**MCA DS Lab Cycle**

**Section 1: Basics**

1. Write programs to demonstrate the use of storage classes in C.

#include<stdio.h>

int a=5,b;               //global variables

void print()

{

    printf("\nThe value of global variable a is %d which is assigned\n",a);

    printf("\nThe value of global variable b is %d default value is zero\n ",b);

}

void display()

{

    static int i;            //sattic variable

    printf("\nvalue of static variable i is %d which is the default value\n",i);

    static int k=1;

    printf("\nassigned value of static variable k=%d \n",k);

    k++;

}

void reg()                   //register

{

    int register h=3;

    printf("\nThe value of h=%d which is a register\n",h);

}

int main()

{

    int c;                 //local variable

    printf("\nThe value of local variable c is %d which is a garbage value\n",c);

    print();

    display();

    reg();

    return 0;

}

1. Use a menu-driven program to insert, search, delete and sort elements in an array using functions (use global variables)

#include<stdio.h>

int ar[10];

void insert( int n)

{

    int i;

    for(i=0;i<n;i++)

    {

        scanf("%d",&ar[i]);

    }

}

void display( int n)

{

    int i;

    if(n>0){

        printf("Array elements are: ");

        for(i=0;i<n;i++)

        {

            printf("%d",ar[i]);

            if(i<n-1) printf(", ");

        }

        printf("\n");

    }

    else

    {

        printf("Array is empty\n");

    }

}

void delete(int n, int x)

{

    if(n==-1){

        printf("array is empty");

    }

    else{

        printf("last element of array is deleted");

        n=n-1;

    }

}

void sort( int n)

{

    int i, j; //declare i and j here

    for (i = 0; i < n - 1; i++) {

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

            if (ar[j] > ar[j + 1]) {

                int temp = ar[j];

                ar[j] = ar[j + 1];

                ar[j + 1] = temp;

            }

        }

    }

    printf("Array sorted successfully.\n");

    return; //return from the function

}

void search( int n, int x)

{

    int i, flag = 0;

    for (i = 0; i < n; i++) {

        if (ar[i] == x) {

            flag = 1;

            printf("Element %d found at index %d.\n", x, i);

            break;

        }

    }

    if (flag == 0) {

        printf("Element %d not found in the array.\n", x);

    }

    return;

}

int menu()

{

    int ch;

    printf("INSERT-1\nDELETE-2\nDISPLAY-3\nSORT-4\nSEARCH-5\nEXIT-6\nENTER YOUR CHOICE: ");

    scanf("%d",&ch);

    return ch;

}

void process()

{

    int ch;

    int n;

    int x;

    int ar[10];

    for(ch=menu();ch!=6;ch=menu())

    {

        switch(ch)

        {

            case 1:

                printf("Enter the value of n: ");

                scanf("%d",&n);

                if(n<=0){

                    printf("Invalid value of n\n");

                    break;

                }

                printf("Enter the elements: ");

                insert(n);

                break;

            case 2:

                break;

            case 3:

                display(n);

                break;

            case 4:

               sort(n);

               break;

            case 5:

                printf("enter element to search");

                scanf("%d",&x);

                search(n,x);

                break;

            default:

               printf("Wrong choice\n");

               break;

        }

    }

}

int main()

{

    process();

    return 0;

}

1. Use a menu-driven program to insert, search, delete and sort elements in an array using functions (use only local variables)

#include<stdio.h>

void insert(int ar[], int n)

{

    int i;

    for(i=0;i<n;i++)

    {

        scanf("%d",&ar[i]);

    }

}

void display(int ar[], int n)

{

    int i;

    if(n>0){

        printf("Array elements are: ");

        for(i=0;i<n;i++)

        {

            printf("%d",ar[i]);

            if(i<n-1) printf(", ");

        }

        printf("\n");

    }

    else

    {

        printf("Array is empty\n");

    }

}

int delete(int ar[], int n)

{

    if(n==-1){

        printf("array is empty");

    }

    else{

        printf("last element of array is deleted");

        n=n-1;

    }

    return n;

}

void sort(int ar[], int n)

{

    int i, j;

    for (i = 0; i < n - 1; i++) {

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

            if (ar[j] > ar[j + 1]) {

                int temp = ar[j];

                ar[j] = ar[j + 1];

                ar[j + 1] = temp;

            }

        }

    }

    printf("Array sorted successfully.\n");

    return;

}

void search(int ar[], int n, int x)

{

    int i, flag = 0;

    for (i = 0; i < n; i++) {

        if (ar[i] == x) {

            flag = 1;

            printf("Element %d found at index %d.\n", x, i);

            break;

        }

    }

    if (flag == 0) {

        printf("Element %d not found in the array.\n", x);

    }

    return;

}

int menu()

{

    int ch;

    printf("\nINSERT-1\nDELETE-2\nDISPLAY-3\nSORT-4\nSEARCH-5\nEXIT-6\nENTER YOUR CHOICE: ");

    scanf("%d",&ch);

    return ch;

}

void process()

{

    int ch;

    int x;

    int ar[10];

    int n = -1;

    for(ch=menu();ch!=6;ch=menu())

    {

        switch(ch)

        {

            case 1:

                printf("Enter the value of n: ");

                scanf("%d",&n);

                if(n<=0){

                    printf("Invalid value of n\n");

                    break;

                }

                printf("Enter the elements: ");

                insert(ar,n);

                break;

            case 2:

               n = delete(ar,n);

               break;

            case 3:

                display(ar,n);

                break;

            case 4:

               sort(ar,n);

               break;

            case 5:

                printf("enter element to search");

                scanf("%d",&x);

                search(ar,n,x);

                break;

            default:

               printf("Wrong choice\n");

               break;

        }

    }

}

int main()

{

   process();

   return 0;

}

1. Search for all the occurrences of an element in an integer array (positions)

#include<stdio.h>

int main()

{

    int a[10],i,n,num,p[10];

    int count=0;

    printf("Enter the size of the array: ");  //Takes input

    scanf("%d",&n);

    for(i=0;i<n;i++)

    {

        printf("Enter array elements: ");

        scanf("%d",&a[i]);

    }

    printf("Enter the array elment to find: ");  //Takes the value of element to find

    scanf("%d",&num);

    for(i=0;i<n;i++)

    {

        if(a[i]==num)                           //searches for the element in the array

        {

            p[count]=i;                         //stores the positions at which element is found into another array

            count++;

        }

    }

    printf("Occurrence of %d is: %d Times\n", num, count);  //Prints the no of occurances

    printf("Positions of %d are index: ", num);

    for(i = 0; i < count; i++)

    {

        printf("%d ", p[i]); //print the positions

    }

    return 0;

}

1. Sort the array elements in ascending order (minimum three functions: read, disp and sort)

#include<stdio.h>

#define MAX\_SIZE 100

int arr[MAX\_SIZE],n,i;

void read()

{

    printf("enter array elements:");

    for(i=0;i<n;i++)

    {

    scanf("%d",&arr[i]);

    }

}

void disp()

{

    printf("array elements are:");

    for(i=0;i<n;i++)

    {

    printf(" %d",arr[i]);

    }

}

void sort()

{

    int j,t;

    for(i=0;i<n;i++){

        for(j=0;j<(n-1)-i;j++){

            if(arr[j]>arr[j+1]){

                t=arr[j];

                arr[j]=arr[j+1];

                arr[j+1]=t;

            }

        }

    }

}

void del()

{

    int i;

    if(n == -1){

        printf("ARRAY IS EMPTY");

    }

    else{

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

        int pos;

        scanf("%d", &pos);

        if(pos < 0 || pos >= n){

            printf("Invalid position");

        }

        else{

            printf("Deleted element is %d: ", arr[pos]);

            for(i = pos; i < n - 1; i++){

                arr[i] = arr[i + 1];

            }

            n--;

        }

    }

}

int menu()

{

    int ch;

    printf("\nEnter your choice:\n INSERT-1\n DISPLAY-2\n SORT-3\n DELETE-4\n EXIT-5: ");

    scanf("%d",&ch);

    return ch;

}

void process()

{

    int ch;

    for(ch=menu();ch!=5;ch=menu())

    {

        switch(ch)

        {

            case 1:

                printf("enter the number of elements to enter: ");

                scanf("%d",&n);

                read();

                break;

            case 2:

                disp();

                break;

            case 3:

                sort();

                break;

            case 4:

                del();

                break;

            default:

                printf("wrong choice");

                break;

        }

    }

}

int main()

{

    process();

}

1. Two-dimensional matrix: using functions
   * 1. Addition
     2. Subtraction
     3. Multiplication
     4. Transpose   [Need to discuss before implementing]
     5. Determinant

#include<stdio.h>

int arr1[10][10],arr2[10][10],arr3[10][10],i,j,k;

void read(int r1,int c1,int r2,int c2)

{

    printf("\nEnter the elements of 1st matrix: ");

    for(i=0;i<r1;i++)

    {

     for(j=0;j<c1;j++)

     {

         scanf("%d",&arr1[i][j]);

     }

    }

    printf("\nEnter the elements of 2nd matrix: ");

    for(i=0;i<r2;i++)

    {

     for(j=0;j<c2;j++)

     {

         scanf("%d",&arr2[i][j]);

     }

    }

}

void disp(int r1,int c1,int r2,int c2)

{

    printf("\nThe elements of 1st matrix are: \n");

    for(i=0;i<r1;i++)

    {

     for(j=0;j<c1;j++)

     {

         printf("%d ",arr1[i][j]);

     }

     printf("\n");

    }

    printf("\nThe elements of 2nd matrix are: \n");

    for(i=0;i<r2;i++)

    {

     for(j=0;j<c2;j++)

     {

            printf("%d ",arr2[i][j]);

     }

     printf("\n");

    }

}

void add(int r1,int c1,int r2,int c2)

{

    printf("\nAddition: \n");

    if(r1==r2 && c1==c2)

    {

    for(i=0;i<r1;i++)

    {

     for(j=0;j<c1;j++)

     {

         arr3[i][j]=arr1[i][j]+arr2[i][j];

         printf("%d\t",arr3[i][j]);

     }

     printf("\n");

    }

    }

    else

    {

     printf("For addition matrix should be of same size");

    }

}

void sub(int r1,int c1,int r2,int c2)

{

    printf("\nSubtraction: \n");

    if(r1==r2 && c1==c2)

    {

     for(i=0;i<r1;i++)

     {

         for(j=0;j<c1;j++)

         {

             arr3[i][j]=arr1[i][j]-arr2[i][j];

             printf("%d\t",arr3[i][j]);

         }

         printf("\n");

     }

    }

    else

    {

     printf("For subtraction matrix should be of same size");

    }

}

void mul(int r1,int c1,int r2,int c2)

{

    printf("\nMultiplication: \n");

    if(c1==r2)

    {

     for(i=0;i<r1;i++)

     {

         for(j=0;j<c2;j++)

         {

             arr3[i][j]=0;

             for(k=0;k<c2;k++)

             {

                 arr3[i][j]=arr3[i][j]+ arr1[i][k] \* arr2[k][j];

             }

                 printf("%d\t",arr3[i][j]);

         }

         printf("\n");

     }

    }

    else

    {

     printf("Multiplication not possible the size should be of the form((axb)(bxc)");

    }

}

void tran(int r1,int c1,int r2,int c2)

{

    printf("\nTransponse of 1st matrix: \n");

    for(i=0;i<c1;i++)

    {

     for(j=0;j<r1;j++)

     {

         printf("%d ",arr1[j][i]);

     }

     printf("\n");

    }

    printf("\nTransponse of 2nd matrix: \n");

    for(i=0;i<c2;i++)

    {

     for(j=0;j<r2;j++)

     {

         printf("%d ",arr2[j][i]);

     }

     printf("\n");

    }

}

int menu()

{

    int ch;

    printf("\nChoices:\n1.Display\n2.Addition\n3.Subtraction\n4.Multiplication\n5.Transpose\n6.Exit\nEnter your choice: ");

    scanf("%d",&ch);

    return ch;

}

void process(int r1,int c1,int r2,int c2)

{

    int ch;

    for(ch=menu();ch!=6;ch=menu())

    {

     switch(ch)

     {

         case 1: disp(r1,c1,r2,c2);

                 break;

         case 2: add(r1,c1,r2,c2);

                 break;

         case 3: sub(r1,c1,r2,c2);

                 break;

         case 4: mul(r1,c1,r2,c2);

                 break;

         case 5: tran(r1,c1,r2,c2);

                 break;

         default:printf("Invalid Choice");

                 break;

        }

    }

}

int main()

{

    int r1,c1,r2,c2;

    printf("Enter the row size and column size of the 1st matrix: ");

    scanf("%d %d",&r1,&c1);

    printf("Enter the row size and column size of the 2nd matrix: ");

    scanf("%d %d",&r2,&c2);

    read(r1,c1,r2,c2);

    process(r1,c1,r2,c2);

}

1. Display the array elements in the same order using a recursive function

#include<stdio.h>

#define MAX\_SIZE 100

void print\_array(int arr[], int start, int len)

{

    if(start >= len)

        return;

    printf(" %d",arr[start]);

    print\_array(arr,start+1,len);

}

int main()

{

    int arr[MAX\_SIZE];

    int n,i;

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

    scanf("%d",&n);

    printf("enter elements into the array:");

    for(i=0;i<n;i++)

    {

        scanf("%d",&arr[i]);

    }

    printf("array elements are");

    print\_array(arr,0,n);

    return 0;

}

1. Display array elements in reverse order using a recursive function

#include<stdio.h>

#define MAX\_SIZE 100

void print\_reverse(int arr[],int end)

{

    if(end<0)

        return;

    printf(" %d",arr[end]);

    print\_reverse(arr,end-1);

}

int main()

{

    int arr[MAX\_SIZE],n,i;

    int len,end;

    printf("ENTER THE SIZE OF THE ARRAY:");

    scanf("%d",&n);

    printf("ENTER ARRAY ELEMENTS:");

    for(i=0;i<n;i++)

    {

        scanf("%d",&arr[i]);

    }

    printf("arary elements are: ");

    print\_reverse(arr,n-1);

    return 0;

}

1. Implement stack operations using arrays.

#include <stdio.h>

int a[5], top = -1;

void push(int e) {

    if (top + 1 == 5) {

        printf("Stack is full\n");

    } else {

        a[++top] = e;

    }

}

void pop() {

    if (top == -1) {

        printf("Stack Underflow\n");

    } else {

        printf("Deleted Element %d\n", a[top--]);

    }

}

void peek() {

    if (top == -1) {

        printf("Stack is empty\n");

    } else {

        printf("Element at top is %d\n", a[top]);

    }

}

int menu() {

    int ch;

    printf("\n1. push\n2. pop\n3. Peek\n4. Exit\nEnter your choice: ");

    scanf("%d", &ch);

    return ch;

}

void process() {

    int ch, e;

    for (ch = menu(); ch != 4; ch = menu()) {

        switch (ch) {

            case 1:

                printf("Enter the element: ");

                scanf("%d", &e);

                push(e);

                break;

            case 2:

                pop();

                break;

            case 3:

                peek();

                break;

            default:

                printf("\nInvalid Choice\n");

        }

    }

}

int main() {

    process();

    return 0;

}

1. Reverse a string using Stack

#include<stdio.h>

#include<string.h>

int stack[10],top =-1;

char a[10];

int i;

void pop()

{

    printf("Reversed string is");

    for(i=top;top>-1;top--)

    {

        printf(" %c",stack[top]);

    }

}

void push()

{

    for(i=0;i<strlen(a);i++)

    {

        top++;

        stack[top]=a[i];

    }

}

int main()

{

    printf("Enter the string:");

    gets(a);

    push(a);

    pop();

}

**Session 3: Stack**

1. Convert an expression from infix to postfix using stack

#include<stdio.h>

#include<string.h>

int stack[10],top =-1;

char a[10];

int i;

void pop()

{

    printf("Reversed string is");

    for(i=top;top>-1;top--)

    {

        printf(" %c",stack[top]);

    }

}

void push()

{

    for(i=0;i<strlen(a);i++)

    {

        top++;

        stack[top]=a[i];

    }

}

int main()

{

    printf("Enter the string:");

    gets(a);

    push(a);

    pop();

}

1. Evaluate an expression using stack

**Session 4: Struct**

1. Define a structure for dates with dd/mm/yyyy. Provide functions for reading, displaying and comparing two dates are equal or not

#include<stdio.h>

struct date

{

    int day1,month1,year1;

    int day2,month2,year2;

};

struct date d;

void insert()

{

    printf("Enter the first date in the format dd/mm/yyyy: ");

    scanf("%d/%d/%d",&d.day1,&d.month1,&d.year1);

    printf("Enter the second date in the format dd/mm/yyyy: ");

    scanf(" %d/%d/%d",&d.day2,&d.month2,&d.year2);

}

void display()

{

    printf("First Date: %d/%d/%d\n",d.day1,d.month1,d.year1);

    printf("Second Date: %d/%d/%d\n",d.day2,d.month2,d.year2);

}

void compare()

{

    if(d.day1==d.day2&&d.month1==d.month2&&d.year1==d.year2)

    {

        printf("Both Date Are Same");

    }

    else

    {

        printf("Both Date Are Different");

    }

}

int main()

{

    insert();

    display();

    compare();

}

1. Define a structure for employees with eno,ename, esal and dno. Read  n employees information and provide functions for the following:
   1. Searching an employee by no
   2. Sorting the employees by
      * Name
      * Salary
   3. Deleting an employee

**Session 5: Polynomials using  Array**

1. Read a polynomial and display it; use array

#include<stdio.h>

int coeff[10];

int i;

int read(degree)

{

    printf("ENTER THE ELEMENTS OF THE POLYNOMIAL\n");

    for(i=degree;i>=0;i--)

    {

        if(i==0)

        {

            printf("enter the constant\n");

            scanf("%d",&coeff[i]);

            break;

        }

        printf("enter the coefficient\n");

        scanf("%d",&coeff[i]);

    }

}

int disp(degree)

{

    printf("the entered polynomial is\n");

    for(i=degree;i>=0;i--)

    {

        if(i==0)

        {

            printf("%d",coeff[i]);

            break;

        }

        printf("%dx^%d+",coeff[i],i);

    }

}

int main()

{

    int degree;

    printf("Enter the degree of polynomial");

    scanf("%d",&degree);

    read(degree);

    disp(degree);

    return 0;

}

1. Add two polynomials using the array itself

#include <stdio.h>

int main()

{

    int c1[10], e1[10], c2[10], e2[10], c3[20], e3[20];

    int i, j, n1, n2, k = 0;

    printf("Enter the number of terms for polynomial 1: ");

    scanf("%d", &n1);

    printf("Enter the polynomial 1 terms:\n");

    for (i = 0; i < n1; i++)

    {

        printf("Enter the coefficient: ");

        scanf("%d", &c1[i]);

        printf("Enter the exponent: ");

        scanf("%d", &e1[i]);

    }

    printf("Enter the number of terms for polynomial 2: ");

    scanf("%d", &n2);

    printf("Enter the polynomial 2 terms:\n");

    for (i = 0; i < n2; i++)

    {

        printf("Enter the coefficient: ");

        scanf("%d", &c2[i]);

        printf("Enter the exponent: ");

        scanf("%d", &e2[i]);

    }

    i = j = 0;

    while (i < n1 && j < n2)

    {

        if (e1[i] == e2[j])

        {

            c3[k] = c1[i] + c2[j];

            e3[k] = e1[i];

            i++;

            j++;

            k++;

        }

        else if (e1[i] > e2[j])

        {

            c3[k] = c1[i];

            e3[k] = e1[i];

            i++;

            k++;

        }

        else

        {

            c3[k] = c2[j];

            e3[k] = e2[j];

            j++;

            k++;

        }

    }

    while (i < n1)

    {

        c3[k] = c1[i];

        e3[k] = e1[i];

        i++;

        k++;

    }

    while (j < n2)

    {

        c3[k] = c2[j];

        e3[k] = e2[j];

        j++;

        k++;

    }

    printf("Resultant polynomial after addition:\n");

    for (i = 0; i < k; i++)

    {

        printf("%dx^%d", c3[i], e3[i]);

        if (i < k - 1)

            printf(" + ");

    }

    return 0;

}

**Session 6: Polynomials using Structure**

1. Read a polynomial and display it; use structure array

#include<stdio.h>

struct poly{

    int coeff;

    int exp;

};

int main()

{

    int i,num;

    struct poly p[10];

    printf("enter the no of terms \n");

    scanf("%d",&num);

    for(i=0;i<num;i++)

    {

        printf("enter the coeff\n");

        scanf("%d",&p[i].coeff);

        printf("enter the degree\n");

        scanf("%d",&p[i].exp);

    }

    printf("the entered polynomial is :");

    for(i=0;i<num;i++)

    {

        printf("%dx^%d+",p[i].coeff,p[i].exp);

    }

    return 0;

}

1. Add two polynomials

#include <stdio.h>

struct poly {

    int coeff;

    int exp;

};

int main() {

    struct poly p1[10], p2[10], p3[20]; // p3 should have enough space for the sum

    int i, j, k, n1, n2;

    printf("Enter the number of terms for the first polynomial: ");

    scanf("%d", &n1);

    for (i = 0; i < n1; i++) {

        printf("Enter the coefficient: ");

        scanf("%d", &p1[i].coeff);

        printf("Enter the degree: ");

        scanf("%d", &p1[i].exp);

    }

    printf("Enter the number of terms for the second polynomial: ");

    scanf("%d", &n2);

    for (i = 0; i < n2; i++) {

        printf("Enter the coefficient: ");

        scanf("%d", &p2[i].coeff);

        printf("Enter the degree: ");

        scanf("%d", &p2[i].exp);

    }

    printf("The first polynomial is: ");

    for (i = 0; i < n1; i++) {

        printf("%dx^%d", p1[i].coeff, p1[i].exp);

        if (i < n1 - 1) {

            printf(" + ");

        }

    }

    printf("\nThe second polynomial is: ");

    for (i = 0; i < n2; i++) {

        printf("%dx^%d", p2[i].coeff, p2[i].exp);

        if (i < n2 - 1) {

            printf(" + ");

        }

    }

    i = j = k = 0;

    while (i < n1 && j < n2) {

        if (p1[i].exp == p2[j].exp) {

            p3[k].coeff = p1[i].coeff + p2[j].coeff;

            p3[k].exp = p1[i].exp;

            i++;

            j++;

            k++;

        } else if (p1[i].exp < p2[j].exp) {

            p3[k].coeff = p1[i].coeff;

            p3[k].exp = p1[i].exp;

            i++;

            k++;

        } else {

            p3[k].coeff = p2[j].coeff;

            p3[k].exp = p2[j].exp;

            j++;

            k++;

        }

    }

    while (i < n1) {

        p3[k].coeff = p1[i].coeff;

        p3[k].exp = p1[i].exp;

        i++;

        k++;

    }

    while (j < n2) {

        p3[k].coeff = p2[j].coeff;

        p3[k].exp = p2[j].exp;

        j++;

        k++;

    }

    printf("\nThe sum of the two polynomials is: ");

    for (i = 0; i < k; i++) {

        printf("%dx^%d", p3[i].coeff, p3[i].exp);

        if (i < k - 1) {

            printf(" + ");

        }

    }

    printf("\n");

    return 0;

}

1. Subtract two polynomials

#include <stdio.h>

struct poly {

    int coeff;

    int exp;

};

int main() {

    struct poly p1[10], p2[10], p3[20]; // p3 should have enough space for the sum

    int i, j, k, n1, n2;

    printf("Enter the number of terms for the first polynomial: ");

    scanf("%d", &n1);

    for (i = 0; i < n1; i++) {

        printf("Enter the coefficient: ");

        scanf("%d", &p1[i].coeff);

        printf("Enter the degree: ");

        scanf("%d", &p1[i].exp);

    }

    printf("Enter the number of terms for the second polynomial: ");

    scanf("%d", &n2);

    for (i = 0; i < n2; i++) {

        printf("Enter the coefficient: ");

        scanf("%d", &p2[i].coeff);

        printf("Enter the degree: ");

        scanf("%d", &p2[i].exp);

    }

    printf("The first polynomial is: ");

    for (i = 0; i < n1; i++) {

        printf("%dx^%d", p1[i].coeff, p1[i].exp);

        if (i < n1 - 1) {

            printf(" - ");

        }

    }

    printf("\nThe second polynomial is: ");

    for (i = 0; i < n2; i++) {

        printf("%dx^%d", p2[i].coeff, p2[i].exp);

        if (i < n2 - 1) {

            printf(" - ");

        }

    }

    i = j = k = 0;

    while (i < n1 && j < n2) {

        if (p1[i].exp == p2[j].exp) {

            p3[k].coeff = p1[i].coeff - p2[j].coeff;

            p3[k].exp = p1[i].exp;

            i++;

            j++;

            k++;

        } else if (p1[i].exp < p2[j].exp) {

            p3[k].coeff = p1[i].coeff;

            p3[k].exp = p1[i].exp;

            i++;

            k++;

        } else {

            p3[k].coeff = p2[j].coeff;

            p3[k].exp = p2[j].exp;

            j++;

            k++;

        }

    }

    while (i < n1) {

        p3[k].coeff = p1[i].coeff;

        p3[k].exp = p1[i].exp;

        i++;

        k++;

    }

    while (j < n2) {

        p3[k].coeff = p2[j].coeff;

        p3[k].exp = p2[j].exp;

        j++;

        k++;

    }

    printf("\nThe sum of the two polynomials is: ");

    for (i = 0; i < k; i++) {

        printf("%dx^%d", p3[i].coeff, p3[i].exp);

        if (i < k - 1) {

            printf(" - ");

        }

    }

    printf("\n");

    return 0;

}

1. Multiply two polynomials