**Job No**: 01

**Job Name**: Linear array data insert and deletion program.

**Data insert program:**

**Code:**

#include <stdio.h>

#include <conio.h>

void main() {

int a[8] = { 4, 6, 7, 9, 8, 2 };

int item = 3, k = 2, n = 6 - 1;

int i = 0, j = n;

printf("Original array Contains......\n");

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

printf("Array[%d] = %d \n", i, a[i]);

}

n = n + 1;

while (j >= k) {

a[j + 1] = a[j];

j = j - 1;

}

a[k] = item;

printf("After insert a new element the array contains.....\n");

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

printf("Array [%d] = %d \n", i, a[i]);

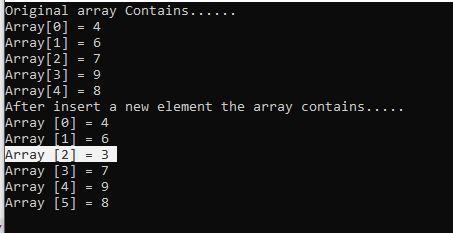
}

getch();

return 0

}

Output:



Data delete program:

**code:**

#include <stdio.h>

#include <conio.h>

void main() {

int a[] = { 1, 3, 5, 7, 8 };

int k = 3, n = 5;

int i, j;

printf("The original array elements are :\n");

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

printf("Array[%d] = %d \n", i, a[i]);

}

j = k;

while (j < n) {

a[j - 1] = a[j];

j = j + 1;

}

n = n - 1;

printf("The array elements after deletion :\n");

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

printf("Array[%d] = %d \n", i, a[i]);

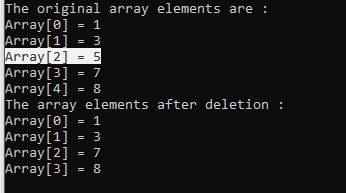
}

getch();

return 0;

}

Output:



**Job No**: 02

**Job Name**: Two Matrix multiplication program & test.

Code:

#include <stdio.h>

#define SIZE 3

int main()

{

int A[SIZE][SIZE];

int B[SIZE][SIZE];

int C[SIZE][SIZE];

int row, col, i, sum;

printf("Enter elements in matrix A of size %dx%d: \n", SIZE, SIZE);

for (row = 0; row<SIZE; row++)

{

for (col = 0; col<SIZE; col++)

{

scanf("%d", &A[row][col]);

}

}

printf("\nEnter elements in matrix B of size %dx%d: \n", SIZE, SIZE);

for (row = 0; row<SIZE; row++)

{

for (col = 0; col<SIZE; col++)

{

scanf("%d", &B[row][col]);

}

}

for (row = 0; row<SIZE; row++)

{

for (col = 0; col<SIZE; col++)

{

sum = 0;

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

{

sum += A[row][i] \* B[i][col];

}

C[row][col] = sum; } }

printf("\nProduct of matrix A \* B = \n");

for (row = 0; row<SIZE; row++)

{

for (col = 0; col<SIZE; col++)

{

printf("%d ", C[row][col]);

}

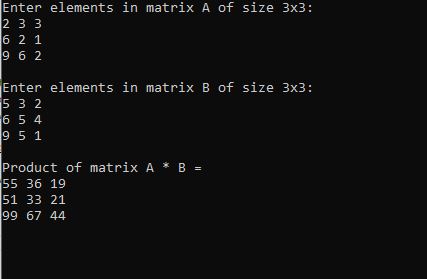
printf("\n");

}

getch();

return 0;

}

Output:

**Job No**: 03

**Job Name**: Linked list data PUSH and POP program.

Code:

#include <stdio.h>

#include <stdlib.h>

struct Node

{

int Data;

struct Node \*next;

}\*top;

void popStack()

{

struct Node \*temp, \*var=top;

if(var==top)

{

top = top->next;

free(var);

}

else

printf("\nStack Empty");

}

void push(int value)

{

struct Node \*temp;

temp=(struct Node \*)malloc(sizeof(struct Node));

temp->Data=value;

if (top == NULL)

{

top=temp;

top->next=NULL;

}

else

{

temp->next=top;

top=temp;

}

}

void display()

{

struct Node \*var=top;

if(var!=NULL)

{

printf("\nElements are as:\n");

while(var!=NULL)

{

printf("\t%d\n",var->Data);

var=var->next;

}

printf("\n");

}

else

printf("\nStack is Empty");

}

int main(int argc, char \*argv[])

{

int i=0;

top=NULL;

printf(" \n1. Push to stack");

printf(" \n2. Pop from Stack");

printf(" \n3. Display data of Stack");

printf(" \n4. Exit\n");

while(1)

{

printf(" \nChoose Option: ");

scanf("%d",&i);

switch(i)

{

case 1:

{

int value;

printf("\nEnter a valueber to push into Stack: ");

scanf("%d",&value);

push(value);

display();

break;

}

case 2:

{

popStack();

display();

break;

}

case 3:

{

display();

break;

}

case 4:

{

struct Node \*temp;

while(top!=NULL)

{

temp = top->next;

free(top);

top=temp;

}

exit(0);

}

default:

{

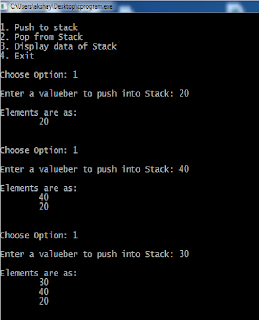
printf("\nwrong choice for operation");

}

}

}

**Output:**



**Job No**: 4

**Job Name**: Stack PUSH & POP operation program.

Code:

#include <stdio.h>

#include <conio.h>

#define MAX 5

int stack[MAX];

int top = 0;

void push (int value)

{

if (top == MAX)

{

printf(“!!! Overflow !!!”);

getch();

}

else

{

stack[top] = value;

top++;

}

}

int pop(void)

{

top--;

return stack[top];

}

void show(void)

{

int i;

if (top == 0) printf(“Stack is empty”);

else

{

printf(“\*\*\*\*Stack\*\*\*\*\n);

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

printf(“%d “, stack[i]);

}}

void main()

{

int option;

int x;

clrscr();

do

{

clrscr();

gotoxy(10, 5); printf(“###### MAIN MENU ######”);

gotoxy(10, 6); printf(“====================)

gotoxy(10, 7); printf(“1.Push”);

gotoxy(10, 8); printf(“2.Pop”);

gotoxy(10, 9); printf(3.Show”);

gotoxy(10, 10); printf(4.Exit”);

gotoxy((10, 12); printf(“Enter your selection[1..4]:”);

scanf(“%d”, &option);

clrscr();

switch(option)

{ case 1:

print(“Enter your value to push: “);

scanf(“%d”, &x);

push(x);

break;

case 2:

if (top != 0)

printf(“The poped value is %d”, pop());

else

printf(“!!! underflow !!!”);

getch();

break;

case 3:

show();

getch();

}

}

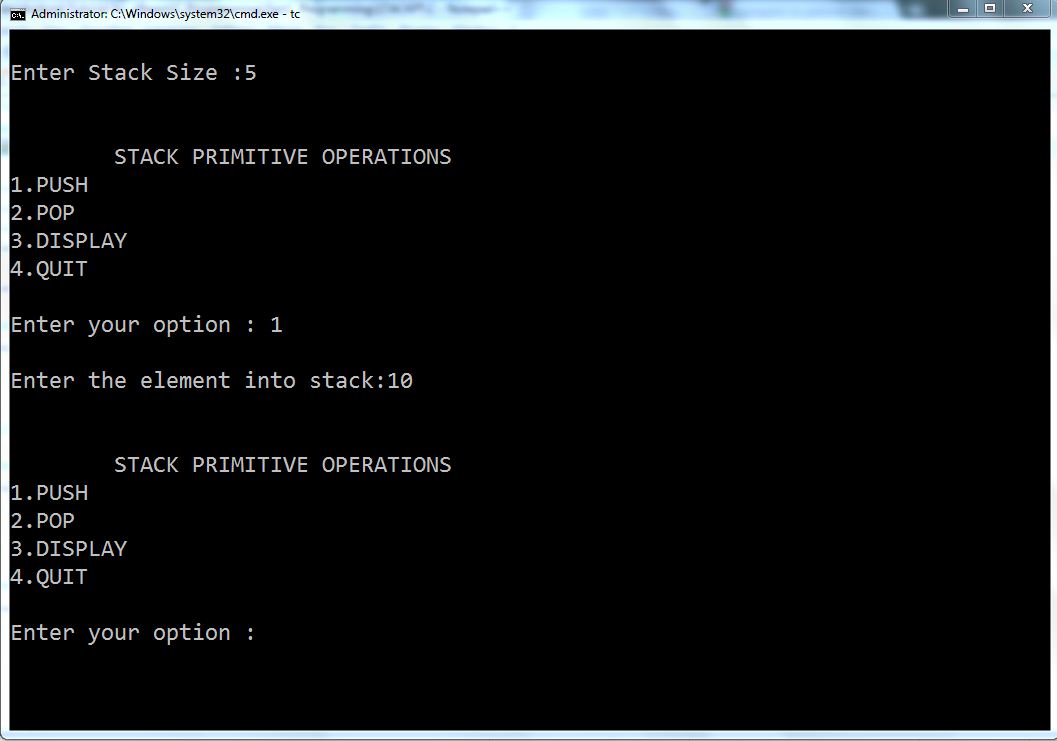
while(option != 4);

gotoxy(10, 22); printf(“Press any key to continue …”);

getch();

}

**Output:**



**Job No**: 05

**Job Name:** Write a program for calculating factorial N number and fibonacci number using Recursion.

## Factorial of a Number Using Recursion:

## Code:

#include <stdio.h>

long int multiplyNumbers(int n);

int main()

{

int n;

printf("Enter a positive integer: ");

scanf("%d", &n);

printf("Factorial of %d = %ld", n, multiplyNumbers(n));

return 0;

}

long int multiplyNumbers(int n)

{

if (n >= 1)

return n\*multiplyNumbers(n-1);

else

return 1;

}

**Output:**

Enter a positive integer: 6

Factorial of 6 = 720

## Fibonaci of a Number Using Recursion:

## Code:

## #include <stdio.h>

## int fibo(int);

int main()

{

int num;

int result;

printf("Enter the nth number in fibonacci series: ");

scanf("%d", &num);

if (num < 0)

{

printf("Fibonacci of negative number is not possible.\n");

}

else

{

result = fibo(num);

printf("The %d number in fibonacci series is %d\n", num, result);

}

return 0;

}

int fibo(int num)

{

if (num == 0)

{

return 0;

}

else if (num == 1)

{

return 1;

}

else

{

return(fibo(num - 1) + fibo(num - 2));

}

}

**Output:**

Enter the nth number in fibonacci series: 8

The 8 number in fibonacci series is 21

Enter the nth number in fibonacci series: 12

The 12 number in fibonacci series is 144

**Job No**: 06

**Job Name:** Infix to Postfix program operation Convert in C.

Code:

#include<stdio.h>

char stack[20];

int top = -1;

void push(char x)

{

    stack[++top] = x;

}

char pop()

{

    if(top == -1)

        return -1;

    else

        return stack[top--];

}

int priority(char x)

{

    if(x == '(')

        return 0;

    if(x == '+' || x == '-')

        return 1;

    if(x == '\*' || x == '/')

        return 2;

}

main()

{

    char exp[20];

    char \*e, x;

    printf("Enter the expression :: ");

    scanf("%s",exp);

    e = exp;

    while(\*e != '\0')

    {

        if(isalnum(\*e))

            printf("%c",\*e);

        else if(\*e == '(')

            push(\*e);

        else if(\*e == ')')

        {

            while((x = pop()) != '(')

                printf("%c", x);

        }

        else

        {

            while(priority(stack[top]) >= priority(\*e))

                printf("%c",pop());

            push(\*e);

        }

        e++;

    }

    while(top != -1)

    {

        printf("%c",pop());

    }

}

## OUTPUT:

Enter the expression :: a+b\*c

abc\*+

Enter the expression :: (a+b)\*c+(d-a)

ab+c\*da-+

**Job No**: 07

**Job Name**: Writing programs to insert and delete data from Queue operations.

Code:

#include<stdio.h>

#include<conio.h>

int i,queue[30],front=1,rear,max;

void insert()

{

clrscr();

if(rear==max)

{

print("\nOverflow"):

getch();

}

else

{

rear++;

printf("Item:");scanf("%d",&queue[rear]);

}

}

void delete()

{

clrscr();

if(rear==1)

{

printf("\nUnderflow");

getch();

}

else {

printf("Item:%d\n",queue[front]);

front++;

// rear--;

}

getch();

}

void show()

{

clrscr();

printf("\nitem:\n(\n");

for(i=front; i<=rear; i++) {

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

}

printf(" )");

getch();

}

void main() {

char ch;

clrscr();

printf("How many sell on Queue..(0..30)");scanf("%d",&max);

clrscr();

do{

clrscr();

printf("1.Insert\n2.Delete\n3.Show\n4.Enter for Exit");

ch=getch();

switch(ch){

case'1':insert();break;

case'2':delet();break;

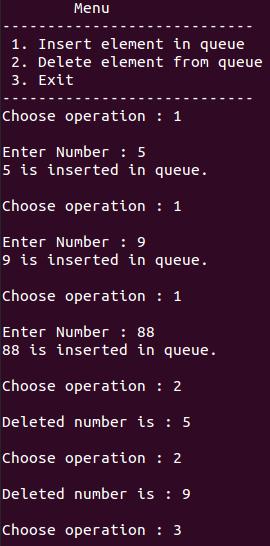
case'3':show();break;

}

}while(ch!='\r');

}

**Output:**



**Job No**: 08

**Job Name**: write a program to search an item from an array using linear search algorithm.

Code:

#include <stdio.h>

#include <conio.h>

int main(){

    int inputArray[100], elementCount, counter, num;

    printf("Enter Number of Elements in Array\n");

    scanf("%d", &elementCount);

    printf("Enter %d numbers \n", elementCount);

    /\* Read array elements \*/

    for(counter = 0; counter < elementCount; counter++){

        scanf("%d", &inputArray[counter]);

    }

    printf("Enter a number to serach in Array\n");

    scanf("%d", &num);

    /\* search num in inputArray from index 0 to elementCount-1 \*/

    for(counter = 0; counter < elementCount; counter++){

        if(inputArray[counter] == num){

            printf("Number %d found at index %d\n", num, counter);

            break;

        }

    }

    if(counter == elementCount){

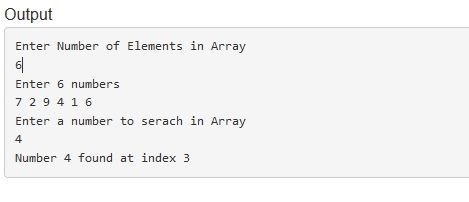
     printf("Number %d Not Present in Input Array\n", num);

    }

    getch();

    return 0;

}



**Job No**: 09

**Job Name:** write a program to sort n data in ascending order using bubble sort algorithm.

Code:

#include <stdio.h>

#define MAXSIZE 10

void main()

{

int array[MAXSIZE];

int i, j, num, temp;

printf("Enter the value of num \n");

scanf("%d", &num);

printf("Enter the elements one by one \n");

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

{

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

}

printf("Input array is \n");

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

{

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

}

/\* Bubble sorting begins \*/

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

{

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

{

if (array[j] > array[j + 1])

{

temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

}

}

}

printf("Sorted array is...\n");

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

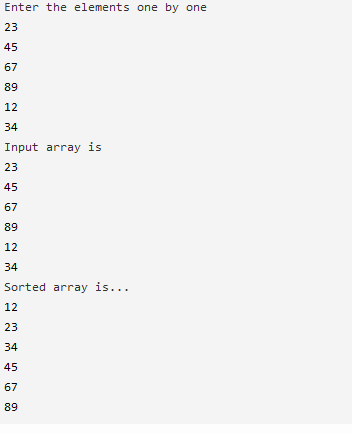
{

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

}

}

**Output:**



**Job No**: 10

**Job Name:** Write a program to arrange Data Ascending and descending using Quick sort Algorithm.

Code

#include <stdio.h>

#include <conio.h>

int main()

{

int a[10] = { 3,4,7,6,5,1,2,8,10,9 };

int n = 10

size

printf("\n\nArray Data : ");

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

of array

{

printf(" %d ", a[i]);

}

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

{

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

values

{

if (a[j] > a[i])

{

int tmp = a[i];

storing last value

a[i] = a[j];

a[j] = tmp;

}

}

}

printf("\n\nAscending : ");

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

after sorting

{

printf(" %d ", a[i]);

}

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

{

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

values

{

if (a[j] < a[i])

{

int tmp = a[i];

storing last value

a[i] = a[j];

a[j] = tmp; }

}

}

printf("\n\nDescending : ");

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

after sorting

{

printf(" %d ", a[i]);

}

\_getch();

user to enter any key

return 0;

}

**Output:**

