\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*All Programs\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#include<string.h>

//.swapping using bitwise operator

void bitwise\_swap(){

int a=1;

int b=2;

printf("%d %d\n",a,b);

a=a^b; //01+10=11=3

b=a^b; //11+10=01=1

a=a^b; //011+001=010

printf("a = %d b = %d\n",a,b);

}

void add(int \*p,int \*q){

int sum=\*p+\*q;

printf("The sum of a + b is = %d \n",sum);

}

void swap(int \*p,int \*q){

int temp=\*p;

\*p=\*q;

\*q=temp;

}

// 1. Write a C program to check Least Significant Bit (LSB) of a number is set or not.

void LSB(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

if (n & 1) {

printf("The least significant bit is set (LSB = 1).\n");

} else {

printf("The least significant bit is not set (LSB = 0).\n");

}

}

// 2. Write a C program to check Most Significant Bit (MSB) of a number is set or not.

void MSB(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int bits=sizeof(int)\*8;

int msb=1<<(bits-1);

if(n&msb){

printf("MSB is set to 1\n");

}

else{

printf("MSB is Set to 0\n");

}

}

// 3. Write a C program to get nth bit of a number.

void NthBit(){

int num;

printf("Enter the numeber : ");

scanf("%d",&num);

int nthNumber;

printf("Enter the nth position : ");

scanf("%d",&nthNumber);

if((num>>nthNumber)&1){

printf("number %d in this the position of %d bit is 1 \n",num,nthNumber);

}

else{

printf("number %d in this the position of %d bit is 0 \n",num,nthNumber);

}

}

//4. Write a C program to set nth bit of a number.

void setNthBit(){

int num;

printf("Enter the number : ");

scanf("%d",&num);

int nthBitPosition;

printf("Enter the position : ");

scanf("%d",&nthBitPosition);

int setNthBit=(1<<nthBitPosition)|num;

printf("Before setting the nth bit : ;%d\n",num);

printf("After setting the nth bit%d\n",setNthBit);

}

// 5. Write a C program to clear nth bit of a number

void clearNthBit(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int nthBitPosition;

printf("Enter the position to clear : ");

scanf("%d",&nthBitPosition);

int newClearNthBit=(1<<nthBitPosition)^n;

printf("Before cleaning the nth Bit : %d\n",n);

printf("After cleaning the nth Bit : %d\n",newClearNthBit);

}

void toggleBit(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos;

printf("Enter the pos : ");

scanf("%d",&pos);

// pos=pos-1;

int bit=((1<<pos-1)^n); // 1 0 1 0

// 0 1 0 0

printf("The new value is : %d\n",bit);

}

void highBit(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos=0;

for(int i=32;i>=0;i--){

if((n>>i)&1){

pos=i;

break;

}

}

printf("The High index of 1 bit is : %d",pos+1);

}

void highBit1(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos=0;

for(int i=32;i>=0;i--){

if((1<<i)&n){

pos=i;

break;

}

}

printf("The high index of 1 bit is : %d",pos+1);

}

void lowBit2(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos=0;

for(int i=32;i>=0;i--){

if((1<<i)&n){

pos=i;

}

}

printf("The low index of 1 bit is : %d",pos+1);

}

void lowBit13(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos=0;

for(int i=0;i<=32;i++){

if((n>>i)&n){

pos=i;

break;

}

}

printf("The low index of 1 bit is : %d",pos+1);

}

void lowBit1(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos=0;

for(int i=0;i<=32;i++){

if((n>>i)&1){

pos=i;

break;

}

}

printf("The low index of 1 bit is : %d",pos+1);

}

////////////////////////////////////////Pointer programs://////////////////////////////////////////////

// 2.Write a C program to add two numbers using pointers.

void add\_using\_Pointer(int \*a,int \*b){

int sum=\*a+\*b;

printf("The sum of 2 number is : %d\n",sum);

}

// 4. Write a C program to input and print array elements using pointer.

void array(){

int n;

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

scanf("%d",&n);

int arr[n];

int \*ptr=arr;

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

printf("Enter the %d element : ",i+1);

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

}

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

printf("%d ",\*ptr);

ptr++;

}

}

// 5. Write a C program to copy one array to another using pointers.

void array1(){

int n;

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

scanf("%d",&n);

int arr[n];

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

printf("Enter the %d elements : ",i+1);

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

}

int arr1[n];

int \*ptr=arr1;

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

ptr[i]=arr[i];

}

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

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

}

}

// 6. Write a C program to swap two arrays using pointers.

void array3(){

int n;

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

scanf("%d",&n);

int arr[n];

int \*ptr=arr;

int arr2[n];

int \*ptr2=arr2;

printf("Enter the 1 array elements \n");

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

printf("Enter the %d elements : ",i+1);

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

}

printf("Enter the 2 array elements \n");

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

printf("Enter the %d elements : ",i+1);

scanf("%d",&ptr2[i]);

}

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

int temp=\*(ptr+i);

\*(ptr+i)=\*(ptr2+i);

\*(ptr2+i)=temp;

}

printf("after swaping the array 1 list is : \n");

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

printf("%d ",\*(ptr+i));

}

printf("\n");

printf("after swaping the array 2 list is : \n");

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

printf("%d ",\*(ptr2+i));

}

}

// 7. Write a C program to reverse an array using pointers.

void array4(){

int n;

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

scanf("%d",&n);

int arr[n];

int \*ptr=arr;

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

printf("Enter the %d elements : ",i+1);

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

}

for(int i=n-1;i>=0;i--){

printf("%d ",\*(ptr+i));

}

}

// 8. Write a C program to search an element in array using pointers.

int array5(){

int n;

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

scanf("%d",&n);

int \*arr=(int\*)malloc(n\*sizeof(int));

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

printf("Enter the %d elements : ",i+1);

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

}

int key;

printf("Enter the key elements : ");

scanf("%d",&key);

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

if(\*(arr+i)==key){ // if(arr[i]==key){

printf("%d Key found postion %d\n",key,i+1);

return 0;

}

}

printf("Not Found\n");

return 0;

}

// 9. Write a C program to access two dimensional array using pointers.

void array6(){

int row,col;

printf("Enter the row and column size : ");

scanf("%d%d",&row,&col);

int arr[row][col];

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

printf("Enter the elemets for %d row \n",i+1);

for(int j=0;j<col;j++){

printf("Enter the %d elements : ",j+1);

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

}

}

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

printf("Here %d row elements : \n",i+1);

for(int j=0;j<col;j++){

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

}

printf("\n");

}

}

// same program as above but difference is using pointer

void array7(){

int row,col;

printf("Enter the row and column size : ");

scanf("%d%d",&row,&col);

// for 2D array we need double pointer

int \*\*arr=(int\*\*)malloc(row\*sizeof(int\*));

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

arr[i]=(int\*)malloc(col\*sizeof(int));

}

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

printf("Enter the elemets for %d row \n",i+1);

for(int j=0;j<col;j++){

printf("Enter the %d elements : ",j+1);

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

}

}

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

printf("Here %d row elements : \n",i+1);

for(int j=0;j<col;j++){

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

}

printf("\n");

}

}

// 10. Write a C program to add two matrix using pointers.

void array8(){

int row,col;

printf("Enter the size of rows and cols : ");

scanf("%d%d",&row,&col);

// First 2D arrays

int \*\*arr1=(int\*\*)malloc(row\*sizeof(int\*));

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

arr1[i]=(int\*)malloc(row\*sizeof(int));

}

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

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

printf("Enter the %d row elements\n",i+1);

for(int j=0;j<col;j++){

printf("Enter the %d element : ",j+1);

scanf("%d",&arr1[i][j]);

}

}

// second 2D arrays

int \*\*arr2=(int\*\*)malloc(row\*sizeof(int\*));

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

arr2[i]=(int\*)malloc(row\*sizeof(int));

}

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

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

printf("Enter the %d rows elements\n",i+1);

for(int j=0;j<col;j++){

printf("Enter the %d element : ",j+1);

scanf("%d",&arr2[i][j]);

}

}

int \*\*arr3=(int\*\*)malloc(row\*sizeof(int\*));

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

arr3[i]=(int\*)malloc(row\*sizeof(int));

}

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

for(int j=0;j<col;j++){

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

}

}

printf("\n");

// printing the array

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

printf("the %d row additions is \n",i+1);

for(int j=0;j<col;j++){

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

}

printf("\n");

}

}

// 11. Write a C program to multiply two matrix using pointers.

void array9(){

int row,col;

printf("Enter the size of rows and columns : ");

scanf("%d%d",&row,&col);

int \*\*arr1=(int\*\*)malloc(row\*sizeof(int\*));

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

arr1[i]=(int\*)malloc(col\*sizeof(int));

}

printf("Arra1 elements : \n");

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

printf("Enter the %d row elements : \n",i+1);

for(int j=0;j<col;j++){

printf("Enter the %d elemet : ",j+1);

scanf("%d",&arr1[i][j]);

}

}

int \*\*arr2=(int\*\*)malloc(row\*sizeof(int\*));

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

arr2[i]=(int\*)malloc(col\*sizeof(int));

}

printf("\nArray2 elements : \n");

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

printf("Enter the %d row elements : \n",i+1);

for(int j=0;j<col;j++){

printf("Enter the %d element : ",j+1);

scanf("%d",&arr2[i][j]);

}

}

// 3 array for storing the multiplayig of 2 array array and storing in the 3rd array

int \*\*arr3=(int\*\*)malloc(row\*sizeof(int\*));

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

arr3[i]=(int\*)malloc(sizeof(int));

}

// multiplying and storing the values

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

for(int j=0;j<col;j++){

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

}

}

// printing the multiplyed elemets

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

printf("here is the %d row elemets : \n",i+1);

for(int j=0;j<col;j++){

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

}

printf("\n");

}

}

// 12. Write a C program to find length of string using pointers.

void len\_string(){

char str[100];

int count=0;

printf("Enter the string : ");

scanf("%s",str);

char \*str1=str;

while(\*str1!='\0'){

count++;

str1++;

}

printf("The string length is : %d\n",count);

}

// 13. Write a C program to copy one string to another using pointers.

void string\_copy(){

char str1[100];

char str2[100];

printf("Enter the string 1 : ");

scanf("%s",str1);

char \*ptr=str1;

int i=0;

while(\*ptr!='\0'){

str2[i]=\*ptr;

i++;

ptr++;

}

printf("The copy string of string 2 is : %s\n",str2);

}

// same as privous program

void string\_copy2(){

char \*str1=(char\*)malloc(100\*sizeof(char));

if(str1==NULL){

printf("Memory is not allocated \n");

}

printf("Enter the string 1 : ");

scanf("%s",str1);

char \*str2=(char\*)malloc(100\*sizeof(char));

if(str2==NULL){

printf("Memory is not allocated \n");

}

int i=0;

while(\*str1!='\0'){

\*(str2+i)=\*str1;

i++;;

str1++;

}

printf("The copy string of string 2 is : %s\n",str2);

free(str1);

free(str2);

}

// 14. Write a C program to concatenate two strings using pointers.

void concatenate\_str(){

// one string is created

char \*str1=(char\*)malloc(100\*sizeof(char));

if(str1==NULL){

printf("Memory is not allocated \n");

}

else{

printf("Enter the string1 : ");

scanf("%s",str1);

}

// another string is created

char \*str2=(char\*)malloc(100\*sizeof(char));

if(str2==NULL){

printf("Memory is not allocated \n");

}

else{

printf("Enter the string2 : ");

scanf("%s",str2);

}

// we are storing the str2 in str 1 so we have we have to count how many character in given string 1

int count=0;

char \*ptr=str1;

while(\*ptr!='\0'){

ptr++;

count++;

}

printf("The count of string 1 is : %d\n",count);

// concatinating str2 in str1

int i = 0;

while (str2[i] != '\0') {

str1[count + i] = str2[i];

i++;

}

str1[count + i] = '\0';

printf("Concatinating of 2 string is : %s\n",str1);

}

// 15. Write a C program to compare two strings using pointers.

void cmp\_str(){

char \*str1=(char\*)malloc(100\*sizeof(char));

if(str1==NULL){

printf("Memory is not allocated\n");

}

else{

printf("Enter the string 1 : ");

scanf("%s",str1);

}

char \*str2=(char\*)malloc(100\*sizeof(char));

if(str2==NULL){

printf("Memory is not allocated\n");

}

else{

printf("Enter the string 1 : ");

scanf("%s",str2);

}

int count=0;

int count1=0;

int count2=0;

char \*ptr=str1;

while(\*ptr!='\0'){

count++;

ptr++;

}

char \*ptr2=str2;

while(\*ptr2!='\0'){

count1++;

ptr2++;

}

if(count==count1){

while(\*str1!='\0' && \*str2!='\0'){

if(\*str1==\*str2){

count2++;

}

str1++;

str2++;

}

}

printf("%d %d %d\n",count,count1,count2);

if(count==count2){

printf("Both strings are same\n");

}

else{

printf("Both string are not same\n");

}

}

void cmp\_str1(){

// string 1

char \*str1=(char\*)malloc(100\*sizeof(char));

if(str1==NULL){

printf("Memory is not allocated \n");

}

else{

printf("Enter the string 1 : ");

scanf("%s",str1);

}

// string 2

char \*str2=(char\*)malloc(100\*sizeof(char));

if(str2==NULL){

printf("Memory is not allocated \n");

}

else{

printf("Enter the string 1 : ");

scanf("%s",str2);

}

int count=0;

int count1=0;

int count2=0;

// string 1 count

char \*ptr=str1;

while(\*ptr!='\0'){

count++;

ptr++;

}

// string 2 count

char \*ptr2=str2;

while(\*ptr2!='\0'){

count1++;

ptr2++;

}

printf("%d %d\n",count,count1);

if(count==count1){

while(\*str1!='\0' && \*str2!='\0'){

if(\*str1==\*str2){

printf("%c %c\n",\*str1,\*str2);

count2++;

}

str1++;

str2++;

}

}

printf("%d\n",count2);

if(count==count2){

printf("Both string are same \n");

}

else{

printf("Both string are not same \n");

}

}

// 16. Write a C program to find reverse of a string using pointers.

void rev\_str(){

char \*str=(char\*)malloc(100\*sizeof(char));

if(str==NULL){

printf("Memory is not allocated\n");

}

else{

printf("Enter the string : ");

scanf("%s",str);

}

char \*ptr=str;

int count=0;

while(\*ptr!='\0'){

count++;

ptr++;

}

printf("%d\n",count);

for(int i=0;i<(count-1)/2;i++){

char temp=\*(str+i);

\*(str+i)=\*(str+count-i-1);

\*(str+count-i-1)=temp;

}

printf("%s",str);

}

void rev\_str2(){

char \*str=(char\*)malloc(100\*sizeof(char));

if(str==NULL){

printf("Memory is not allocated\n");

}

else{

printf("Enter the string : ");

scanf("%s",str);

}

int count=0;

char \*ptr=str;

while(\*ptr!='\0'){

count++;

ptr++;

}

int start=0;

int end=count-1;

while(start<end){

char temp=\*(str+start);

\*(str+start)=\*(str+end);

\*(str+end)=temp;

start++;

end--;

}

printf("The reverse string is : %s\n",str);

}

// 17. Write a C program to sort array using pointers.

void sort\_array(){

int n;

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

scanf("%d",&n);

int \*arr=(int\*)malloc(n\*sizeof(int));

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

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

printf("Enter the %d element : ",i+1);

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

}

// sorting the array

for(int i=1;i<n;i++){

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

if(\*(arr+j)>\*(arr+j+1)){

int temp=\*(arr+j);

\*(arr+j)=\*(arr+j+1);

\*(arr+j+1)=temp;

}

}

}

// printing the array

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

printf("%d ",\*(arr+i));

}

}

// 18. Write a C program to return multiple value from function using pointers.

void multiple\_values(int \*a,int \*b,int \*c){

\*a=100;

\*b=200;

\*c=300;

}

void multiple(){

int n,n2,n3;

multiple\_values(&n,&n2,&n3);

printf("The n value is : %d\n",n);

printf("The n2 value is : %d\n",n2);

printf("The n3 value is : %d\n",n3);

}

// String Programs:\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// 1. Write a C program to find length of a string.

void str1(){

char str[100];

printf("Enter the string : ");

scanf("%s",str);

int count=0;

int i=0;

while(str[i]!='\0'){

count++;

i++;

}

printf("The string length is : %d\n",count);

}

// write a c program to find the length of a string.

void strPolindrome(){

char str[50];

printf("Enter the string : ");

scanf("%s",str);

int len=0;

int i=0;

bool flag=0;

while(\*(str+i)!='\0'){

len++;

i++;

}

printf("string length is : %d\n",len);

int n=0;

while(n<len/2){

if(\*(str+n)==\*(str+len-1-n)){ // 0 4 // 1 5-1-1 3// 2 5-1-22

flag=1;

printf("\*\n");

}

else{

flag=0;

break;

}

n++;

}

if(flag==1){

printf("Polindrome");

}

else{

printf("Not");

}

}

// 1. Write a C program to print all natural numbers between 1 to n using recursion.

void print(int start,int n){

if(start<=n){

printf("%d ",start);

print(start+1,n);

}

}

void naturalNumber(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

if(n<0){

printf("Please write the positive number \n");

}

else{

printf("The natural is 1 to %d is : ",n);

print(1,n);

printf("\n");

}

}

// 2. Write a C program to print all even or odd numbers in given range using recursion.

#include <stdio.h>

void printEvenOddNumber(int start,int end,int choice){

if(start>end){

return ;

}

else if((choice==1 && start%2==0) || (choice==2 && start%2!=0)) {

printf("%d ",start);

}

printEvenOddNumber(start+1,end,choice);

}

void EvenOdd(){

int start,end,choice;

printf("Enter the starting range : ");

scanf("%d",&start);

printf("Enter the ending range : ");

scanf("%d",&end);

printf("Enter\n1 for Even\n2 for Odd\n");

scanf("%d",&choice);

if(choice==1){

printf("Even number from 0 to %d \n",start);

}

else if(choice==2){

printf("Odd number from 0 to %d \n",start);

}

else {

printf("Please enter the correct choice \n");

}

printEvenOddNumber(start,end,choice);

}

// 3. Write a C program to find sum of all natural numbers between 1 to n using recursion.

int naturalSum(int start,int end){

int sum=0;

if(start>end){

return 0;

}

else {

return start+naturalSum(start+1,end);

}

}

void sumNatural(){

int n;

printf("Enter the natural number : ");

scanf("%d",&n);

if(n<0){

printf("Please enter the positive number \n");

}

else if(n>0){

printf("some of natural number is : \n");

}

int sum=naturalSum(0,n);

printf("Some of Natural number is : %d\n",sum);

}

// 4. Write a C program to find sum of all even or odd numbers in given range using recursion.

int printSumEvenOdd(int start,int end,int choice){

int sum=0;

if(start>end){

return 0;

}

else if((choice==1 && start%2==0) ||(choice==2 && start%2!=0)){

sum=start;

}

return sum+printSumEvenOdd(start+1,end,choice);

}

void sumEvenOdd(){

int start,end,choice;

printf("Enter the starting range : ");

scanf("%d",&start);

printf("Enter the ending range : ");

scanf("%d",&end);

printf("Enter\n1 for EvenSum\n2 for OddSum");

scanf("%d",&choice);

if(choice==1){

printf("The sum of even number from %d to %d = ",start,end);

}

else if(choice==2){

printf("The sum of odd number from %d to %d = ",start,end);

}

else{

printf("Wrong input");

}

int sum=printSumEvenOdd(start,end,choice);

printf("%d",sum);

}

// number to binary;

void convertBinary(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int bit[32];

int i=0;

while(n>0){

int rem=n%2;

bit[i]=rem;

n/=2;

i++;

}

for(int j=i-1;j>=0;j--){

printf("%d ",bit[j]);

}

}

// convert string to decimal

void strToDecimal(){

char str[10];

printf("Enter the string : ");

scanf("%s",str);

int size=strlen(str);

int decimal=0;

int bit=0;

for(int i=size-1;str[i]!='\0';i--){

if(str[i]=='1'){

decimal+=1<<bit;

}

bit++;

}

printf("%d\n",decimal);

}

// 2. Write a C program to copy one string to another string.

void strCopy(){

char str[100];

printf("Enter the string : ");

scanf("%s",str);

char str2[100];

int i=0;

while(str[i]!='\0'){

str2[i]=str[i];

i++;

}

printf("The string2 is : %s\n",str2);

}

// 3. Write a C program to concatenate two strings.

void strCon(){

char str1[100];

char str2[100];

printf("Etner the string 1 : ");

scanf("%s",str1);

printf("Enter the string 2 : ");

scanf("%s",str2);

int len=0;

while(str1[len]!='\0'){

len++;

}

printf("The length of string 1 is : %d\n",len);

int i=0;

while(str2[i]!='\0'){

str1[len+i]=str2[i];

i++;

}

printf("The concatination of string is : %s\n",str1);

}

// 4. Write a C program to compare two strings.

void strCmp(){

char str1[100];

printf("Enter the string 1 : ");

scanf("%s",str1);

char str2[100];

printf("Enter the string 2 : ");

scanf("%s",str2);

int i=0;

bool flag=1;

while(str1[i]!='\0'){

if(str1[i]==str2[i]){

flag=0;

}

else{

flag=1;

break;

}

i++;

}

if(flag==0){

printf("String is Same \n");

}

else{

printf("String is not Same \n");

}

}

// 5. Write a C program to find total number of alphabets, digits or special character in a string.

void countAlph(){

char str[100];

printf("Enter the string : ");

scanf("%s",str); // altafalinadaf@gmail.com

printf("The string is : %s\n",str);

int vowelCount=0;

int conCount=0;

int numCount=0;

int sepCount=0;

for(int i=0;str[i]!='\0';i++){

char ch=str[i];

if(ch=='a' || ch=='e' || ch=='i' || ch=='o' || ch=='u'){

vowelCount++;

}

else if(ch>='a' && ch<='z' || ch>='A' && ch<='Z'){

conCount++;

}

else if(ch>='0' && ch<='9'){

numCount++;

}

else{

sepCount++;

}

}

printf("Total vowel char count : %d\n",vowelCount);

printf("Total con char count : %d\n",conCount);

printf("Total num count : %d\n",numCount);

printf("Total spe char count : %d\n",sepCount);

}

// 6. Write a C program to count total number of vowels and consonants in a string.

void vowConCount(){

char str[100];

printf("Enter the string : ");

scanf("%s",str);

int vowCount=0;

int conCount=0;

char str2[]="aeiou";

for(int i=0;str[i]!='\0';i++){

for(int j=0;str2[j]!='\0';j++){

if(str[i]==str2[j]){

vowCount++;

}

}

if( (str[i]>='a' && str[i]<='z') || (str[i]>='A' && str[i]<='Z')){

conCount++;

}

}

printf("Total vowel is : %d\n",vowCount);

printf("Total con is : %d\n",conCount-vowCount);

}

// 7. Write a C program to count total number of words in a string.

void countWords(){

char str[1000];

printf("Enter the string : ");

gets(str);

printf("%s\n",str);

int i=0,c=0;

for(int i=0;str[i]!='\0';i++){

if(str[i]==' '){

c++;

}

}

printf("The word count is : %d\n",c+1);

}

int show(){

char str[100];//try to use dynamic memory allocation

printf("Enter the string : ");

scanf("%s",str);

int size=strlen(str); //dont use inbuilt functions

printf("size of string is : %d\n",size);

if(size!=17){

printf("Wrong Input ");

return 0;

}

else{

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

char ch=str[i];

if((i == 2) || (i == 5) || (i == 8) || (i == 11) || (i == 14)){ // false

if(!(str[i]==':')){

printf("Wrong Input \n");

return 0;

}

}

else{

for(int i=0;i<size;i+=3){ // 43:54:54:53

for(int j=i;j<i+2;j++){

char ch=str[j];

if((ch>='A' && ch<='F') || (ch>='a' && ch<='f')||(ch>='0' && ch<='9')){ // false

// printf("%c",ch);

}

else{

printf("Wrong input ");

return 0;

}

}

}

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

char ch=str[i];

if(!((ch>='A' && ch<='F') || (ch>='a' && ch<='f')||(ch>='0' && ch<='9') || ch==':')){

printf("Wrong input \n");

return 0;

}

}

}

}

}

printf("Correct input : %s\n",str);

return 0;

}

//8. Write a C program to find reverse of a string.

void revStr(){

char str[100];

printf("Enter the string : ");

scanf("%s",str);

int n=strlen(str);

int i=0;

while(i<n/2){

char ch=str[i];

str[i]=str[n-i-1];

str[n-i-1 ]=ch;

i++;

}

printf("The reverse string is : %s",str);

}

// 9. Write a C program to check whether a string is palindrome or not.

// 10.character count

void charCount(){

char str[100];

printf("Enter the string : ");

scanf("%s",str);

for(int i=0;str[i]!='\0';i++){

int count=0;

bool flag=0;

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

if(str[i]==str[j]){

flag=1;

break;

}

}

if(flag==1){

continue;

}

for(int j=0;str[j]!='\0';j++){

if(str[i]==str[j]){

count++;

}

}

printf("%c %d\n",str[i],count);

}

}

int main(){

// Pointer All Programs:

int a=10;

int b=20;

// swap(&a,&b);

// printf("%d %d\n",a,b);

// add(&a,&b);

// bitwise\_swap();

// LSB();

// MSB();

// NthBit();

// setNthBit();

// clearNthBit();

// toggleBit();

// highBit();

// lowBit1();

// add\_using\_Pointer(&a,&b);

// array();

// array1();

// array3();

// array4();

// array5();

// array6();

// array7();

// array8();

// array9();

// len\_string();

// string\_copy();

// string\_copy2();

// concatenate\_str();

// cmp\_str();

// cmp\_str1();

// rev\_str();

// rev\_str2();

// sort\_array();

// multiple();

// String All Programs:

// str1();

// strPolindrome();

// strCopy();

// strCon();

// strCmp();

// countAlph();

// vowConCount();

// countWords();

// show();

// revStr();

charCount();

// naturalNumber();

// EvenOdd();

// sumNatural();

// sumEvenOdd();

// reverseNumber();

// convertBinary();

// strToDecimal();

return 0;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ALL POINTERS PROGRAMS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include<stdio.h>

#include<stdlib.h>

void null\_pointer(){

printf("This is example of null pointer \n");

int \*ptr1=NULL; // null pointer

printf("There is no output because pointer is null\n");

printf("to accessing the null value : ",\*ptr1); // we will not get any output because we cannot dereference the null pointer

}

void void\_pointer(){

int a=100;

void \*ptr=&a;

printf("This is the example of void pointer \n");

// X printf("\*p holding the variable address value is : %d\n",\*ptr); // we cannot accessing the void pointer value without type casting

printf("\*p holding the variable address value is : %d\n",\*(int\*)ptr); // type casting void to int

}

void pointer(){

int a=100;

int \*p=&a;

printf("Single pointer \n");

printf("\*p holding the variable address value is : %d\n",\*p);

}

void double\_pointers(){

int a=10;

int \*p=&a;

int\* \*q=&p;

printf("Double pointer using : \n");

printf("\*p addressing variable value is : %d\n",\*p);

printf("\*q addressing variavle value is : %d\n",\*\*q);

printf("\n\n");

}

void three\_pointers(){

int a=10;

int \*p=&a;

int\* \*q=&p;

int\*\* \*r=&q;

printf("Three pointer using : \n");

printf("\*p addressing variable value is : %d\n",\*p);

printf("\*q addressing variavle value is : %d\n",\*\*q);

printf("\*r addressing variavle value is : %d\n",\*\*\*r);

printf("\n\n");

}

void pointer\_addition(){

int arr[]={2,3,4,5,6,7};

int \*p=arr; //pointing the base address

printf("This is the example of adding pointer or addition of pointer : \n");

printf("the p index addrees value is : %d\n",\*p);

p=p+2; // adding to pointer position // first it pointing base address after it increment to 2 position

printf("the p+2 index addrees value is : %d\n",\*p);

p+=2;

printf("the p+2+2 index addrees value is : %d\n",\*p);

printf("\n\n");

}

void pointer\_substraction(){

int arr[]={2,3,4,5,6,7};

int \*p=&arr[5]; //pointing the 5th index address

printf("This is the example of substracting pointer or substraction of pointer : \n");

printf("the p index addrees value is : %d\n",\*p);

p=p-2; // adding to pointer position // first it pointing 5th address after it decrement to 2 position

printf("the p-2 index addrees value is : %d\n",\*p);

p+=2; // decrement 2

printf("the p-2-2 index addrees value is : %d\n",\*p);\

printf("\n\n");

}

void dangling\_pointer(){

int \*ptr;

printf("This is the example of dangling pointer \n");

printf("declaring the pointer without initializing and accessing the value is callled dangling pointer \n");

printf("%d\n",\*ptr);

}

int main(){

printf("1.void\_pointer()\n2.null\_pointer() \n3.pointer()\n4.double\_pointers()\n5.three\_pointers()\n6.pointer\_addition()\n7.pointer\_substraction()\n8.dangling\_pointer()\n");

int input;

printf("Enter the any one : ");

scanf("%d",&input);

switch(input){

case 1:

void\_pointer();

break;

case 2:

null\_pointer();

break;

case 3:

pointer();

break;

case 4:

double\_pointers();

break;

case 5:

three\_pointers();

break;

case 6:

pointer\_addition();

break;

case 7:

pointer\_substraction();

break;

case 8:

dangling\_pointer();

break;

default:

printf("Invalid input \n");

}

null\_pointer();

return 0;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ALL BITWISE PROGRAMS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include<stdio.h>

#include<stdlib.h>

#define Bit 1

#define Bit2(n,Bit) n&Bit

#define MSB(bits,Bit) 1<<(bits-Bit)

#define NTHBIT(num,nthNumber) num>>nthNumber

#define SetBit(Bit,nthBitPosition,num) (Bit<<nthBitPosition)|num

#define ClearBit(Bit,nthBitPosition,n) (Bit<<nthBitPosition)^n

#define ToggleBit(bit,pos,Bit,n) (Bit<<pos-Bit)^n

#define HighBit(n,i,Bit) (n>>i)&Bit

#define HighBit1(Bit,i,n) Bit<<i&n

#define LowBit(Bit,i,n) (Bit<<i)&n

#define LowBit13(i,Bit,n) (i<<Bit)&n

#define LowBit1(n,i,Bit) (n>>i)&Bit

#define TrailingZero(Bit,i,n) (1<<i)&n

#define LeadingZero(Bit,i,n) (1<<i)&n

#define Flip(n) ~(n)

#define CountBit(Bit,i,n) (1<<i)&n

//.swapping using bitwise operator

void bitwise\_swap(){

int a=1;

int b=2;

printf("%d %d\n",a,b);

a=a^b; //01+10=11=3

b=a^b; //11+10=01=1

a=a^b; //011+001=010

printf("a = %d b = %d\n",a,b);

}

void add(int \*p,int \*q){

int sum=\*p+\*q;

printf("The sum of a + b is = %d \n",sum);

}

void swap(int \*p,int \*q){

int temp=\*p;

\*p=\*q;

\*q=temp;

}

// 1. Write a C program to check Least Significant Bit (LSB) of a number is set or not.

void LSB(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

if (Bit2(Bit,n)) {

printf("The least significant bit is set (LSB = 1).\n");

} else {

printf("The least significant bit is not set (LSB = 0).\n");

}

}

// 2. Write a C program to check Most Significant Bit (MSB) of a number is set or not.

void MSB1(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int bits=sizeof(int)\*8;

int msb=MSB(bits,Bit);

if(n&msb){

printf("MSB is set to 1\n");

}

else{

printf("MSB is Set to 0\n");

}

}

// 3. Write a C program to get nth bit of a number.

void NthBit(){

int num;

printf("Enter the numeber : ");

scanf("%d",&num);

int nthNumber;

printf("Enter the nth position : ");

scanf("%d",&nthNumber);

if((NTHBIT(num,nthNumber))&Bit){

printf("number %d in this the position of %d bit is 1\n",num,nthNumber);

}

else{

printf("number %d in this the position of %d bit is 0 \n",num,nthNumber);

}

}

//4. Write a C program to set nth bit of a number.

void setNthBit(){

int num;

printf("Enter the number : ");

scanf("%d",&num);

int nthBitPosition;

printf("Enter the position : ");

scanf("%d",&nthBitPosition);

int setNthBit=SetBit(Bit,nthBitPosition,num);

printf("Before setting the nth bit : %d\n",num);

printf("After setting the nth bit : %d\n",setNthBit);

}

// 5. Write a C program to clear nth bit of a number

void clearNthBit(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int nthBitPosition;

printf("Enter the position to clear : ");

scanf("%d",&nthBitPosition);

int newClearNthBit=ClearBit(Bit,nthBitPosition,n);

printf("Before cleaning the nth Bit : %d\n",n);

printf("After cleaning the nth Bit : %d\n",newClearNthBit);

}

void toggleBit(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos;

printf("Enter the pos : ");

scanf("%d",&pos);

// pos=pos-1;

int bit=ToggleBit(bit,pos,Bit,n); // 1 0 1 0

// 0 1 0 0

printf("The new value is : %d\n",bit);

}

void highBit(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos=0;

for(int i=32;i>=0;i--){

if(HighBit(n,i,Bit)){

pos=i;

break;

}

}

printf("The High index of 1 bit is : %d",pos+1);

}

void highBit1(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos=0;

for(int i=32;i>=0;i--){

if(HighBit1(Bit,i,n)){ // #define HighBit1(Bit,i,n) Bit<<i)&n

pos=i;

break;

}

}

printf("The high index of 1 bit is : %d",pos+1);

}

void lowBit2(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos=0;

for(int i=32;i>=0;i--){

if(LowBit(Bit,i,n)){

pos=i;

}

}

printf("The low index of 1 bit is : %d",pos+1);

}

void lowBit13(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos=0;

for(int i=0;i<=32;i++){

if(LowBit13(i,Bit,n)){

pos=i;

break;

}

}

printf("The low index of 1 bit is : %d",pos+1);

}

void lowBit1(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int pos=0;

for(int i=0;i<=32;i++){

if(LowBit1(n,i,Bit)){

pos=i;

break;

}

}

printf("The low index of 1 bit is : %d",pos+1);

}

// 9. Write a C program to count trailing zeros in a binary number.

void trailingZero(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int count=0;

for(int i=0;i<32;i++){

if(TrailingZero(Bit,i,n)){

break;

}

else{

count++;

}

}

printf("Count of trailing zeros in number : %d",count);

}

#define mul(a,b) a\*b

void show(){

printf("Redult : %d",mul(2,3));

}

// 10. Write a C program to count leading zeros in a binary number

void leadingZero(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int binBit,count=0;

printf("For how many bit you want : ");

scanf("%d",&binBit);

for(int i=binBit-1;i>=0;i--){

if(LeadingZero(Bit,i,n)){

break;

}

else{

count++;

}

}

printf("The count of Leading count is : %d\n",count);

}

// 11. Write a C program to flip bits of a binary number using bitwise operator.

void flipBin(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

printf("The flip value is : %d",Flip(n));

}

// 12. Write a C program to count total zeros and ones in a binary number.

void countBit(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

int temp=n;

int count0=0,count1=0;

for(int i=0;i<32;i++){

if(CountBit(Bit,i,n)){

count1++;

}

else{

count0++;

}

}

printf("The number 1 count is : %d\n",count1);

printf("The number 0 count is : %d\n",count0);

}

// 13. Write a C program to rotate bits of a given number.

// 15. Write a C program to swap two numbers using bitwise operator.

void swap(){

int a,b;

printf("Enter the a and b value : ");

scanf("%d%d",&a,&b);

printf("a and b value is\na = %d\nb = %d\n",a,b);

printf("a and b value is\na = %d\nb = %d\n",Swap(a,b));

}

// 16. Write a C program to check whether a number is even or odd using bitwise operator.

void evenOdd(){

int n;

printf("Enter the number : ");

scanf("%d",&n);

}

int main(){

//lowBit1();

// show();

// LSB();

// MSB1();

// NthBit();

// setNthBit();

// clearNthBit();

// toggleBit();

// highBit();

// lowBit2();

// lowBit13();

// lowBit1();

// trailingZero();

// leadingZero();

// flipBin();

// countBit();

swap();

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*JENNY’S VIDEO\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include<stdio.h>

#include<conio.h>

#include<string.h>

#include<stdlib.h>

void do\_while\_loop1();

void input\_char(){

char ch;

printf("Enter the character : ");

ch=getchar();

printf("The character is : %c \n",ch);

getch(); //optinal

}

// Bitwise Operator

void and\_op(){

int n1,n2;

printf("Enter the number 1 : ");

scanf("%d",&n1);

printf("Enter the number 2 : ");

scanf("%d",&n2);

printf("The and num1 &(and) num2 is = %d\n",n1&n2);

}

void or\_op(){

int n1,n2;

printf("Enter the number 1 : ");

scanf("%d",&n1);

printf("Enter the number 2 : ");

scanf("%d",&n2);

printf("The and num1 |(or) num2 is = %d\n",n1|n2);

}

void not\_op(){

int n1;

printf("Enter the number 1 : ");

scanf("%d",&n1);

printf("The and num1 ~(not) = %d\n",~n1);

}

void left\_shift(){

int n1,shift;

printf("Enter the number 1 : ");

scanf("%d",&n1);

printf("Enter how many number you want to shift : ");

scanf("%d",&shift);

printf("The left shifted number %d to postion %d = %d\n",n1,shift,n1<<shift);

}

void right\_shift(){

int n1,shift;

printf("Enter the number 1 : ");

scanf("%d",&n1);

printf("Enter how many number you want to shift : ");

scanf("%d",&shift);

printf("The right shifted number %d to postion %d = %d\n",n1,shift,n1>>shift);

}

void xor\_op(){

int n1,n2;

printf("Enter the number 1 : ");

scanf("%d",&n1);

printf("Enter the number 2 : ");

scanf("%d",&n2);

// 0 1 =1 why means one is false another one is true then the condtion is true else its false

printf("The and num1 ^(xor-op) num2 is = %d\n",n1^n2);

}

void do\_while\_loop(){

int i=0;

do{ // first enter inside and execute the statements after that it will check the condition

// so it's called as exit level loop // it execute atleast once condtion true or false

printf("One time it print : \n");

}

while(i>0); // it's mandatory to use while ended with the semicolon

do\_while\_loop1(); // calling another function

}

void do\_while\_loop1(){

int i=0;

do{

printf("%d \n",i);

i++;

}

while(i<10);

}

// 2-D array concepts

void sum\_of\_individual\_array(){

int arr[3][3];

int sr,sc;

for(int i=0;i<3;i++){

printf("Enter the %d rom arrays here : \n",i+1);

for(int j=0;j<3;j++){

printf("Enter the %d array %d : ",i+1,j+1);

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

}

}

for(int i=0;i<3;i++){

sr=0,sc=0;

for(int j=0;j<3;j++){

sr=sr+arr[i][j];

sc=sc+arr[j][i];

}

printf("Sum of Rows : %d\n",sr);

printf("Sum of Column : %d\n",sc);

}

}

void sum\_of\_matrix(){

int arr1[2][3]={{1,2,3},{1,1,1}};

int arr2[2][3]={{1,0,1},{1,1,1}};

int arr3[2][3];

for(int i=0;i<2;i++){

for(int j=0;j<3;j++){

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

}

}

for(int i=0;i<2;i++){

for(int j=0;j<3;j++){

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

}

printf("\n");

}

}

void mul\_2Matrics(){

int arr1[3][3];

int arr2[3][3];

int arr3[3][3];

for(int i=0;i<3;i++){

printf("Enter the %d row values : \n",i+1);

for(int j=0;j<3;j++){

printf("Enter array values %d : ",j+1);

scanf("%d",&arr1[i][j]);

}

}

printf("2 Array Values \n");

for(int i=0;i<3;i++){

printf("Enter the %d row values : \n",i+1);

for(int j=0;j<3;j++){

printf("Enter array values %d : ",j+1);

scanf("%d",&arr2[i][j]);

}

}

for(int i=0;i<3;i++){

for(int j=0;j<3;j++){

arr3[i][j]=0;

for(int k=0;k<3;k++){

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

}

}

}

for(int i=0;i<3;i++){

for(int j=0;j<3;j++){

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

}

printf("\n");

}

}

void mul\_2Matrics1(){

int arr1[2][2]={{2,1},{2,3}};

int arr2[2][2]={{2,3},{5,2}};

int arr3[2][2];

for(int i=0;i<2;i++){

for(int j=0;j<2;j++){

arr3[i][j]=0;

for(int k=0;k<2;k++){

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

}

}

}

for(int i=0;i<2;i++){

for(int j=0;j<2;j++){

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

}

printf("\n");

}

}

void pointers(){

int a=10,b=9;

int \*p=&a,\*q=&b;

printf("a = %d\n",a);

printf("a = %d\n",\*p);

printf("a = %d\n",\*(&a));

printf("a = %d\n",\*(p));

printf("Address of a \n");

printf("a = %d\n",&a);

printf("a = %d\n",p);

int c=\*q;

printf("c = %d\n",c);

\*p=20;

printf("a = %d\n",a);

printf("a = %d\n",\*p);

printf("address of a \n");

printf("a = %d\n",p);

\*q=19;

printf("b = %d\n",b);

q=&b;

printf("b = %d\n",\*(q));

printf("Pointers adreess : \n");

printf("p = %d\n",&p);

}

void pointers1(){

int a=10,b=9;

int \*p,\*q;

p=&a; //assogn the value address of a to p

q=p; // this is ilegal that we can assign the p value to q because p in the pointer //assign a address to q

// q=\*p; // this is ilegal that we can't assign the value of \*p(a) to q pointer

\*q=\*p; // this is legal that we can assign the \*q(address of a) to \*p(value of a)

printf("Value of a is : \n");

printf("a = %d\n",a); // a value

printf("a = %d\n",\*p); //

printf("a = %d\n",\*q);

printf("a = %d\n",\*q);

}

void pointers2(){

int a=100;

int \*p=&a;

int \*\*q=&p;

printf("The values of a \n");

printf("a = %d\n",a);

printf("a = %d\n",\*p);

printf("a = %d\n",\*(\*q));

printf("Way of modifying the a \n");

\*p=200;

printf("update a = %d\n",\*p);

printf("Another Way of modifying the a \n");

\*\*q=250;

printf("update a = %d\n",\*\*q);

a=300;

printf("a = %d\n",\*\*q);

\*\*q=a; // this is ok that we can assign the direct value to \*\*q // same as \*\*q=300;

// \*\*q=&a; // this is ilegal it's wrong that we are assigning the address of a to value of \*\*q so its wrong

\*p=350;

printf("\*q=p;,\*\*q=\*p; are same both\n");

\*q=p;//350 // this is also wrong because we can assign this to pointer address

\*\*q=\*p; //

printf("%d\n",\*\*q);

printf("Address of values : \n");

printf("a = %d\n",&a);

printf("a(p) = %d\n",p); // this is the address of a because pointer p is holding the value of a

printf("p = %d\n",&p);

printf("p(q) = %d\n",q);

printf("q = %d\n",&q);

int \*\*\*r=&q;

printf("q = %d\n",r);

}

void pointers3(){

int arr[5]={0,1,-1,10,1};

int \*p=arr; // this will take the base address of array[0]

// no need to wite this int \*ptr=&arr; // because array is alredy a constant pointer so we can't give

printf("By using pointer accesiing value \n");

// for(int i=0;i<5;i++){

// printf("arr %d = %d\n",i,\*p++);

// }

p=p+2;

printf("%d\n",\*p);

p=&arr[0];

printf("%d\n",\*p);

p=&arr[2];

printf("%d\n",\*p);

p=p+1;

\*p=2;

printf("%d\n",\*p);

printf("%d\n",arr[3]);

int \*p2;

p2=&arr[1];

}

void pointers4(){

int arr[]={1,8,-1,5,6,7};

// This below to declaration is same

int \*p=&arr[0];

int \*q=arr;

printf("This below both the value is same \n");

printf("p = %d\n",\*p);

printf("q = %d\n",\*q);

p=&arr[0];

q=&arr[3];

int d=q-p;

printf("size difference between q to p is : %d\n",d);

p=&arr[0];

q=&arr[3];

d=p-q;

printf("size difference between p to q is : %d\n",d);

q=q-2;

printf("%d\n",\*q);

}

void pointers5(){

int a[]={10,11,-1,56,67,5,4};

int \*p,\*q;

p=a;

printf("%d\n",\*p);

printf("%d %d %d\n",(\*p)++,\*p++,\*++p);

q=p+3;

printf("%d\n",\*q-3);

printf("%d\n",\*--p+5);

printf("%d\n",\*p+\*q);

printf("%d\n",a[1]);

}

void pointers6(){

int a[]={10,11,-1,56,67,7,5,4};

int \*p,\*q;

p=a;

q=&a[0]+3;

printf("%d %d %d\n",(\*p)++,(\*p)++,\*(++p));

printf("%d\n",\*p);

printf("%d\n",(\*p)++);

printf("%d\n",(\*p)++);

q--;

printf("%d\n",(\*(q+2))--);

printf("%d\n",\*(p+2)-2);

printf(" %d \n",\*p);

printf("%d\n",\*(p++ -2)-1); // in this it contains a garbage value

printf("%d\n",\*(p++ -2)-1);

}

void pointers7(){

const int a=-11;

const int \*p=&a;

printf("%d\n",a);

}

void string1(){

char name[5];

printf("Enter the name : ");

scanf("%s",name); // without & can also we use it

printf("the string is = %s\n",name);

}

void string2(){

char name[5];

printf("Enter the name : ");

scanf("%s",name); // without & can also we use it

printf("the string is = %s\n",name); // it though declare the size then also it can acess the entire string and print it

}

void string3(){

char name[5];

printf("Enter the name : ");

gets(name); // we can use this also

printf("the string is = %s\n",name); // it though declare the size then also it can acess the entire string and print it

}

// difference between gets and scanf

void string4(){

char name1[5];

char name2[5];

printf("Enter the name : ");

gets(name1); // we can use this also // here can't directle select how much characater we want because it take only one argument

printf("the string is = %s\n",name1); // it though declare the size then also it can acess the entire string and print it

printf("Enter the name2 : ");

scanf("%3s",name2); // you here directle select how much character you want in output

printf("The string2 is = %s\n",name2);

}

// difference between puts and printf

void string5(){

char name1[5];

char name2[5];

printf("Enter the name : ");

gets(name1); //

puts(name1); // we can use this also // here can't directle select how much characater we want because it take only one argumenta

printf("Enter the name2 : ");

scanf("%s",name2);

printf("The string2 is = %2s\n",name2); //// you here directle select how much character you want in output

}

void string6(){

char name1[10];

printf("Enter the name : ");

scanf("%s",name1);

printf("%s\n",&name1); // you can give & also ok it will print the string

printf("%s\n",&name1[2]); // it will print the character string index of 2 to end of string (till null pointer)

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Code Vault Channels Video\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// strings Examples

// string declaration uses and how many ways we can declare

// example 1

// here we can modify the the string

void string101(){

char name[]="Hello!";

printf("%s\n",name);

name[0]='h';

printf("%s\n",name);

}

// example 2

// here we can't modify the the string

// even there is no error then also it will not modify the string and show some warnings

void string102(){

char \*name="Hello!";

printf("%s\n",name);

name[0]='h';

printf("%s\n",name);

}

//// example 3

//// here we can't modify the the string

//// it's a error

//

//void string103(){

// const static char \*name="Hello";

// printf("%s\n",name);

// name[0]='h'; // it's a read only so we can't modify

// printf("%s\n",name);

//}

// example 4

// here we can modify the the string

// even there is no error then also it will not modify the string and show some warnings

// this example is same as above example 2

void string104(){

char \*name=(char\*)malloc(10\*sizeof(char));

strcpy(name,"Hello");

printf("%s\n",name);

name[0]='h'; // it's a read only so we can't modify

printf("%s\n",name);

}

int main(){

// input\_char();

// and\_op();

// or\_op();

// not\_op();

// left\_shift();

// right\_shift();

// xor\_op();

// do\_while\_loop();

// sum\_of\_individual\_array();

// sum\_of\_matrix();

// mul\_2Matrics();

// mul\_2Matrics1();

// pointers();

// pointers1();

// pointers2();

// pointers3();

// pointers4();

// pointers5();

// pointers6();

// pointers7();

// string1();

// string2();

// string3();

// string4();

// string5();

// string6();

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Code Vault Channels Video\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//string101();

// string102();

//string103();

string104();

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*NESTED STRUCTURE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct Employee {

char empname[20];

int id;

int sal;

};

struct Company {

int id;

char cmpname[20];

char address[100];

struct Employee emp;

};

// here the printing details of emp and comp

void disp\_Values(struct Company \*ptr,int n){

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

Company \*temp=&ptr[i];

printf("Here is the details of %d\n",i+1);

printf("Emp name : %s\n",temp->emp.empname);

printf("Emp ID : %d\n",temp->emp.id);

printf("Emp sal : %d\n",temp->emp.sal);

printf("Comp Emp ID : %d\n",temp->id);

printf("Company name : %s\n",temp->cmpname);

printf("Company address : %s\n",temp->address);

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

}

}

// here intializing the values of comp

void intialiazingComplValues(struct Company \*ptr, int n) {

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

Company \*temp=&ptr[i];

printf("Enter here details of Company %d\n",i+1);

printf("Enter the Emp ID : ");

scanf("%d",&temp->id);

printf("Enter the Company name : ");

scanf("%s",&temp->cmpname);

printf("Enter the Company address : ");

scanf("%s",&temp->address);

}

}

// here intializing the values of emp

void intialiazingEmpValues(struct Company \*ptr, int n) {

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

Company \*temp=&ptr[i];

printf("Enter here the details of Employee %d\n",i+1);

printf("Enter the Emp name : ");

scanf("%s",&temp->emp.empname);

printf("Enter the Emp ID : ");

scanf("%d",&temp->emp.id);

printf("Enter the Emp salary : ");

scanf("%d",&temp->emp.sal);

intialiazingComplValues(temp,1);

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

}

}

int main() {

int n;

printf("Enter the number of Employee : \n");

scanf("%d", &n);

// we are creating structure array

struct Company \*cmp = (struct Company\*)malloc(n \* sizeof(struct Company));

// intialiazing the values of emp and comp

intialiazingEmpValues(cmp, n);

// printing the values of emp and comp

disp\_Values(cmp,n);

// Don't forget to free the allocated memory when done

free(cmp);

return 0;

}

2)

//#include <stdio.h>

//#include<stdlib.h>

//float PercentageCalculator(){

// int sub;

// printf("Enter the how many subject : ");

// scanf("%d",&sub);

// int totalMarks=sub\*100;

// int arr[sub];

// for(int i=0;i<sub;i++){

// printf("Enter the sub %d marks : ",i+1);

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

// }

// int obtainMarks=0;

// for(int i=0;i<sub;i++){

// obtainMarks=obtainMarks+arr[i];

// }

//

// float percentage = ((float)obtainMarks/totalMarks)\*100;

// // printf("%.2f%%\n",percentage);

// return percentage;

//

//}

//void Students\_Array(){

// struct students{

// char name[10];

// float subject;

// char number[10];

// int cls;

// };

// int n;

// printf("Enter how many student details you want to add : \n");

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

// struct students \*student=(students\*)malloc(n\*sizeof(students));

//

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

// students \*currentStudent = &student[i];

// printf("Enter the student name : ");

// scanf("%s",&currentStudent->name);

// currentStudent->subject = PercentageCalculator();

// printf("Enter the Phone number of the student : ");

// scanf("%s",&currentStudent->number);

// printf("Enter the Standard : ");

// scanf("%d",&currentStudent->cls);

// }

//

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

// students \*currentStudent = &student[i]; // Get a pointer to the current student

// printf("Here is the details of %d student \n",i+1);

// printf("The name of student is : %s\n",currentStudent->name);

// printf("The phone number of student : %s\n",currentStudent->number);

// printf("The class of the student is : %d\n",currentStudent->cls);

// printf("The percentage of the student is : %.2f%%\n",currentStudent->subject);

// printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

// }

//

//}

//int main()

//{

// Students\_Array();

// return 0;

//}

// Example of Employee

#include<stdio.h>

#include<stdlib.h>

struct Employee {

char empname[10];

int empid;

int sal;

char company[20];

};

int main(){

int n;

printf("Enter the number of employee you want : ");

scanf("%d",&n);

Employee \*emp=(Employee\*)malloc(n\*sizeof(emp));

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

Employee \*temp=&emp[i];

printf("Enter the %d Employee Details : \n",i+1);

printf("Enter the Emp name : ");

scanf("%s",&temp->empname);

printf("Enter the Emp id : ");

scanf("%d",&temp->empid);

printf("Enter the Emp salary : ");

scanf("%d",&temp->sal);

printf("Enter the company name : ");

scanf("%s",&temp->company);

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

}

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

Employee \*temp=&emp[i];

printf("Here is the Details of Employee %d \n",i+1);

printf("Emp name : %s\n",temp->empname);

printf("Emp id : %d\n",temp->empid);

printf("Emp salary : %d\n",temp->sal);

printf("Emp company name : %s\n",temp->company);

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

}

return 0;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*STRING TYPES\*\*\*\*\*\*\*\*\*\*\*(\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include<stdio.h>

#include<string.h>

int main(){

char s1[10]="Ali";

char s2[10]="Ali";

// 1st type of string type

printf("This is the cancatination of String \n");

puts(strcat(s1,s2)); // concatination of two string

// 2nd type of string type

printf("This is the reversing of String \n");

puts(strrev(s1));

//3rd type of String type

printf("This is the checking the length of String \n");

printf("%d\n",strlen(s1));

// 4th type of string type

printf("This is way of comparing the String \n");

printf("%d\n",strcmp(s1,s2));

if(strcmp(s1,s2)){

printf("The given string is same \n");

}

else{

printf("The given string is not same \n");

}

// 5th type of String type

printf("This is way of copying string from one string to another string \n");

char s3[30];

strcpy(s3,s2);

puts(s3);

// another way of type 5

printf("First concatinating 2 string and then copy that string to another string \n");

char a[]="ALi";

char b[]="AS";

char c[40];

strcpy(c,strcat(a,b));

puts(c);

return 0;

}