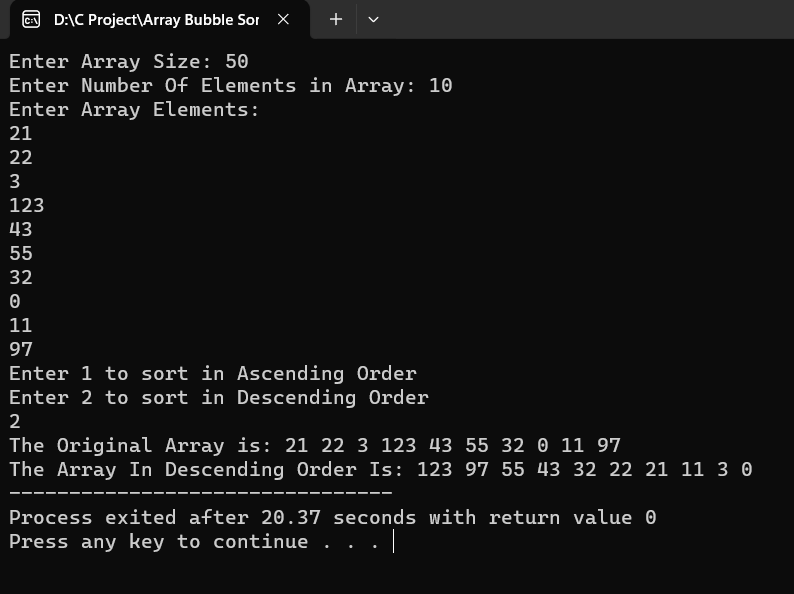
**}**

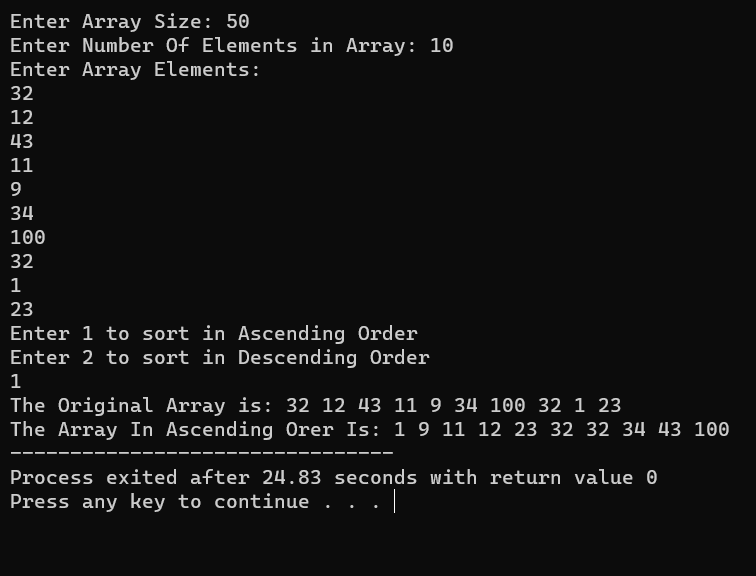
**break;**

**}**

**}**

**Output:-**





**Q6.) Menu Driven Program for Insertion And Deletion Of Elements in an Array.**

**Source Code:-**

**#include <stdio.h>**

**int i;**

**void insert(int ar[] , int \*n , int MAX);**

**void deletion(int ar[] , int MAX ,int \*\*n);**

**int main()**

**{**

**int n,MAX;**

**printf("Enter MAX value of Array: ");**

**scanf("%d",&MAX);**

**int ar[MAX];**

**printf("Enter No. of Elements In The Array: ");**

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

**if(MAX<n)**

**{**

**printf("No. Of Elements to be inserted cannot be greater than the MAX Size..");**

**}**

**else**

**{**

**printf("Enter Array Elements:\n");**

**for(i=0;i<n;i++)**

**{**

**scanf("%d",&ar[i]);**

**}**

**printf("Array Elements Are: ");**

**for(i=0;i<n;i++)**

**{**

**printf("%d ",ar[i]);**

**}**

**printf("\n");**

**insert(ar,&n,MAX);**

**}**

**}**

**void insert(int ar[] , int \*n, int MAX)**

**{**

**printf("Do you Want to insert an Element?(y/n): ");**

**char ins;**

**scanf(" %c",&ins);**

**if(ins=='y'||ins=='Y')**

**{**

**char c;**

**int ch;**

**printf("Make your selection...but beware the consequences\n");**

**printf("Enter 1 to Insert Element in the Beginning\n");**

**printf("Enter 2 to Insert Element in The End\n");**

**printf("Enter 3 to Insert Element At a Particular Location\n");**

**scanf("%d",&ch);**

**switch(ch)**

**{**

**case 1: //Insertion At the Beginning**

**printf("You Entered Insertion At the Beginning..\n");**

**printf("Enter Element to Enter: ");**

**int data;**

**scanf("%d",&data);**

**for(i=\*n;i>0;i--)**

**{**

**ar[i]=ar[i-1];**

**}**

**++(\*n);**

**ar[0]=data;**

**printf("Array After Insertion At Beginning: ");**

**for(i=0;i<\*n;i++)**

**{**

**printf("%d ",ar[i]);**

**}**

**printf("\n");**

**printf("Value Of n is now: %d",\*n);**

**printf("\n");**

**printf("Do you Want to Delete an Element(y/n): ");**

**scanf(" %c",&c);**

**if(c=='y'||c=='Y')**

**{**

**deletion(ar , MAX , &n);**

**}**

**else if(c=='n'||c=='N')**

**{**

**printf("Exiting....");**

**}**

**else**

**{**

**printf("INVALID CHOICE (T-T)");**

**}**

**break;**

**case 2: //Insertion At the End**

**printf("You Entered Insertion At the End..\n");**

**printf("Enter Element to Enter: ");**

**scanf("%d",&data);**

**ar[(\*n)]=data;**

**++(\*n);**

**printf("Array After Insertion At End: ");**

**for(i=0;i<\*n;i++)**

**{**

**printf("%d ",ar[i]);**

**}**

**printf("\n");**

**printf("Value Of n is now: %d",\*n);**

**printf("\n");**

**printf("Do you Want to Delete an Element(y/n): ");**

**scanf(" %c",&c);**

**if(c=='y'||c=='Y')**

**{**

**deletion(ar , MAX , &n);**

**}**

**else if(c=='n'||c=='N')**

**{**

**printf("Exiting....");**

**}**

**else**

**{**

**printf("INVALID CHOICE (T-T)");**

**}**

**break;**

**case 3: //Insertion At Particular Location**

**printf("You Entered Insertion At a Particular Location..\n");**

**int loc;**

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

**scanf("%d",&loc);**

**if(loc<0||loc>\*n)**

**{**

**printf("Invalid position.Sorry(T-T)\n");**

**}**

**else**

**{**

**printf("Enter Element to Insert: ");**

**scanf("%d",&data);**

**for(i=\*n;i>loc;i--)**

**{**

**ar[i]=ar[i - 1];**

**}**

**ar[loc]=data;**

**++(\*n);**

**printf("Array After Insertion At Particular Position Is: ");**

**for (int i = 0; i < \*n; i++)**

**{**

**printf("%d ",ar[i]);**

**}**

**printf("\n");**

**printf("Value Of n is now : %d\n",\*n);**

**printf("\n");**

**}**

**printf("Do you Want to Delete an Element(y/n): ");**

**scanf(" %c",&c);**

**if(c=='y'||c=='Y')**

**{**

**deletion(ar , MAX , &n);**

**}**

**else if(c=='n'||c=='N')**

**{**

**printf("Exiting....");**

**}**

**else**

**{**

**printf("INVALID CHOICE (T-T)");**

**}**

**break;**

**default:**

**printf("INVAlID...");**

**break;**

**}**

**}**

**else**

**{**

**deletion(ar , MAX , &n);**

**}**

**}**

**void deletion(int ar[] , int MAX ,int \*\*n)**

**{**

**int ch,i;**

**printf("Make your selection...but beware the consequences\n");**

**printf("Enter 1 to Delete Element in the Beginning\n");**

**printf("Enter 2 to Delete Element in The End\n");**

**printf("Enter 3 to Delete Element At a Particular Location\n");**

**scanf("%d",&ch);**

**switch(ch)**

**{**

**case 1:** //Deletion From the Beginning

**printf("You Entered Deletion At the Beginning..\n");**

**if(\*n==0)**

**{**

**printf("Array is empty. Nothing to delete.\n");**

**}**

**else**

**{**

**printf("Element at the Beginning Is: %d\n",ar[0]);**

**printf("Are you sure you want to delete? (y/n): ");**

**char c;**

**scanf(" %c",&c);**

**if(c=='y'||c=='Y')**

**{**

// Shift elements to the left

**for(i = 0; i < \*\*n - 1; i++)**

**{**

**ar[i]=ar[i + 1];**

**}**

**(\*\*n)--;**

**printf("Element Deleted from the beginning.\n");**

**printf("Array After Deletion From the Beginning: ");**

**for (int i=0;i<\*\*n;i++)**

**{**

**printf("%d ",ar[i]);**

**}**

**printf("\n");**

**printf("Value Of n is now : %d",\*\*n);**

**}**

**else if(c=='n'||c=='N')**

**{**

**printf("Will Not Delete. Exiting...\n");**

**}**

**else**

**{**

**printf("INVALID CHOICE (T-T)\n");**

**}**

**}**

**break;**

**case 2: //Deletion From the End**

**printf("You Entered Deletion At the End..\n");**

**if(\*\*n==0)**

**{**

**printf("Array is empty. Nothing to delete.\n");**

**}**

**else**

**{**

**printf("Element at the End Is: %d\n",ar[\*\*n - 1]);**

**printf("Are you sure you want to delete?(y/n): ");**

**char c;**

**scanf(" %c",&c);**

**if(c=='y'||c=='Y')**

**{**

**(\*\*n)--;**

**printf("Element deleted from the end.\n");**

**printf("Array After Deletion From the End: ");**

**for (int i=0;i<\*\*n;i++)**

**{**

**printf("%d ",ar[i]);**

**}**

**printf("\n");**

**printf("Value Of n is now : %d",\*\*n);**

**}**

**else if(c=='n'||c=='N')**

**{**

**printf("Will Not Delete. Exiting...\n");**

**}**

**else**

**{**

**printf("INVALID CHOICE (T-T)\n");**

**}**

**}**

**break;**

**case 3:** //Deletion From a Prticular Location

**printf("You Entered Deletion At a Particular Location..\n");**

**if(\*\*n==0)**

**{**

**printf("Array is empty. Nothing to delete.\n");**

**}**

**else**

**{**

**int loc;**

**printf("Enter the position of the element you want to delete: ");**

**scanf("%d",&loc);**

**if (loc<0||loc>=\*\*n)**

**{**

**printf("Invalid position. Element not found.\n");**

**}**

**else**

**{**

**printf("Element at position %d Is: %d\n",loc,ar[loc]);**

**printf("Are you sure you want to delete? (y/n): ");**

**char c;**

**scanf(" %c",&c);**

**if(c=='y'||c=='Y')**

**{**

// Shift elements to the left

**for(i=loc;i<\*\*n-1; i++)**

**{**

**ar[i]=ar[i + 1];**

**}**

**(\*\*n)--;**

**printf("Element deleted from position %d.\n", loc);**

**for (int i=0;i<\*\*n;i++)**

**{**

**printf("%d ",ar[i]);**

**}**

**printf("\n");**

**printf("Value Of n is now : %d",\*\*n);**

**}**

**else if(c=='n'||c=='N')**

**{**

**printf("Will Not Delete. Exiting...\n");**

**}**

**else**

**{**

**printf("INVALID CHOICE (T-T)\n");**

**}**

**}**

**}**

**break;**

**default:**

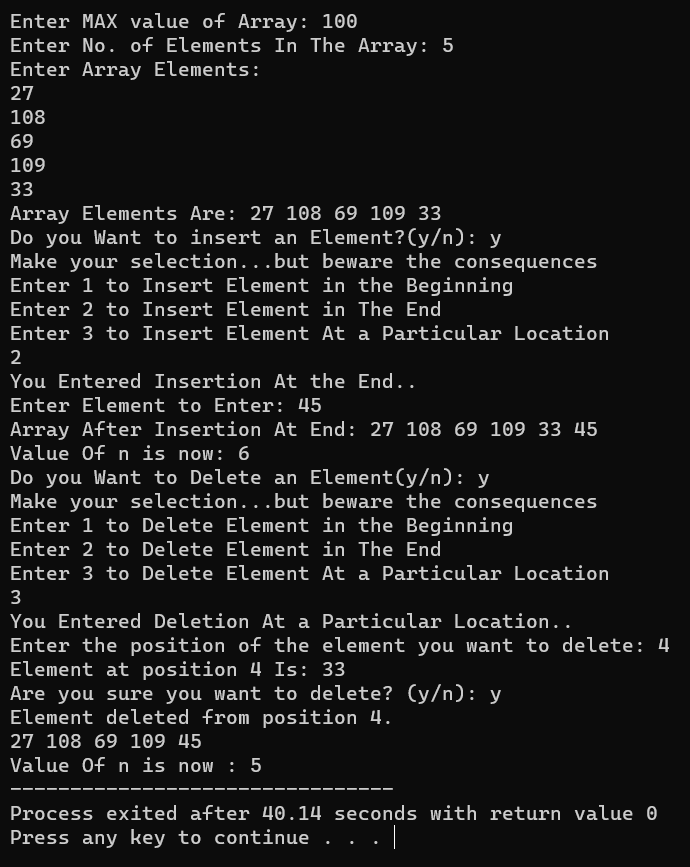
**printf("INVALID CHOICE (T-T)\n");**

**break;**

**}**

**}**

**OUTPUT:-**



**Q.7) Menu Driven Program for Insertion And Deletion Of Nodes in a Singular Linked List.**

**Source Code:-**

**#include <stdio.h>**

**#include <stdlib.h>**

**struct node**

**{**

**int data;**

**struct node \*next;**

**};**

**int main()**

**{**

**struct node \*head, \*newnode, \*tmp, \*prevnode, \*ptmp;**

**int count = 0, loc;**

**char ch;**

**head = NULL;**

**printf("Do you want to insert a node (y/n): ");**

**scanf("%c" , &ch);**

**while (ch == 'y' || ch == 'Y')**

**{**

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

**if (newnode == NULL)**

**{**

**printf("No Space Available... Bye Bye\n");**

**break;**

**}**

**printf("Enter the Value of the node: ");**

**int val;**

**scanf("%d" , &val);**

**newnode->data = val;**

**newnode->next = NULL;**

**if(head == NULL)**

**{**

**head=newnode;**

**}**

**else**

**{**

**printf("1. Insert at the beginning\n");**

**printf("2. Insert at the end\n");**

**printf("3. Insert after a specific element\n");**

**printf("4. Insert after a specific location number\n");**

**printf("Enter your choice: ");**

**int choice;**

**scanf("%d" , &choice);**

**switch (choice)**

**{**

**case 1:** // Insertion At The Beginning Of LL

**newnode->next = head;**

**head = newnode;**

**break;**

**case 2: // Insertion At the End**

**tmp = head;**

**while(tmp->next != NULL)**

**{**

**tmp = tmp->next;**

**}**

**tmp->next=newnode;**

**break;**

**case 3:** // Insertion After a Specific Element

**printf("Enter the element to insert after: ");**

**int value;**

**scanf("%d",&value);**

**tmp=head;**

**while(tmp!=NULL && tmp->data!=value)**

**{**

**tmp=tmp->next;**

**}**

**if (tmp!=NULL)**

**{**

**newnode->next=tmp->next;**

**tmp->next=newnode;**

**}**

**else**

**{**

**printf("Element not found(T\_T)\n");**

**}**

**break;**

**case 4:** // Insertion After a Specific Location Number

**printf("Enter the location number from Indexing of 0: ");**

**scanf("%d", &loc);**

**count=0;**

**tmp=head;**

**while(tmp!=NULL && count<loc)**

**{**

**count++;**

**prevnode=tmp;**

**tmp=tmp->next;**

**}**

**if(tmp!=NULL)**

**{**

**newnode->next=tmp;**

**if(loc==0)**

**{**

**head=newnode;**

**}**

**else**

**{**

**prevnode->next=newnode;**

**}**

**}**

**else**

**{**

**printf("Invalid location(Y\_Y).\n");**

**}**

**break;**

**default:**

**printf("Invalid choice.\n");**

**}**

**}**

**printf("Do you want to continue (y/n): ");**

**scanf(" %c",&ch);**

**}**

**tmp=head;**

**printf("The Singly Linked list is: ");**

**while (tmp!=NULL)**

**{**

**printf("%d ",tmp->data);**

**tmp=tmp->next;**

**}**

**printf("\n\n");**

**printf("Do you Want to Delete A Node(y/n): ");**

**char del;**

**scanf(" %c",&del);**

**while(del=='y'||del=='Y')**

**{**

**if(head==NULL)**

**{**

**printf("No Node Present In the Linked List\n");**

**}**

**else**

**{**

**printf("1. Deletion at the beginning\n");**

**printf("2. Deletion at the end\n");**

**printf("3. Deletion after a specific element\n");**

**printf("Enter your choice: ");**

**int cho;**

**scanf("%d",&cho);**

**switch (cho)**

**{**

**case 1:** // Deletion of Node At The Beginning

**tmp=head;**

**head=head->next;**

**free(tmp);**

**break;**

**case 2:** // Deletion of Node At The End

**tmp=head;**

**while (tmp->next->next!=NULL)**

**{**

**tmp=tmp->next;**

**}**

**free(tmp->next);**

**tmp->next=NULL;**

**break;**

**case 3:** // Deletion after a Specific Element

**printf("Enter the element to delete after: ");**

**int value;**

**scanf("%d",&value);**

**tmp=head;**

**while(tmp != NULL && tmp->data != value)**

**{**

**tmp=tmp->next;**

**}**

**if(tmp!=NULL && tmp->next!=NULL)**

**{**

**struct node \*tmpnxt=tmp->next;**

**tmp->next=tmpnxt->next;**

**free(tmpnxt);**

**}**

**else**

**{**

**tmp=head;**

**while(tmp->next!=NULL)**

**{**

**ptmp=tmp;**

**tmp=tmp->next;**

**}**

**free(tmp);**

**ptmp->next=NULL;**

**}**

**break;**

**default:**

**printf("Invalid choice.\n");**

**}**

**}**

**printf("Do you Want to Delete A Node(y/n): ");**

**scanf(" %c",&del);**

**}**

**tmp=head;**

**printf("The Singly Linked List After Deletion is:\n");**

**while(tmp!=NULL)**

**{**

**printf("The Value Is:%d\n",tmp->data);**

**printf("The Next Is :%p\n",tmp->data);**

**tmp=tmp->next;**

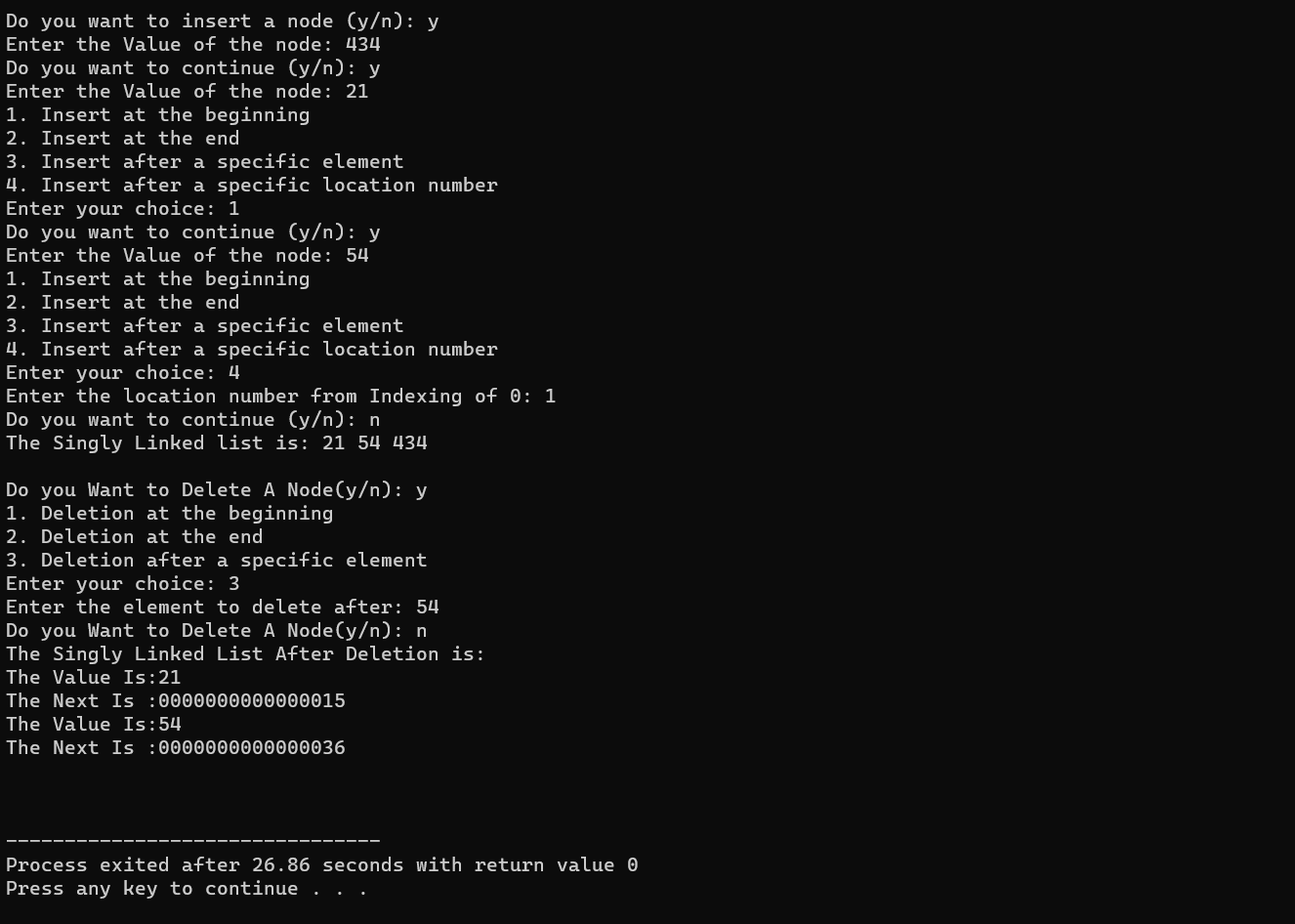
**}**

**printf("\n\n");**

**return 0;**

**}**

**OUTPUT:-**



**Q8.) Menu Driven Program for Insertion And Deletion Of Nodes in a Doubly Circular Linked List.**

**Source Code:-**

**#include <stdio.h>**

**#include <stdlib.h>**

**struct node**

**{**

**struct node \*prev;** //Structure to create a node containing prev of node, next of node and the data in the node.

**int data;**

**struct node \*next;** //pointer type next which is of type struct node

**};**

**int main()**

**{**

**struct node \*head,\*newnode,\*tmp,\*ptmp;**

//pointer type head, newnode, tmp which are of type struct node.

**int count=0, loc;**

**head=NULL;** //Initially head is empty

**printf("Do you want to insert a node (y/n): ");**

//Asks the user if he wants to enter a node or not

**char ch;**

**scanf("%c",&ch);**

**while(ch=='y'||ch=='Y')**

**{**

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

//Checks if memory is available for the insertion of node

**if(newnode==NULL)**

**{**

**printf("No Space Available...Bye Bye\n");**

**}**

**printf("Enter the Value of the node: ");**

**int val;**

**scanf("%d" , &val);**

**newnode->data = val;**

**if (head == NULL)** //If no node is present in the Doubly circular linked list

**{**

**newnode->prev = newnode;**

**newnode->next = newnode;**

**head = newnode;**

**}**

**else**

**{**

**printf("1. Insert at the beginning\n");**

**printf("2. Insert at the end\n");**

**printf("3. Insert after a specific element\n");**

**printf("4. Insert after a specific location number\n");**

**printf("Enter your choice: ");** //Asks the user for his choice

**int choice;**

**scanf("%d",&choice);**

**switch(choice)**

**{**

**case 1:** //Insertion At The Beginning

**newnode->next=head;**

**newnode->prev=head->prev;**

**head->prev->next=newnode;**

**head->prev=newnode;**

**head=newnode;**

**break;**

**case 2:** //Insertion At the End

**newnode->next=head;**

**newnode->prev=head->prev;**

**head->prev->next=newnode;**

**head->prev=newnode;**

**break;**

**case 3:** //Insertion After a Specific Element

**printf("Enter the element to insert after: ");**

**int value;**

**scanf("%d" , &value);**

**tmp = head;**

**while(tmp != NULL && tmp->data != value)**

**{**

/\*checks till tmp is not NULL or tmp data is not equal to value\*/

**tmp = tmp->next;**

**}**

**if(tmp != NULL)**

/\*checks if tmp holds a valid memory address, and it's safe to access the value tmp is pointing to.\*/

**{**

**newnode->next=tmp->next;**

**newnode->prev=tmp;**

**tmp->next->prev=newnode;**

**tmp->next=newnode;**

**}**

**else**

**{**

**printf("Element not found(T\_T)\n");**

**}**

**break;**

**case 4:** //Insertion After a Specefic Location Number

**printf("Enter the location number from Indexing of 0: ");**

**scanf("%d" , &loc);**

**count = 0;**

**tmp = head;**

**while(tmp != NULL && count<loc)**

**{**

**count++;**

**tmp = tmp->next;**

**}**

**if(tmp != NULL)**

**{**

**newnode->next = tmp->next;**

**newnode->prev = tmp;**

**tmp->next->prev = newnode;**

**tmp->next = newnode;**

**}**

**else**

**{**

**printf("Invalid location(Y\_Y).\n");**

**}**

**break;**

**default:**

**printf("Invalid choice.\n");**

**}**

**}**

**printf("Do you want to continue (y/n): ");** //Asks the user if he wants to continue Inserting

**scanf(" %c" , &ch);**

**}**

**tmp = head;**

**printf("The Doubly Circular Linked list is: ");**

**do**

**{**

**printf("%d " , tmp->data);** //Traverses through the list and prints the List

**tmp = tmp->next;**

**} while( tmp != head);**

**printf("\n\n");**

**printf("Do you Want to Delete A Node(y/n): ");**

**char del;**

**scanf(" %c" , &del);** //Asks the user if he wants to delete a node from the list

**while(del == 'y'|| del == 'Y')**

**{**

**if(head==NULL)** // if no node is present in the list

**{**

**printf("No Node Present In the Linked List\n");**

**}**

**else if(head->next==head)** //If only One Node is Present

**{**

**free(head);**

**printf("Deleted the Only Existing Node..\n");**

**head = NULL;**

**}**

**else**

**{**

**printf("1. Deletion at the beginning\n");**

**printf("2. Deletion at the end\n");**

**printf("3. Deletion of a specific element\n");**

**printf("Enter your choice: ");**

**int cho;**

**scanf("%d", &cho);** //Asks user for his choice

**switch (cho)**

**{**

**case 1:** //Deletion of Node At The Beginning

**tmp=head;**

**head=head->next;**

**head->prev=tmp->prev;**

**tmp->prev->next=head;**

**free(tmp);**

**break;**

**case 2:** //Deletion of Node At The End

**tmp = head->prev;**

**head->prev = tmp->prev;**

**tmp->prev->next = head;**

**free(tmp);**

**break;**

**case 3:**

//Asks the user the value to delete the node containing that value

**printf("Enter the element to delete: ");**

**int value;**

**scanf("%d", &value);**

**if (head!=NULL && head->data==value)**

**{** //If value foung at head or first node

**tmp=head->next;**

**tmp->prev=head->prev;**

**head->prev->next=tmp;**

**free(head);**

**head=tmp;**

**}**

**tmp=head;**

**while(tmp!=NULL && tmp->data!=value)**

**{** //if found somewhere else other than head

**ptmp=tmp;**

**tmp=tmp->next;**

**}**

**ptmp->next=tmp->next;**

**tmp->next->prev=ptmp;**

**break;**

**default:**

//If user choses Wrong Choice

**printf("Invalid choice.\n");**

**}**

**}**

**printf("Do you Want to Delete A Node(y/n): ");**

//Asks user if he wants to continue deleting

**scanf(" %c",&del);**

**}**

**tmp=head;**

**printf("The Doubly Circular Linked List After Deletion is:\n");**

**do**

**{**

**printf("The Prev Is: %p\n",tmp->prev);**

**printf("The Value Is: %d\n",tmp->data);**

//Traverses the entire linked List And Prints it.

**printf("The Next Is: %p\n",tmp->next);**

**tmp=tmp->next;**

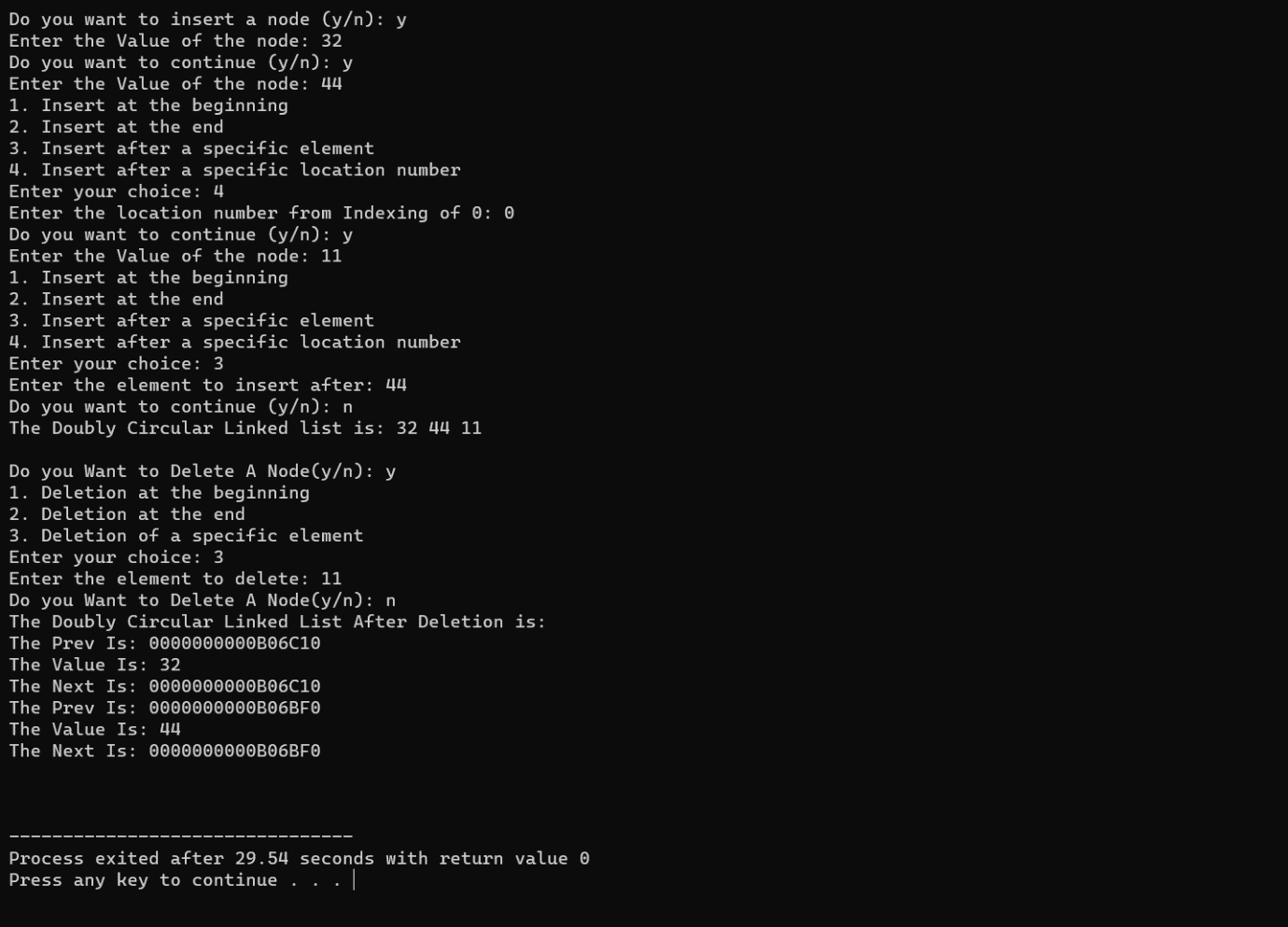
**} while(tmp!=head);**

**printf("\n\n");**

**return 0;**

**}**

**Output:-**

****

**Q9.) Menu Driven Program for Insertion And Deletion Of Nodes in a Circular Linked List.**

**Source Code:-**

**#include <stdio.h>**

**#include <stdlib.h>**

**struct node**

**{**

**int data;**

**struct node \*next;**

**};**

**int main()**

**{**

**struct node \*head, \*newnode, \*tmp;**

**head = NULL;**

**printf("Do you want to insert a node (y/n): ");**

**char ch;**

**scanf(" %c", &ch);**

**while (ch == 'y' || ch == 'Y')**

**{**

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

**if (newnode == NULL)**

**{**

**printf("No Space Available...Bye Bye\n");**

**break;**

**}**

**printf("Enter the Value of the node: ");**

**int val;**

**scanf("%d", &val);**

**newnode->data=val;**

**if (head == NULL)**

**{**

**head = newnode;**

**newnode->next = head; // Points back to itself to form a circular list**

**}**

**else**

**{**

**printf("1. Insert at the beginning\n");**

**printf("2. Insert at the end\n");**

**printf("3. Insert after a specific element\n");**

**printf("Enter your choice: ");**

**int choice;**

**scanf("%d", &choice);**

**switch (choice)**

**{**

**case 1: // Insertion At The Beginning**

**{**

**newnode->next = head;**

**tmp = head;**

**while (tmp->next != head)**

**{**

**tmp = tmp->next;**

**}**

**tmp->next = newnode;**

**head = newnode;**

**}**

**break;**

**case 2: // Insertion At the End**

**{**

**tmp = head;**

**while (tmp->next != head)**

**{**

**tmp = tmp->next;**

**}**

**tmp->next = newnode;**

**newnode->next = head;**

**}**

**break;**

**case 3: // Insertion After a Specific Element**

**{**

**printf("Enter the element to insert after: ");**

**int value;**

**scanf("%d", &value);**

**tmp = head;**

**while (tmp->next != head && tmp->data != value)**

**{**

**tmp = tmp->next;**

**}**

**if (tmp->data != value)**

**{**

**printf("Element not found(T\_T)\n");**

**free(newnode);**

**break;**

**}**

**newnode->next = tmp->next;**

**tmp->next = newnode;**

**}**

**break;**

**default:**

**printf("Invalid choice.\n");**

**free(newnode);**

**break;**

**}**

**}**

**printf("Do you want to continue (y/n): ");**

**scanf(" %c", &ch);**

**}**

**// Print the circular linked list**

**if (head != NULL)**

**{**

**printf("The Circular Linked list is: ");**

**tmp = head;**

**do**

**{**

**printf("%d ", tmp->data);**

**tmp = tmp->next;**

**} while (tmp != head);**

**printf("\n\n");**

**}**

**// Deletion part**

**printf("Do you Want to Delete A Node(y/n): ");**

**char del;**

**scanf(" %c", &del);**

**while (del == 'y' || del == 'Y')**

**{**

**if (head == NULL)**

**{**

**printf("No Node Present In the Linked List\n");**

**break;**

**}**

**else**

**{**

**printf("1. Deletion at the beginning\n");**

**printf("2. Deletion at the end\n");**

**printf("3. Deletion of a specific element\n");**

**printf("Enter your choice: ");**

**int cho;**

**scanf("%d", &cho);**

**switch (cho)**

**{**

**case 1: // Deletion at The Beginning**

**{**

**tmp = head;**

**while (tmp->next != head)**

**{**

**tmp = tmp->next;**

**}**

**tmp->next = head->next;**

**free(head);**

**head = tmp->next;**

**}**

**break;**

**case 2: // Deletion at the End**

**{**

**tmp = head;**

**while (tmp->next->next != head)**

**{**

**tmp = tmp->next;**

**}**

**free(tmp->next);**

**tmp->next = head;**

**}**

**break;**

**case 3: // Deletion of a Specific Element**

**{**

**printf("Enter the element to delete: ");**

**int value;**

**scanf("%d", &value);**

**tmp = head;**

**while (tmp->next != head && tmp->next->data != value)**

**{**

**tmp = tmp->next;**

**}**

**if (tmp->next->data != value)**

**{**

**printf("Element not found.\n");**

**break;**

**}**

**struct node \*temp = tmp->next;**

**tmp->next = temp->next;**

**free(temp);**

**}**

**break;**

**default:**

**printf("Invalid choice.\n");**

**break;**

**}**

**}**

**printf("Do you Want to Delete A Node(y/n): ");**

**scanf(" %c", &del);**

**}**

**// Print the circular linked list after deletion**

**if (head != NULL)**

**{**

**printf("The Circular Linked list after deletion is:\n");**

**tmp = head;**

**do**

**{**

**printf("The Value Is:%d\n", tmp->data);**

**printf("The Next Is:%p ", tmp->next);**

**tmp = tmp->next;**

**} while (tmp != head);**

**printf("\n\n");**

**}**

**return 0;**

**}**

**Output:-**



**Q10.) Menu Driven Program for Insertion And Deletion Of Nodes in a Doubly Linked List.**

**Source Code:-**

**#include <stdio.h>**

**#include <stdlib.h>**

**struct node**

**{**

**struct node \*prev;**

**int data;**

**struct node \*next;**

**};**

**int main()**

**{**

**struct node \*head, \*newnode, \*tmp;**

**head = NULL;**

**printf("Do you want to insert a node (y/n): ");**

**char ch;**

**scanf(" %c", &ch);**

**while (ch == 'y' || ch == 'Y')**

**{**

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

**if (newnode == NULL)**

**{**

**printf("No Space Available...Bye Bye\n");**

**break;**

**}**

**printf("Enter the Value of the node: ");**

**int val;**

**scanf("%d", &val);**

**newnode->data=val;**

**if (head == NULL)**

**{**

**head = newnode;**

**newnode->next = head;**

**newnode->prev = head;**

**}**

**else**

**{**

**printf("1. Insert at the beginning\n");**

**printf("2. Insert at the end\n");**

**printf("3. Insert after a specific element\n");**

**printf("Enter your choice: ");**

**int choice;**

**scanf("%d", &choice);**

**switch (choice)**

**{**

**case 1: // Insertion At The Beginning**

**newnode->next = head;**

**newnode->prev = head->prev;**

**head->prev->next = newnode;**

**head->prev = newnode;**

**head = newnode;**

**break;**

**case 2: // Insertion At the End**

**newnode->next = head;**

**newnode->prev = head->prev;**

**head->prev->next = newnode;**

**head->prev = newnode;**

**break;**

**case 3: // Insertion After a Specific Element**

**printf("Enter the element to insert after: ");**

**int value;**

**scanf("%d", &value);**

**tmp = head;**

**while (tmp != NULL && tmp->data != value)**

**{**

**tmp = tmp->next;**

**}**

**if (tmp == NULL)**

**{**

**printf("Element not found(T\_T)\n");**

**free(newnode);**

**break;**

**}**

**newnode->next = tmp->next;**

**newnode->prev = tmp;**

**tmp->next->prev = newnode;**

**tmp->next = newnode;**

**break;**

**default:**

**printf("Invalid choice.\n");**

**free(newnode);**

**break;**

**}**

**}**

**printf("Do you want to continue (y/n): ");**

**scanf(" %c", &ch);**

**}**

**// Print the doubly linked list**

**if (head != NULL)**

**{**

**printf("The Doubly Linked list is: ");**

**tmp = head;**

**do**

**{**

**printf("%d ", tmp->data);**

**tmp = tmp->next;**

**} while (tmp != head);**

**printf("\n\n");**

**}**

**// Deletion part**

**printf("Do you Want to Delete A Node(y/n): ");**

**char del;**

**scanf(" %c", &del);**

**while (del == 'y' || del == 'Y')**

**{**

**if (head == NULL)**

**{**

**printf("No Node Present In the Linked List\n");**

**break;**

**}**

**else**

**{**

**printf("1. Deletion at the beginning\n");**

**printf("2. Deletion at the end\n");**

**printf("3. Deletion of a specific element\n");**

**printf("Enter your choice: ");**

**int cho;**

**scanf("%d", &cho);**

**switch (cho)**

**{**

**case 1: // Deletion at The Beginning**

**tmp = head;**

**head = head->next;**

**head->prev = tmp->prev;**

**tmp->prev->next = head;**

**free(tmp);**

**break;**

**case 2: // Deletion at the End**

**tmp = head->prev;**

**head->prev = tmp->prev;**

**tmp->prev->next = head;**

**free(tmp);**

**break;**

**case 3: // Deletion of a Specific Element**

**printf("Enter the element to delete: ");**

**int value;**

**scanf("%d", &value);**

**tmp = head;**

**while (tmp != NULL && tmp->data != value)**

**{**

**tmp = tmp->next;**

**}**

**if (tmp == NULL)**

**{**

**printf("Element not found.\n");**

**break;**

**}**

**tmp->prev->next = tmp->next;**

**tmp->next->prev = tmp->prev;**

**free(tmp);**

**break;**

**default:**

**printf("Invalid choice.\n");**

**break;**

**}**

**}**

**printf("Do you Want to Delete A Node(y/n): ");**

**scanf(" %c", &del);**

**}**

**// Print the doubly linked list after deletion**

**if (head != NULL)**

**{**

**printf("The Doubly Linked list after deletion is:\n");**

**tmp = head;**

**do**

**{**

**printf("The Prev Is:%p\n", tmp->prev);**

**printf("The Value Is:%d\n", tmp->data);**

**printf("The Next Is:%p\n", tmp->next);**

**tmp = tmp->next;**

**} while (tmp != head);**

**printf("\n\n");**

**}**

**return 0;**

**}**

**Output:-**

