SRI VENKATESWARA UNIVERSITY: TIRUPATI

Common to all BCA Honours

General / Data Science/Big Data/Artificial Intelligence/Cloud Computing

II Year III Semester

COURSE6: Mathematical and Statistical Foundation Practical

(w.e.f. 2024-25)

Practical **Credits: 1** **2 hrs/week**

List of Lab Experiments & simple implementation using C language

1) Addition, Subtraction of Matrices.
2) Multiplication of Matrices.
3) Determinant of a Matrix and Inverse of a Matrix.
4) Singular and Non-Singular Matrices.
5) Matrix Inversion Method.
6) Cramer's Rule
7) Rank of a Matrix.
8) Preparation of two – way frequency table
9) Problem on Mean and Median.
10) Empirical relationship between mean, median and mode

1. Addition, Subtraction of Matrices

AIM:

To perform addition and subtraction of two matrices using C.

ALGORITHM:

step1: Start
Step2: Read the values for m,n.
Step3: Read matrix values of a[i][j] and b[i][j]
Step4: Initialize i=1
Step5: If(i<=m) then goto step6
else goto step10 .
step6: Initialize j=1
Step7: if j<=n then goto step8
else
set i=i+1 and goto step5.
Step8: set c[i][j] = a[i][j] + b[i][j]
d[i][j] = a[i][j] - b[i][j]
j=j+1 goto step7 .
Step10: Display the value of c[i][j]
Step11: Display the value of d[i][j]
Step12: Stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
 int a[5][5],b[5][5],c[5][5],d[5][5],i,j,m,n;
 clrscr();
 printf("Enter size of the matrices\n");
 scanf("%d%d",&m,&n);
 printf("Enter Matrix A of order %dX%d\n",m,n);
 for(i=0;i<m;i++)
     for(j=0;j<n;j++)
        scanf("%d",&a[i][j]);
 printf("Enter Matrix B of order %dX%d\n",m,n);
 for(i=0;i<m;i++)
     for(j=0;j<n;j++)
          scanf("%d",&b[i][j]);
 for(i=1;i<=m;i++)
 for(j=1;j<=n;j++)
```

```
 {
   c[i][j]=a[i][j]+b[i][j];
   d[i][j]=a[i][j]-b[i][j];
 }
 printf("A+B=\n");
 for(i=1;i<=m;i++)
 {
  for(j=1;j<=n;j++)
     printf("%d\t",c[i][j]);
  printf("\n");
 }
 printf("A-B=\n");
 for(i=1;i<=m;i++)
 {
  for(j=1;j<=n;j++)
     printf("%d\t",d[i][j]);
  printf("\n");
 }
 getch();
}
```

OUTPUT:

2. Multiplication of Matrices

AIM:

To perform multiplication of two matrices using C

ALGORITHM:

Step1: Start
Step2: read size of matrices m1,n1,m2,n2
Step3: if(n1!=m2)
print "Matrix Multiplication not possible"
and goto step14
else
goto step4
Step4: read matrix values a[i][j] and b[i][j]
Step5: initialize i=1
Step6: if i<=m1 goto step 7 else goto step
Step7: initialize j=1
Step8: if j<=n2 goto step9
else
print new line ,set i=i+1 and goto step6
Step9: c[i][j]=0
Step10:initialize k=1
Step11:if k<=n1
goto step12
else
print c[i][j]
set j=j+1 and goto step8
Step12:c[i][j]=c[i][j]+a[i][k]*b[k][j]
Step13: set k=k+1 and goto step11
Step14: stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
 int a[10][10],b[10][10],c[10][10],m1,n1,m2,n2,i,j,k;
 clrscr();
 printf("Enter size of matrix A\n");
 scanf("%d%d",&m1,&n1);
 printf("Enter size of matrix B\n");
 scanf("%d%d",&m2,&n2);
 if(n1!=m2)
```

```
    printf("Matrix multiplication not possible ");
 else
 {
   printf("Enter Matrix A of order %dX%d\n",m1,n1);
   for(i=1;i<=m1;i++)
      for(j=1;j<=n1;j++)
         scanf("%d",&a[i][j]);
   printf("Enter Matrix B of order %dX%d\n",m2,n2);
   for(i=1;i<=m2;i++)
      for(j=1;j<=n2;j++)
         scanf("%d",&b[i][j]);
   printf("The Product matrix AB is\n");
   for(i=1;i<=m1;i++)
   {
      for(j=1;j<=n2;j++)
      {
         c[i][j]=0;
         for(k=1;k<=n1;k++)
            c[i][j]=c[i][j]+a[i][k]*b[k][j];
         printf("%d\t",c[i][j]);
      }
      printf("\n");
    }
  }
 getch();
}
```

OUTPUT:

3.Determinant of a Matrix and Inverse of a Matrix

AIM:

To find determinant and Inverse of a 3X3 matrix using C.

ALGORITHM:

Step1: start
Step2: set det=0
Step3: read values of a 3X3 matrix for a[i][j]
Step4: initialize i=0
Step5: if i<3 goto step6
else
goto step9
step6: initialize j=0
step7: if j<3
goto step8
else
set i=i+1 and goto step5
step8: det = det+(a[0][i]*(a[1][(i+1)%3]*a[2][(i+2)%3]
-a[1][(i+2)%3]*a[2][(i+1)%3]))
set j=j+1 and goto step7
step9: print det vaue
step10: if det=0
print "Inverse does not exists" and goto step14
else
goto step11
step11: calculate cofactor matrix values
cof[i][j]=((a[(i+1)%3][(j+1)%3]*
a[(i+2)%3][(j+2)%3])
-(a[(i+1)%3][(j+2)%3]*a[(i+2)%3][(j+1)%3]))
step12: calculate adjoint matrix values
adj[i][j]=cof[j][i]
step13: print values of a[i][j]/det
step14: stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
  int a[3][3],i,j;
  float cof[3][3],adj[3][3],det=0,t;
```

```
  clrscr();
  printf("Enter a 3X3 matrix:\n");
  for(i=0;i<3;i++)
      for(j=0;j<3;j++)
           scanf("%d",&a[i][j]);
  for(i=0;i<3;i++)
      det = det + (a[0][i]*(a[1][(i+1)%3]*a[2][(i+2)%3] -
a[1][(i+2)%3]*a[2][(i+1)%3]));
  printf("Determinant of given matrix=%f",det);
  if(det==0)
     printf("\n Inverse does not exists for given matrix");
  else
  {
     printf("\n Inverse of matrix is: \n\n");
     for(i=0;i<3;i++)
          for(j=0;j<3;j++)
            cof[i][j]=((a[(i+1)%3][(j+1)%3]
*a[(i+2)%3][(j+2)%3])-
(a[(i+1)%3][(j+2)%3]*a[(i+2)%3][(j+1)%3]));
     for(i=0;i<3;i++)
     {
        for(j=0;j<3;j++)
        {
           adj[i][j]=cof[j][i];
           t=adj[i][j]/det;
           printf("%0.2f\t",t);        }
        printf("\n");
     }
  }
getch();
}
```

OUTPUT:

4. Singular and Non-Singular Matrices

AIM: To check whether a given 3X3 matrix is Singular or Non-Singular using C

ALGORITHM:

Step1: start
Step2: set det=0
Step3: read values of a 3X3 matrix for a[i][j]
Step4: initialize i=0
Step5: if i<3
goto step6
else
goto step9
step6: initialize j=0
step7: if j<3
goto step8
else
set i=i+1 and goto step5
step8: det = det+(a[0][i]*(a[1][(i+1)%3]*a[2][(i+2)%3] - a[1][(i+2)%3]*a[2][(i+1)%3]))
set j=j+1 and goto step7
step9: print det vaue
step10: if det=0
print "Given matrix is singular"
else
print "Given matrix is Non-singular"
step11: Stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
  int a[3][3],i,j,det=0;
  clrscr();
  printf("Enter a 3X3 matrix:\n");
  for(i=0;i<3;i++)
      for(j=0;j<3;j++)
           scanf("%d",&a[i][j]);
  for(i=0;i<3;i++)
```

```
     det = det + (a[0][i]*(a[1][(i+1)%3]*a[2][(i+2)%3] -
a[1][(i+2)%3]*a[2][(i+1)%3]));
  printf("Determinant of given matrix=%d",det);
  if(det==0)
    printf("\n Given Matrix is singular");
  else
    printf("|n Given Matrix is Non-Singular");
  getch();
}
```

OUTPUT:

5. Matrix Inversion Method

AIM: To solve a given system of linear equations with 3 unknowns by Matrix Inversion method using C

ALGORITHM:

Step1: start
Step2: set det=0
Step3: read values of a coefficient matrix a[i][j]and
constant matrix b[i][j]
Step4: initialize i=0
Step5: if i<3
goto step6
else
goto step9
step6: initialize j=0
step7: if j<3 goto step8
else
set i=i+1 and goto step5
step8: det = det+(a[0][i]*(a[1][(i+1)%3]*a[2][(i+2)%3]
-a[1][(i+2)%3]*a[2][(i+1)%3]))
set j=j+1 and goto step7
step9: if det=0
print "Solution does not exists"
and goto step14
else
goto step11
step11: calculate cofactor matrix values
cof[i][j]=((a[(i+1)%3][(j+1)%3] *
a[(i+2)%3][(j+2)%3])-
(a[(i+1)%3][(j+2)%3]*a[(i+2)%3][(j+1)%3]))
step12: calculate adjoint matrix values
adj[i][j]=cof[j][i]
step13: calculate inverse matrix inv[i][j]= a[i][j]/det
Step14: initialize i=0
Step15: if i<3 goto step 16 else goto step
Step16: initialize j=1
Step17: if j<1
goto step18
else
print new line ,set i=i+1 and goto step15
Step18: t[i][j]=0

Step19: initialize k=0
Step20: if k<3
goto step21
else
print t[i][j]
set j=j+1 and goto step17
Step21: t[i][j]=t[i][j]+inv[i][k]*b[k][j]
Step22: set k=k+1 and goto step20
Step23: stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
  int a[3][3],b[3][1],i,j,k;
  float cof[3][3],inv[3][3],t[3][1],det=0;
  clrscr();
  printf("Enter coefficient matrix:\n");
  for(i=0;i<3;i++)
      for(j=0;j<3;j++)
           scanf("%d",&a[i][j]);
  printf("Enter constant matrix:\n");
  for(i=0;i<3;i++)
      for(j=0;j<1;j++)
           scanf("%d",&b[i][j]);
  for(i=0;i<3;i++)
      det = det + (a[0][i]*(a[1][(i+1)%3]*a[2][(i+2)%3] -
                  a[1][(i+2)%3]*a[2][(i+1)%3]));
  if(det==0)
    printf("Solution does not exists");
  else
  {
   printf("\nThe values of x, y, z are:\n");
   for(i=0;i<3;i++)
      for(j=0;j<3;j++)
         cof[i][j]=((a[(i+1)%3][(j+1)%3]*
             a[(i+2)%3][(j+2)%3])-(a[(i+1)%3][(j+2)%3]*
             a[(i+2)%3][(j+1)%3]));
   for(i=0;i<3;i++)
      for(j=0;j<3;j++)
         inv[i][j]=cof[j][i]/det;
```

```
    for(i=0;i<3;i++)
       for(j=0;j<1;j++)
       {
        t[i][j]=0;
        for(k=0;k<3;k++)
           t[i][j]=t[i][j]+inv[i][k]*b[k][j];
        printf("%.2f",t[i][j]);
        printf("\n");
       }
  }
getch();
}
```

OUTPUT:

6. Cramer's Rule

AIM: To solve a given system of linear equations with 3 unknowns by Matrix Cramer's method using C

PROGRAM:

Step1: start
Step2: define a function det() to calculate determinant
Step3: read values of a coefficient matrix a[i][j]and
set t1[i][j]=t2[i][j]=t3[i][j]=a[i][j]
step4: read values of constant matrix b[i][j]
Step4: calculate d=det(a)
Step5: if d=0
print "Solution does not exist"
else
goto step6
step6: replace column1 values of t1[i][j] with b[i][j]
then d1=det(t1)
step7: replace column2 values of t2[i][j] with b[i][j]
then d2=det(t2)
step8: replace column3 values of t3[i][j] with b[i][j]
then d3=det(t3)
step9: x=d1/d,y=d2/d,z=d3/d
step10: print the values of x,y,z
step11: Stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
float det(int a[3][3])
{
 float d=0.0;
 for(int i=0;i<3;i++)
    d=d+(a[0][i]*(a[1][(i+1)%3]*a[2][(i+2)%3] -
        a[1][(i+2)%3]*a[2][(i+1)%3]));
 return d;
 }


void main()
{
  int a[3][3],b[3][1],t1[3][3],t2[3][3],t3[3][3],i,j;
```

```
  float d, d1, d2, d3;
  clrscr();
  printf("Enter coefficients of variables:\n");
  for(i=0;i<3;i++)
      for(j=0;j<3;j++)
        {
         scanf("%d",&a[i][j]);
         t1[i][j]=t2[i][j]=t3[i][j]=a[i][j];
         }
  printf("Enter constants:\n");
  for(i=0;i<3;i++)
      for(j=0;j<1;j++)
           scanf("%d",&b[i][j]);
  d=det(a);
  if(d==0.0)
    printf("Solution does not exists");
  else
  {
     for(i=0;i<3;i++)
        for(j=0;j<1;j++)
           t1[i][j]=b[i][j];
     d1=det(t1);
     for(i=0;i<3;i++)
        for(j=1;j<2;j++)
           t2[i][j]=b[i][j-1];
     d2=det(t2);
     for(i=0;i<3;i++)
        for(j=2;j<3;j++)
           t3[i][j]=b[i][j-2];
     d3=det(t3);
     printf("\nSolution of given system of linear equations
is:\n");
     printf("x=%0.2f,\t y=%0.2f,\t z=%0.2f",d1/d, d2/d,
     d3/d);
  }
getch();
}
```

OUTPUT:

7. Rank of a Matrix.

AIM: To find the rank of a given 3X3 Matrix by reducing to Echelon form using C

ALGORITHM:

Step1: Initialize the rank to the number of columns.

Step2: Iterate through each row:

- If the leading element (diagonal element) is not zero, make all elements below it in the same column zero by subtracting appropriate multiples of the row.
- If the leading element is zero, check if there is a row below with a non-zero element in the same column. If so, swap the rows.
- If all elements in the column are zero, reduce the rank by 1 and move to the next column.
- The rank is the number of non-zero rows left after the above operations

PROGRAM :

```
#include <stdio.h>
#include<conio.h>
#define R 3
#define C 3
void swap(int mat[R][C],int row1,int row2,int col)
 {
    for (int i = 0; i < col; i++)
    {
        int temp = mat[row1][i];
        mat[row1][i] = mat[row2][i];
        mat[row2][i] = temp;
    }
 }
int rankOfMatrix(int mat[R][C])
{
  int rank = C;
  for(int row=0;row<rank;row++)
  {
    if(mat[row][row])
    {
      for (int col=0;col < R;col++)
      {
```

```
        if (col!=row)
        {
          double mult =(double)mat[col][row]/ mat[row][row];
          for (int i=0;i<rank;i++)
            mat[col][i] -= mult * mat[row][i];
         }
       }
      }
     else
     {
      int reduce=1;
      for(int i=row+1;i<R;i++)
      {
        if (mat[i][row])
        {
           swap(mat,row,i,rank);
           reduce = 0;
           break;
         }
        }
       if(reduce)
       {
         rank--;
         for (int i=0;i<R;i++)
            mat[i][row] = mat[i][rank];
       }
       row--;
      }
     }
     return rank;
 }
void main()
{
  int mat[3][3],i,j,rank;
  clrscr();
  printf("Enter a 3X3 matrix:\n");
  for(i=0;i<3;i++)
      for(j=0;j<3;j++)
           scanf("%d",&mat[i][j]);
  rank = rankOfMatrix(mat);
  printf("\nRank of the matrix is: %d\n", rank);
  getch();
}
```

OUTPUT:

8. Arrangement of two-way frequency distribution table

PROBLEM: Prepare a two-way frequency distribution for the following data given by ages in years and Blood Pressure using class Intervals (45, 141); (26, 130); (62, 150); (28, 114); (55, 138); (36, 120); (48, 142); (40, 139); (28, 105); (32, 135); (31, 153); (37, 151); (59, 149); (50, 151); (48, 121); (47, 126); (33, 131); (42, 154); (49, 151); (34, 118).

PROCEDURE:

Let, X = Age in years, Y = Blood Pressure.

Bivariate Frequency Distribution is to be prepared by taking class intervals for X as 25-35, 35-45, 45-55, etc., and for Y as 105-120, 120-135, etc. using 'four and cross method'

CALCULATION:

Bivariate Frequency Distribution

Blood Pressure (Y)/ Age in years (X)	**105-120**	**120-135**	**135-150**	**150-165**	**Total f_x**
25-35	|||	||	|	|	**7**
35-45		|	|	||	**4**
45-55		||	||	||	**6**
55-65			||	|	**3**
Total f_Y	**3**	**5**	**6**	**6**	**20**

9. Problem on Mean and Median

AIM: To find mean and Median of a list of n values using C

ALGORITHM:

step1: start
step2: read n
step3: read n values a[i]
step4: initialize i=1
step5: if i<=n
goto step6
else
goto step10
step6: initialize j=i+1
step7: if j<=n goto step8
else
set i=i+1 and goto step5
step8: if a[i]>a[j]
t=a[i]
a[i]=a[j]
a[j]=t
step9: set j=j+1 and goto step7
step10: set sum=0
step11: initialize i=1
step12: if i<=n goto step13
else goto step14
step13: set sum=sum+a[i] and i=i+1
step14: print mean=sum/n
step15: if(n%2==0)
print Median=(a[n/2]+a[(n/2)+1])/2
else
printf Median=a[(n+1)/2]
step16: Stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>

void main()
{
  int n,i,j;
  float a[50],t,sum=0.0;
```

```
  clrscr();
  printf("How many values:\n");
  scanf("%d",&n);
  printf("Enter %d values:\n",n);
  for(i=1;i<=n;i++)
    scanf("%f",&a[i]);
  for(i=1;i<=n;i++)
  {
   for(j=i+1;j<=n;j++)
   {
     if(a[i]>a[j])
     {
       t=a[i];
       a[i]=a[j];
       a[j]=t;
     }
   }
  }
 for(i=1;i<=n;i++)
  sum=sum+a[i];
 printf("\nMean=%0.2f",sum/n);
 if(n%2==0)
   printf("\nMedian=%0.2f",(a[n/2]+a[(n/2)+1])/2);
 else
   printf("\nMedian=%0.2f",a[(n+1)/2]);
getch();
}
```

OUTPUT:

10. Empirical relationship between mean, median and mode

AIM: To Establish the empirical relationship among mean,median and mode by using C

ALGORITHM:

step1: start
step2: print "1.Mean 2.Median 3.Mode"
step3: read choice as d
step4: switch(d)
case 1: read median and Mode
print mean=(3*median-mode)/2
goto step5
case 2: read mean and Mode
print median=(mode+2*mean)/3
goto step5
case 3: read median and Mode
print mode=3*median-2*mean
goto step5
case default: print "Wrong choice" and goto step2
step5:Stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
 float mean,median,mode;
 int d;
 clrscr();
 nxt:printf("Enter your choice
     \n1.Mean\n2.Median\n3.Mode\n");
 scanf("%d",&d);
 switch(d)
 {
  case 1:printf("Enter median and Mode\n");
         scanf("%f%f",&median,&mode);
         printf("Mean=%f",(3*median-mode)/2);
         break;
  case 2:printf("Enter Mean and Mode\n");
```

```
        scanf("%f%f",&mean,&mode);
        printf("Median=%f",(mode+2*mean)/3);
        break;
  case 3:printf("Enter Mean and Median\n");
        scanf("%f%f",&mean,&median);
        printf("Mode=%f",3*median-2*mean);
        break;
  default:printf("Wrong Choice\n");
        goto nxt;
 }
}
```

OUTPUT: