|  |  |  |
| --- | --- | --- |
| **Name: SUNDEEP A** | **SRN: PES1UG20CS445** | **Section: O** |
| **Date: 10-06-21** | **Week Number: 6** |

|  |  |
| --- | --- |
| **1** | **Write a C program to generate Pascal triangle using two dimensional array**  **Input:**  Enter the n value:  4  **Output:**  1  1 1  1 2 1  1 3 3 1 |
|  | **Program:**  **#include<stdio.h>**  **void pasctri(int n);**  **int main() //main**  **{**  **int n;**  **printf("Enter the value of n = "); //taking input from the user**  **scanf("%d",&n);**  **pasctri(n); //passing the value entered by the user to the function**  **return 0;**  **}**  **void pasctri(int n) //user defined function to solve the task**  **{**  **int a[10][10];**  **int i,j;**  **printf("The pascal triangle is:\n");**  **for(i=1;i<=n;i++) //outer loop**  **{**  **for(j=1;j<=i;j++) //inner loop**  **{**  **if(j==1 || j==i) //storing 1 at the beginning and ending of the row**  **a[i][j]=1;**  **else**  **a[i][j]=a[i-1][j]+a[i-1][j-1]; //adds the numbers which are in between and stores it in the array**  **printf("%d ",a[i][j]); //displays the value that is stored**  **}**  **printf("\n");**  **}**  **}** |
|  | **Output Screenshot:** |
| **2** | Write a C program to read elements in a matrix and check whether the given matrix is symmetric matrix or not.  **Input:**  Enter the value of m  3  Enter the value of n  3  Enter elements in matrix of size 3x3:  1  0  0  0  1  0  0  0  1  **Output:**  The given matrix is Symmetric matrix:  1 0 0  0 1 0  0 0 1 |
|  | **Program:**  **#include<stdio.h>**  **void read(int a[10][10],int m,int n);**  **void disp(int a[10][10],int m,int n);**  **void sym(int a[10][10],int trans[10][10],int m,int n);**  **int main() //main**  **{**  **int m,n;**  **printf("Enter the value of rows and columns of the matrix\n");**  **scanf("%d %d",&m,&n);**  **int a[10][10];**  **int trans[10][10];**  **read(a,m,n); //passing the size of the matrix to the read function**  **disp(a,m,n); //displays the matrix entered by the user**  **sym(a,trans,m,n); //Checks if the matrix entered by the user is symmertic or not,by finging th transpose of the matrix**  **return 0;**  **}**  **void read(int a[10][10],int m,int n) //user-defined read function to read the elements of the matrix**  **{**  **printf("Enter the elements of the matrix\n");**  **for(int i=0;i<m;i++) //outer loop**  **{**  **for(int j=0;j<n;j++) //inner loop**  **scanf("%d",&a[i][j]); //storing the values in a[i][j]**  **}**  **}**  **void disp(int a[10][10],int m,int n) // user defined function to display the elements entered by the user**  **{**  **printf("Entered elements of the matrix are :\n");**  **for(int i=0;i<m;i++) //outer loop**  **{**  **for(int j=0;j<n;j++) //inner loop**  **printf("%d ",a[i][j]); //displays the element stored in a[i][j]**  **printf("\n"); //this is used to go to new row**  **}**  **}**  **void sym(int a[10][10],int trans[10][10],int m,int n) //User defined function to check if a matrix is symmetric or not**  **{**  **int i;int j;**  **for(i=0;i<m;i++) //outer loop**  **{**  **for(j=0;j<n;j++) //inner loop**  **trans[j][i]=a[i][j]; //finding the transpose of a matrix and storing it in trans[][]**  **}**  **printf("The transpose matrix\n");**  **for(i=0;i<m;i++) //outer loop**  **{**  **for(j=0;j<n;j++) //inner loop**  **printf("%d ",trans[i][j]); //displaying the transpose of a matrix**  **printf("\n"); //to navigate to the next row**  **}**  **if(m==n) //checks if the number of rows and columns are equal**  **{**  **int count=0;**  **for(i=0;i<m;i++) //outer loop**  **for(j=0;j<n;j++) //inner loop**  **if(a[i][j]!=trans[i][j]) //checks if matrix and its transpose are equal**  **count=count+1; //increments the count if the elements are not equal**  **if(count==0)**  **printf("\nThe matrix is symmetric"); //prints symmetric if all the elements are equal**  **else**  **printf("\nThe matrix is not symmetric");**  **}**  **else**  **printf("\nThe matrix is not symmetric");**  **}** |
|  | **Output Screenshot:** |
| **3** | Write a C program to compare 2 dates and print appropriate message using structures  **Input1:**  Enter Date1 in the format dd/mm/yyyy  12/2/2000  Enter Date2 in the format dd/mm/yyyy  12/2/2000  Date1=12/2/2000  Date2=12/2/2000  **Output1:**  Date1 is equal to Date2  Input2:  Enter Date1 in the format dd/mm/yyyy  12/3/2000  Enter Date2 in the format dd/mm/yyyy  12/3/2001  Date1=12/3/2000  Date2=12/3/2001  **Output2:**  Date1 is smaller than Date2  **Input3:**  Enter Date1 in the format dd/mm/yyyy  12/4/1999  Enter Date2 in the format dd/mm/yyyy  12/2/1999  Date1=12/4/1999  Date2=12/2/1999  **Output3:**  Date1 is greater than Date2 |
|  | **Program:**  **#include<stdio.h>**  **typedef struct date{ //structure to store date**  **int dd;**  **int mm;**  **int yy;**  **}date\_info; //typedef date\_info**  **void date\_read(date\_info \*d);**  **void date\_disp(const date\_info \*d);**  **int date\_cmp(const date\_info \*d1,const date\_info \*d2);**  **int main() //main**  **{**  **date\_info d1; //d1 variable of date\_info is created**  **date\_info d2; //d2 variable of date\_info is created**  **printf("Enter a valid first date in the order dd|mm|yyyy\n");**  **date\_read(&d1); //user defined function to read the date entered by the user**  **printf("Enter a valid second date in the order dd|mm|yyyy\n");**  **date\_read(&d2); //user defined function to read the date entered by the user**  **printf("\nFirst date:");**  **date\_disp(&d1); //user defined function to display the date entered by the user**  **printf("\nSecond date:");**  **date\_disp(&d2); //user defined function to display the date entered by the user**  **printf("\n");**  **int res;**  **res=date\_cmp(&d1,&d2); //address of both the dates are passed to the function to compare the dates**  **if(res==0)**  **printf("The dates are equal\n");**  **else if(res>0)**  **printf("First date is greater than the second date\n");**  **else**  **printf("First date is smaller than the second date\n");**  **return 0;**  **}**  **void date\_read(date\_info \*d)**  **{**  **scanf("%d|%d|%d",&(d->dd),&(d->mm),&(d->yy)); //date is stored in the structure**  **}**  **int date\_cmp(const date\_info \*d1,const date\_info \*d2)**  **{**  **int res;**  **if((d1->dd==d2->dd) && (d1->mm==d2->mm) && (d1->yy==d2->yy)) //date 1 is compared with date2**  **res=0; //return 0 if they are equal**  **else if( d1->yy > d2->yy) //checking if date 1 is greater than date 2**  **res=1;**  **else if(d1->mm > d2->mm)**  **res=1;**  **else if(d1->dd > d2->dd)**  **res=1;**  **else**  **res=-1;**  **return res;**  **}**  **void date\_disp(const date\_info \*d)**  **{**  **printf("%d|%d|%d",d->dd,d->mm,d->yy); //displays the date entered by the user**  **}** |
|  | **Output Screenshot:** |
| 4 | Write a C Program to Add and subtract two Complex Numbers by Passing Structure to a Function  **Input:**  For 1st complex number  Enter the real and imaginary parts: 5  4  For 2nd complex number  Enter the real and imaginary parts: 3  2  **Output:**  Sum = 8.0 + 6.0i  Sub = 2.0 - 2.0i |
|  | Program:  #include<stdio.h>  typedef struct complex{ //structure is created to store the real and imaginary part  float real;  float img;  }complex; //structure of typedef complex  complex add(complex n1,complex n2); //function to add two complex numbers  complex sub(complex n1,complex n2); //function to subtract two complex numbers  int main()  {  complex n1,n2;  printf("First complex number\n");  printf("Enter the real and imaginary part\n"); //reads the first complex number  scanf("%f %f",&n1.real,&n1.img);  printf("Second complex number\n");  printf("Enter the real and imaginary part\n"); //reads the second complex number  scanf("%f %f",&n2.real,&n2.img);  complex sum=add(n1,n2);  complex diff=sub(n1,n2);  printf("Sum=%.2f+(%.2f)i\n",sum.real,sum.img); //prints the sum of the complex numbers  printf("Difference=%.2f+(%.2f)i\n",diff.real,diff.img); //prints the difference of two complex numbers  return 0;  }  complex add(complex n1,complex n2)  {  complex temp;  temp.real=n1.real+n2.real; //the real part are added  temp.img=n1.img+n2.img; //imaginary part are added  return temp;  }  complex sub(complex n1,complex n2)  {  complex temp;  temp.real=n1.real-n2.real; //the real part are subtracted  temp.img=n1.img-n2.img; //the imaginary part are subtracted  return temp;  } |
|  | Output Screenshot: |
| 1 | **Practice Programs**  Write a program that fills a five-by-five matrix as follows:  Upper left triangle with +1s  Lower right triangle with -1s  Right to left diagonal with zeros  Display the contents of the matrix using not more than two printf statements  **Output:**  This is 5x5 Matrix  1 1 1 1 0  1 1 1 0 -1  1 1 0 -1 -1  1 0 -1 -1 -1  0 -1 -1 -1 -1 |
|  | Program:  #include<stdio.h>  int main() //main  {  int a[10][10];  for(int i=0;i<5;i++) //outer loop  {  for(int j=0;j<5;j++) //inner loop  {  if(i+j<4)  a[i][j]=1; //filling Upper left triangle with +1  else if(i+j>=5)  a[i][j]=-1; //filling Lower right triangle with -1  else  a[i][j]=0; //filling the diagonal with 0  printf("%5d",a[i][j]); //printing the elements one by one  }  printf("\n");  }  return 0;  } |
|  | Output Screenshot: |
| 2 | Write a Program to add two distances in the inch-feet system using structures  **Input:**  Enter 1st distance  Enter feet: 23  Enter inch: 10  Enter 2nd distance  Enter feet: 34  Enter inch: 2.4  **Output:**  Sum of distances = 58'-0.4" |
|  | Program:  #include<stdio.h>  struct inchfeet  {  int feet; //Structure is created to hold inches ,feet  float inches;  }typedef length;  length sum(length l1,length l2)  {  length sum;  sum.feet=l1.feet+l2.feet; //adds the two feet  sum.inches=l1.inches+l2.inches; //adds the inches  if(sum.inches>12) //checks if inches >12  {  sum.inches-=12; //inches-12  sum.feet+=1; //feet+1  }  return sum;  }  int main()  {  length l1;  length l2;  length total;  printf("Enter the first distance\n"); //inputs the first distance  printf("Enter feet\n");  scanf("%d",&l1.feet);  printf("Enter inches\n");  scanf("%f",&l1.inches);  printf("Enter the second distance\n"); //inputs the second distance  printf("Enter feet\n");  scanf("%d",&l2.feet);  printf("Enter inches\n");  scanf("%f",&l2.inches);  total=sum(l1,l2); //the two distances are passed to the function sum  printf("The sum = %d feet-%.2f inch",total.feet,total.inches);  return 0;  } |
|  | Output Screenshot: |
|  | **Extra program :**  #include<stdio.h>  void pasctri(int n);  int main()  {  int n;  printf("Enter the value of n = "); //taking input from the user  scanf("%d",&n);  pasctri(n); //passing the value entered by the user to the function  return 0;  }  void pasctri(int n)  {  int a[100][100];  int i;int j;  printf("The pascal triangle is:\n");  for(i=1;i<=n;i++)  {  for(int z=0;z<(n-i);z++) printf(" "); //it works similar to the basic code but i have added an additional for loop to get spaces  for(j=1;j<=i;j++)  {  if(j==1 || j==i)  a[i][j]=1;  else  a[i][j]=a[i-1][j]+a[i-1][j-1];  printf("%d ",a[i][j]);  }  printf("\n");  }  }  **OUTPUT :** |