Data Structure

C program to insert an element in an array

```
#include <stdio.h>

int main()

{

int n;

scanf(“%d”,&n);

int arr[n];

int i;

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

{

scanf(“%d”,&arr[i]);

}

int pos;

scanf(“%d”,&pos);

int ele;

scanf(“%d”,&ele);

if(pos > n)

printf(“Invalid Input”);

else

{

for (i = n – 1; i >= pos – 1; i–)

arr[i+1] = arr[i];
```

```
arr[pos-1] = ele;

printf(“Array after insertion is:\n”);

for (i = 0; i <= n; i++)
printf(“%d\n”, arr[i]);
}

return 0;
}
```

C program to delete an element in an array */

```
#include <stdio.h>

int main()
{
int array[100], position, c, n;

printf(“Enter the number of elements of the array : “);
scanf(“%d”, &n);

printf(“\nInput the array elements : “);

for (c = 0; c < n; c++)
scanf(“%d”, &array[c]);
```

```
printf(“\nEnter the position : “);

scanf(“%d”, &position);


if (position >= n+1)

printf(“\nDeletion not possible.\n”);

else

{

for (c = position – 1; c < n – 1; c++)

array[c] = array[c+1];


printf(“\nArray after deletion : “);


for (c = 0; c < n – 1; c++)

printf(“%d\n”, array[c]);

}


return 0;

}
```

C program to delete an element in an array */

```
#include <stdio.h>


int main()

{
```

```
int array[100], position, c, n;

printf(“Enter the number of elements of the array : “);
scanf(“%d”, &n);

printf(“\nInput the array elements : “);

for (c = 0; c < n; c++)
scanf(“%d”, &array[c]);

printf(“\nEnter the position : “);
scanf(“%d”, &position);

if (position >= n+1)
printf(“\nDeletion not possible.\n”);
else
{
for (c = position – 1; c < n – 1; c++)
array[c] = array[c+1];

printf(“\nArray after deletion : “);

for (c = 0; c < n – 1; c++)
printf(“%d\n”, array[c]);
}
```

return 0;

}

Mearged Two Array

```c
#include <stdio.h>

int main()

{

    int n1,n2,n3;              //Array Size Declaration

    int a[10000], b[10000], c[20000];

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

    scanf("%d",&n1);

    printf("Enter the array elements: ");

    for(int i = 0; i < n1; i++)

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

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

        scanf("%d",&n2);

    printf("Enter the array elements: ");

    for(int i = 0; i < n2; i++)

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

    n3 = n1 + n2;

    for(int i = 0; i < n1; i++)
```

```
    c[i] = a[i];

for(int i = 0; i < n2; i++)

    c[i + n1] = b[i];


printf("The merged array: ");

for(int i = 0; i < n3; i++)

    printf("%d ", c[i]);          //Print the merged array


printf("\nFinal array after sorting: ");

for(int i = 0; i < n3; i++){

    int temp;

    for(int j = i + 1; j < n3; j++) {

        if(c[i] > c[j]) {

            temp = c[i];

            c[i] = c[j];

            c[j] = temp;

        }

    }

}

for(int i = 0; i < n3 ; i++)          //Print the sorted Array
```

```
        printf(" %d ",c[i]);

    return 0;

}
```

Write a program Insert, traversing, Delete, Sorting a 2-Dimension Array.

```
#include <stdio.h>
int main()
{
int b[2][3];
int i,j,num;
printf("Enter elements into 2-D array: ");
for(i=0;i<2;i++)
{
for(j=0;j<3;j++)
{
scanf("%d" , &b[i][j]);
}
}
}
```

Update

```
#include <stdio.h>
int main()
{
int b[2][3];
int i,j,num;
printf("Enter elements into 2-D array: ");
for(i=0;i<2;i++)
{
for(j=0;j<3;j++)
{
scanf("%d" , &b[i][j]);
}
}
```

```
b[0][2]=10;
for(i=0;i<2;i++)
{
for(j=0;j<3;j++)
{
printf("\t%d" , b[i][j]);
}
printf("\n");
}
return 0;
}
```

Delete

```
#include <stdio.h>
int main()
{
int b[2][3],i,j,num,x;
printf("Enter elements into 2-D array: ");
for(i=0;i<2;i++)
{
for(j=0;j<3;j++)
{
scanf("%d" , &b[i][j]);
}
}
printf("Enter the value of row number :");
scanf("%d", &x);
for(i=0;i<2;i++)
{
if(i==x)
{
for(j=0;j<3;j++)
{
if((i+1)<2)
{
printf("\t%d" , b[i+1][j]);
}
}
i++;}
else
```

```
{
for(j=0;j<3;j++)
{
printf("\t%d" , b[i][j]);
}
}
printf("\n");
}
}
```

Matrix operation

Write a program for Addition, Subtraction, Multiplication and Transpose of matrices.

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

// function to add two 3x3 matrix
void add(int m[3][3], int n[3][3], int sum[3][3])
{
  for(int i=0;i<3;i++)
    for(int j=0;j<3;j++)
      sum[i][j] = m[i][j] + n[i][j];
}

// function to subtract two 3x3 matrix
void subtract(int m[3][3], int n[3][3], int result[3][3])
{
  for(int i=0;i<3;i++)
    for(int j=0;j<3;j++)
      result[i][j] = m[i][j] - n[i][j];
}

// function to multiply two 3x3 matrix
void multiply(int m[3][3], int n[3][3], int result[3][3])
{
  for(int i=0; i < 3; i++)
  {
    for(int j=0; j < 3; j++)
    {
      result[i][j] = 0; // assign 0
      // find product
      for (int k = 0; k < 3; k++)
      result[i][j] += m[i][k] * n[k][j];
    }
```

```
  }
}

// function to find transpose of a 3x3 matrix
void transpose(int matrix[3][3], int trans[3][3])
{
  for (int i = 0; i < 3; i++)
    for (int j = 0; j < 3; j++)
      trans[i][j] = matrix[j][i];
}

// function to display 3x3 matrix
void display(int matrix[3][3])
{
  for(int i=0; i<3; i++)
  {
    for(int j=0; j<3; j++)
      printf("%d\t",matrix[i][j]);

    printf("\n"); // new line
  }
}

// main function
int main()
{
  // matrix
  int a[][3] = { {5,6,7}, {8,9,10}, {3,1,2} };
  int b[][3] = { {1,2,3}, {4,5,6}, {7,8,9} };
  int c[3][3];

  // print both matrix
  printf("First Matrix:\n");
  display(a);
  printf("Second Matrix:\n");
  display(b);

  // variable to take choice
  int choice;

  // menu-driven
  do
  {
    // menu to choose the operation
    printf("\nChoose the matrix operation,\n");
    printf("---------------------------\n");
    printf("1. Addition\n");
    printf("2. Subtraction\n");
    printf("3. Multiplication\n");
```

```
    printf("4. Transpose\n");
    printf("5. Exit\n");
    printf("---------------------------\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    switch (choice) {
      case 1:
        add(a, b, c);
        printf("Sum of matrix: \n");
        display(c);
        break;
      case 2:
        subtract(a, b, c);
        printf("Subtraction of matrix: \n");
        display(c);
        break;
      case 3:
        multiply(a, b, c);
        printf("Multiplication of matrix: \n");
        display(c);
        break;
      case 4:
        printf("Transpose of the first matrix: \n");
        transpose(a, c);
        display(c);
        printf("Transpose of the second matrix: \n");
        transpose(b, c);
        display(c);
        break;
      case 5:
        printf("Thank You.\n");
        exit(0);
      default:
        printf("Invalid input.\n");
        printf("Please enter the correct input.\n");
    }
  }while(1);

  return 0;
}
```

```
#include <stdio.h>

#include <stdlib.h>

#define SIZE 4
```

```c
int top = -1, inp_array[SIZE];
void push();
void pop();
void show();

int main()
{
    int choice;

    while (1)
    {
        printf("\nPerform operations on the stack:");
        printf("\n1.Push the element\n2.Pop the element\n3.Show\n4.End");
        printf("\n\nEnter the choice: ");
        scanf("%d", &choice);

        switch (choice)
        {
        case 1:
            push();
            break;
        case 2:
            pop();
            break;
        case 3:
            show();
            break;
        case 4:
            exit(0);

        default:
            printf("\nInvalid choice!!");
        }
    }
}

void push()
```

```
{
    int x;

    if (top == SIZE - 1)
    {
        printf("\nOverflow!!");
    }
    else
    {
        printf("\nEnter the element to be added onto the stack: ");
        scanf("%d", &x);
        top = top + 1;
        inp_array[top] = x;
    }
}

void pop()
{
    if (top == -1)
    {
        printf("\nUnderflow!!");
    }
    else
    {
        printf("\nPopped element: %d", inp_array[top]);
        top = top - 1;
    }
}

void show()
{
    if (top == -1)
    {
        printf("\nUnderflow!!");
    }
    else
    {
        printf("\nElements present in the stack: \n");
```

```
        for (int i = top; i >= 0; --i)
            printf("%d\n", inp_array[i]);
    }
}
```

```
#include<stdio.h>
#include<conio.h>
#define MAX 3

int a[MAX], top = -1;
void push();
void pop();
void peep();
void change();
void display();

void main()
{
 int ch;
 clrscr();
 while(1) {
   printf("\n1. PUSH or INSERT");
 printf("\n2. POP or DELETE");
 printf("\n3. PEEP or SEARCH");
 printf("\n4. CHANGE or UPDATE");
 printf("\n5. Display");
 printf("\n6. End program");
   printf("\nEnter Choice : ");
   scanf("%d",&ch);
   clrscr();
   switch(ch)
   {
    case 1:
{
  push();
  break;
}
    case 2:
{
  pop();
  break;
}
case 3:
    {
 peep();
```

```
  break;
      }
      case 4:
      {
  change();
  break;
      }
      case 5:
{
   display();
   break;
}
      case 6:
{
   exit(0);
}
      default:
 printf("\ninvalid choice !!!");
    } // switch close
    getch();
 }
 //getch();
}

void push(){
 int data;
 if(top==MAX-1)
 {
    printf("\noverflow or stack is full !!!");
 }
 else
 {
    printf("\nEnter the element : ");
    scanf("%d",&data);
    top++;
    a[top]=data;
 }
}

void pop()
{
 if(top==-1)
 {
   printf("\nunder flow STACK or STACK is empty");
 }

 else
```

```
 {
   printf("\nPOP or DELETE element :  %d",a[top]);
   top--;
 }
}

void display()
{
  int i;
  if(top>=0)
  {
     printf("\nElemets : ");
     for(i=top; i>=0; i--)
     {
 printf("\n%d",a[i]);
     }
  }
  else
  {
     printf("\nThe STACK is Empty");
  }
}
void peep()
{
  int p;
  printf("\nEnter the position : ");
  scanf("%d",&p);
  if(top-p<=-1)
  {
    printf("\nSTACK is overflow !!!");
  }
  else
  {
    printf("\nThe Elements is : %d",a[top-p]);
  }
}
void change()
{
 int v1,v2;
 printf("\nEnter Position for change : ");
 scanf("%d",&v1);
 printf("\nEneter the Number for change : ");
 scanf("%d",&v2);
 if(top-v1<=-1)
 {
   printf("\nSTACK is overflow !!!");
 }
```

```
  else
  {
    a[top-v1]=v2;
    printf("\nCHANGE successfull !!!");
  }
}
```