|  |  |
| --- | --- |
| **Name** | **Hatim Yusuf Sawai** |
| **UID no.** | **2021300108** |
| **Experiment No.** | 6 |

|  |  |
| --- | --- |
| **AIM:** | Demonstrate the use of two-dimensional arrays to solve a given problem. |
| **Program 1** | |
| **PROBLEM STATEMENT:** | Write a program to perform Matrix Addition, Subtraction, Multiplication, Transpose of Matrix and Norm of Matrix. Dimensions of matrices will be decided by user. |
| **ALGORITHM:** | 1. START  2. Define void function zero with a float 2D array mat[m][n] as  parameter  3. Initialize all elements to 0  4. Define void function print with a float 2D array mat[m][n] as  parameter  5. I=0  6. J=0  7. Print mat[i][j]  8. J++  9. Repeat 7,8 till j<n  10. I++  11. Repeat 6,7,8,9 and 10 till i<m  12. Define void function add with 2 2D float array mat1[m][n] and  mat2[a][b] as parameters  13. I=0  14. J=0  15. Print mat1[i][j]+mat2[i][j]  16. J++  17. Repeat 15,16 till j<n  18. I++  19. Repeat 14,15,16,17 and 18 till i<m  20. Define void function sub with 2 2D float array mat1[m][n] and  mat2[a][b] as parameters  21. I=0  22. J=0  23. Print mat1[i][j]-mat2[i][j]  24. J++  25. Repeat 23,24 till j<n  26. I++  27. Repeat 22,23,24,25 and 26 till i<m  28. Define void function multiply with 2 2D float array mat1[m][n] and  mat2[a][b] as parameters  29. Initialize 2D array mat3  30. Call function zero(m,b,mat3)  31. I=0  32. J=0  33. K=0  34. mat3[i][j] += mat1[i][k]\*mat2[k][j]  35. k++  36. repeat 34 and 35 till k<n  37. j++  38. repeat 33, 34, 35, 36 and 37 till j<b  39. i++  40. repeat 32, 33, 34, 35, 36, 37, 38 and 39 till i<m  41. call function print(m,b,mat3)  42. Define void function transpose with a 2D float array mat[m][n] as  parameter  43. Initialize 2D array newmat of dimension n x m  44. I=0  45. J=0  46. Newmat[i][j]=mat[j][i]  47. J++  48. Repeat 46 and 47 till j<m  49. I++  50. Repeat 45, 46, 47, 48 and 49 till i<n  51. Call function print(m,b,newmat)  52. Define int function norm with a 2D float array mat[m][n]  53. Initialize sum = 0.00  54. I=0  55. J=0  56. Sum += square of mat[i][j]  57. J++  58. Repeat 54 and 55 till j<n  59. I++  60. Repeat 55, 56, 57 ,58 and 59 till i<m  61. Sum = sqaure root of sum  62. Return sum  63. Define integer main function  64. Input dimensions of matrix 1 m and n  65. Input matrix 1 [m][n]  66. Input dimensions of matrix 2 a and b  67. Input matrix 2 [a][b]  68. If (m=a and b=n)  call function add(m,n,mat1,a,b,mat2)  else  print Addition not possible  69. If (m=a and b=n)  call function sub(m,n,mat1,a,b,mat2)  else  print subtraction not possible  70. If(n=a)  call function multiplication(m,n,mat1,a,b,mat2)  else  print multiplication not possible  71. Call function transpose(m,n,mat1)  72. Call function norm(a,b,mat2)  73. Print value of function norm(m,n,mat1)  74. Print value of function norm(a,b,mat2)  75. Return 0  76. STOP |
| **PROGRAM:** | #include <stdio.h>  #include <math.h>  void add\_mat(int m, int n, int mat1[m][n], int a, int b, int mat2[a][b])  {      for(int i=0;i<m;i++)      {          for(int j=0;j<n;j++)              printf("%d ",mat1[i][j]+mat2[i][j]);          printf("\n");      }  }  void sub\_mat(int m, int n, int mat1[m][n], int a, int b, int mat2[a][b])  {      for (int i = 0; i < m; i++)      {          for (int j = 0; j < n; j++)              printf("%d ", mat1[i][j] - mat2[i][j]);          printf("\n");      }  }  void print\_mat(int m,int n,int mat[m][n])  {      for(int i=0;i<m;i++)      {          for(int j=0;j<n;j++)              printf("%d ",mat[i][j]);          printf("\n");      }  }  void zero\_mat(int m, int n,int mat[m][n])  {      for (int i = 0; i < m; i++)      {          for (int j = 0; j < n; j++)              mat[i][j] = 0;      }  }  void mul\_mat(int m, int n, int mat1[m][n], int a, int b, int mat2[a][b])  {      int mat3[m][b];      zero\_mat(m,b,mat3);      for(int i=0;i<m;i++)      {          for(int j=0;j<b;j++)          {              for(int k=0;k<n;k++)                  mat3[i][j]+=mat1[i][k]\*mat2[k][j];          }      }      print\_mat(m,b,mat3);  }  void trans\_mat(int m,int n, int mat[m][n])  {      int mat3[n][m];      for (int i=0;i<n;i++)      {          for (int j=0;j<m;j++)              mat3[i][j] = mat[j][i];      }      print\_mat(n,m,mat3);  }  int frob\_norm(int m,int n,int mat[m][n])  {      float fr=0;      for(int i=0;i<m;i++)      {          for(int j=0;j<n;j++)              fr+= mat[i][j] \* mat[i][j];      }      return fr;  }  int main()  {      int m, n, a, b, i, j;      printf("Enter dimensions of Matrix 1:\n");      scanf("%d %d", &m, &n);      int mat1[m][n];      printf("Enter elements of Matrix 1:\n");      for (i = 0; i < m; i++)      {          for (j = 0; j < n; j++)              scanf("%d", &mat1[i][j]);      }      printf("Enter dimensions of Matrix 2:\n");      scanf("%d %d", &a, &b);      int mat2[a][b];      printf("Enter elements of Matrix 2:\n");      for (i = 0; i < m; i++)      {          for (j = 0; j < n; j++)              scanf("%d", &mat2[i][j]);      }      if(m==a && n==b)      {          printf("Additon of Matrices:\n");          add\_mat(m, n, mat1, a, b, mat2);          printf("Subtraction of Matrices:\n");          sub\_mat(m, n, mat1, a, b, mat2);      }      else          printf("Dimensions should be the same for addition & subtraction\n");      if(n==a)      {          printf("Multiplication of Matrices:\n");          mul\_mat(m, n, mat1, a, b, mat2);      }      else          printf("Multiplication is not possible");      printf("Transpose of Matrix 1:\n");      trans\_mat(m, n, mat1);      printf("Transpose of Matrix 2:\n");      trans\_mat(a, b, mat2);      printf("Frobenius Norm of Matrix 1 = %f\n",sqrt(frob\_norm(m,n,mat1)));      printf("Frobenius Norm of Matrix 2 = %f\n",sqrt(frob\_norm(a,b,mat2)));      return 0;  } |
| **RESULT:** | |
| **Program 2** | |
| **PROBLEM STATEMENT:** | Write a program that reads the current year followed by N followed by a list of N employee numbers and their current ages. Produce a list showing the years in which the employees retire (become 65 years old). If more than one employee retires in a given year then include them all under the same heading. |
| **ALGORITHM:** | 1. START  2. Define void function selection sort with an 2D integer array mat[n][2]  3. Define integer variables min, index  4. I=0  5. Index = i  6. J=I+1  7. If(mat[j][0] < mat[index][0])  index = j  8. J++  9. Repeat 7 and 8 till j<n  10. Initialize temp1 to mat[index][0]  11. Mat[index][0] = mat[i][0]  12. Mat[i][0] = temp1  13. Initialize temp2 to mat[index][1]  14. Mat[index][1] = mat[i][1]  15. Mat[i][1] = temp1  16. I++  17. Repeat steps 5 to 16 till i<n-1  18. Define integer main function  19. Input current year year  20. Input the number of employees n  21. I=0  22. Input current age mat[i][0] and employee number mat[i][1]  23. Mat[i][0] = year + 65 – mat[i][0]  24. Call function selection sort(n,mat)  25. I=0  26. If(I not equal to 0 and mat[i][0]=mat[i-1][0])  print Tabspace mat[i][1]  else  print mat[i][0] Tabspace mat[i][1]  27. Return 0  28. STOP |
| **FLOWCHART:** |  |
| **PROGRAM:** | #include<stdio.h>  void print\_emp(int n, int emp[n][2])  {      printf("Ret Year    Emp No.\n");      for (int i=0;i<n;i++)      {          if (i!= 0 && emp[i][0]==emp[i-1][0])              printf("            %d\n", emp[i][1]);          else              printf("%d        %d\n", emp[i][0], emp[i][1]);      }  }  void selectionsort(int n, int emp[n][2])  {      int min,temp1,temp2;      for (int i = 0; i < n - 1; i++)      {          min = i;          for (int j = i + 1; j < n; j++)          {              if (emp[j][0] < emp[min][0])              {                  min = j;              }          }          temp1 = emp[min][0];          emp[min][0] = emp[i][0];          emp[i][0] = temp1;          temp2 = emp[min][1];          emp[min][1] = emp[i][1];          emp[i][1] = temp2;      }      print\_emp(n,emp);  }  int main()  {      int cy,n,i,j,ra;      printf("Enter the Current Year:\n");      scanf("%d",&cy);      printf("Enter Retirement Age:\n");      scanf("%d", &ra);      printf("Enter number of entries:\n");      scanf("%d",&n);      int ret\_emp[n][2];      for(i=0;i<n;i++)      {          printf("Enter Employee number & current age: ");          scanf("%d %d", &ret\_emp[i][1], &ret\_emp[i][0]);          ret\_emp[i][0] = cy + ra - ret\_emp[i][0];      }      selectionsort(n,ret\_emp);      return 0;  } |
| **RESULT:** | |
| **Program 3** | |
| **PROBLEM STATEMENT:** | A matrix is singular if and only if its determinant is 0. Write a function which determines whether a matrix is singular or not. |
| **ALGORITHM:** | 1. START(cofact) 2. For row=0,row++ 3. For col=0,col++ 4. If row!=r and col!=c 5. cof[i][j++] = mat[row][col] 6. if j==n-1 7. j=0 8. i++ 9. Repeat steps 4-8 till col<n 10. Repeat steps 3-9 till row<n 11. STOP 12. START(det\_matrix) 13. If n==1 14. Return det[0][0] 15. Cof[n][n] 16. Sign=1,d=0 17. For j=0,j++ 18. cofact(n,def,cof,0) 19. d += sign\*det[0][j]\*det\_matrix(n-1,cof) 20. sign = -sign 21. repeat steps 7-9 till j<n 22. return d 23. STOP 24. START (main) 25. Input n 26. Input det[n][n] 27. if(det\_matrix(n,det)==0) 28. print “This matrix is singular” 29. else 30. print "This matrix is not singular” 31. STOP |
| **PROGRAM:** | #include<stdio.h>  void print\_mat(int n, int mat[n][n])  {      for (int i=0;i<n;i++)      {          for (int j=0;j<n;j++)              printf("%d ", mat[i][j]);          printf("\n");      }  }  void cofact(int n, int mat[n][n], int cof[n-1][n-1], int r, int c)  {      int i=0,j=0;      for(int row=0;row<n;row++)      {          for(int col=0;col<n;col++)          {              if(row!=r && col!=c)              {                  cof[i][j++] = mat[row][col];                  if(j==n-1)                  {                      j=0;                      i++;                  }              }          }      }  }  int det\_matrix(int n, int det[n][n])  {      int d=0;      if(n==1)          return det[0][0];      int cof[n][n];      int sign=1;      for(int j=0;j<n;j++)      {          cofact(n,det,cof,0,j);          d += sign\*det[0][j]\*det\_matrix(n-1,cof);          sign = -sign;      }      return d;  }  int main()  {      int n,i,j;      printf("Enter dimension of the matrix:\n");      scanf("%d",&n);      int det[n][n];      printf("Enter Elements of Matrix:\n");      for(i=0;i<n;i++)      {          for(j=0;j<n;j++)          scanf("%d",&det[i][j]);      }      printf("Entered Matrix:\n");      print\_mat(n,det);      if(det\_matrix(n,det)==0)          printf("This matrix is singular\n");      else          printf("This matrix is not singular\nDeterminant = %d",det\_matrix(n,det));      return 0;  } |
| **RESULT:** | |
| **CONCLUSION:** | In this experiment, how to create and perform operations on 2D arrays. We learned how to code basic operations on Matrices like Addition, Subtraction, Multiplication and Transpose. We learned how we can sort entire rows of an 2D array and lastly how use of 2D matrices helps solving graphical problems easily. |