# Experiment - 1

C प्रोग्राम को लिखकर निम्न रैखिक समीकरणों के निकाय को गॉस विलोपन विधि द्वारा हल कीजिए ?

$$2x + 8y + 2z = 14$$
$$x + 6y - z = 13$$
$$2x - y + 2z = 5$$

गॉस विलोपन विधि (Gauss Elimination method):-

इस विधि के अन्तर्गत दिए गए रैखीय बीजीय समीकरणों के निकाय का सवर्धिन्त आव्यूह (Augmented matrix) में प्रारंभिक पंक्ति रूपान्तरण (Elementary row operations) के द्वारा संवर्धित आव्यूह में गुणांक आव्यूह (cofficient matrix) को ऊपरी त्रिभुजाकार आव्यूह (upper triangular matrix) में परिवर्तित करते हैं। तत्पश्चात परच् प्रतिस्थापन विधि द्वारा अज्ञात चरों का मान ज्ञात करते हैं।

अब हम C-प्रोग्राम के द्वारा (कम्प्यूटर की सहायता से) उपरोक्त समीकरण निकाय का हल ज्ञात करते हैं।

```
/*C- Program For Gauss- Elimination Method*/
# include <stdio.h>
# include <conio.h>
void main ( )
{
int i, j, k, n;
Float a[10][10], x [10], Sum, temp;
Print F("Enter Number of System of equations\n\n");
Scan("%d", &n);
Print F("Number of Equations are = %d\n",n);
Clrscr ( );
```

```
Print f("Enter coefficients of linear equations/
                coefficients of augmented matrix\n\n");
For(i=0; i<n; i++)
{
 For (i=0; j<n+1; j++)
    scanf ("%f', &a[i][j]);
 }
 /* Calculate upper triangular matrix */
 For (i=1; i<n; i++)
 {
      temp = 0.0;
   For (j=i+1; j<n; j++)
 {
        temp = a[j][i]/a[i][i];
   For (k=i; k<=n+1; k++)
           a[j][k]=a[j][k]-(temp* a[i][k]);
  }
  }
 /* Print upper triangular matrix */
 Print f("\n upper Triangular matrix is; \n\n");
   For(i=1; i<n; i++)
```

[C भाषा का संख्यात्मक विधि में अनुप्रयोग]

```
 {
  For(j=1; j<=n+1; j++)
   Print f ("% 8.21", a[i][j]);
   Print f ("\n");
  }
 /* evaluate unknowns by back substitution */
```

```
x[n] = a[n][n+1]/a[n][n];
For (i=n-1; i>=1; i--)
{
 Sum = 0.0;
 For (j = i+1; j<n; j++)
  {
 Sum = Sum + (a[i][j] * x[j]);
 }
  x[i] = (a[i][n+1] - Sum/a[i][i]
 }
  Printf ("n\Solution of the equation is :\n\n");
   For (i=1; i<=n; i++)
      Printf ("x[%d] = % 8.3F\n", i x[i]);
   getch ( );
  }
```

Input: Enter Number of System of equations

Number of equations are = 3

Enter coefficients of linear equations/coefficients of augmented matrix.

| | | | |
|---|---|---|---|
| 2 | 8 | 2 | 14 |
| 1 | 6 | -1 | 13 |
| 2 | -1 | 2 | 5 |

Output:

Upper Triangular matrix is;

| | | | |
|---|---|---|---|
| 2.00 | 8.00 | 2.00 | 14.00 |
| 0.00 | 2.00 | -2.00 | 6.00 |
| 0.00 | 0.00 | -9.00 | 18.00 |

Solution of the equation is:

x[1] = 5.00
x[2] = 1.00
x[3] = -2.00

## Experiment - 2

उपरोक्त C-प्रोग्राम द्वारा निम्न समीकरण निकाय का गॉस विलोपन विधि द्वारा हल कीजिए :

$$8x_1 + 4x_2 + 2x_3 = 24$$
$$4x_1 + 10x_2 + 5x_3 + 4x_4 = 32$$
$$2x_1 + 5x_2 + 6.5x_3 + 4x_4 = 26$$
$$4x_2 + 4x_3 + 9x_4 = 21$$

Input: Enter Number of system of equations,
Number of Equations are = 4
Enter coefficients of linear equations/
coefficients of augmented matrix:

| | | | | |
|---|---|---|---|---|
| 8 | 4 | 2 | 0 | 24 |
| 4 | 10 | 5 | 4 | 32 |
| 2 | 5 | 6.5 | 4 | 26 |
| 0 | 4 | 4 | 9 | 21 |

Output:
Upper Triangular matrix is :

| | | | | |
|---|---|---|---|---|
| 8.00 | 4.00 | 2.00 | 0.00 | 24.00 |
| 0.00 | 8.00 | 4.00 | 4.00 | 20.00 |
| 0.00 | 0.00 | 4.00 | 2.00 | 10.00 |
| 0.00 | 0.00 | 0.00 | 6.00 | 6.00 |

solution of the equation is :

X[1] = 2.00    X[2] = 1.00

X[3] = 2.00    X[4] = 1.00

# Experiment - 3

C-प्रोग्राम को लिखकर, द्विभाजन विधि का प्रयोग करते हुए समीकरण $x^3-2x-5=0$ का मूल जो अन्तराल [2, 3] मे स्थित हैं दशमलव के तीन स्थान तक ज्ञात कीजिए।

हल:- हम जानते हैं कि यदि फलन $f(x)$ दो वास्तविक मानों $x_0, x_1$ के मध्य संतत हो तथा $F(x_0)$ और $F(x_1)$ विपरीत चिन्ह के हों अर्थात् $F(x_0).F(x_1)<0$, तो $x_0$ तथा $x_1$ के मध्य $F(x)=0$ का एक मूल अवश्य होगा। यदि माना कि $F(x_0)$ का मान ऋण तथा $F(x_1)$ का मान धन चिह्न हैं। अब अन्तराल $[x_0, x_1]$ को उनके मध्य स्थित बिन्दु $x_2 = \frac{x_0+x_1}{2}$ द्वारा दो समान भागों मे विभाजित किया। यदि $F(x_2)=0$ है, तब $x_2$, $F(x)=0$ का सही मूल है। परन्तु यदि $F(x_2)\neq 0$ तो $F(x)=0$ का मूल $[x_0, x_2]$ या $[x_2, x_1]$ में किसी एक में विद्यमान होगा। यदि $F(x_2)$ का मान धनात्मक हैं, तो मूल $[x_0, x_2]$ में स्थित होगा। क्योंकि $F(x_0).F(x_2)<0$ होगा और यदि $F(x_2)$ का मान ऋणात्मक हैं तो मूल $[x_2, x_1]$ में स्थित होगा क्योंकि $F(x_2).F(x_1)<0$ होगा।

इसी प्रकार आगें भी जिस अन्तराल में मूलबिन्दु हों उसके दो समान भागों में विभाजित करतें जातें हैं जब तक कि मूल का वांछित शुद्धता तक मान ज्ञात नहीं हो जाए।

द्विभाजन विधि द्वारा समी. $f(x)=0$ के मूल ज्ञात करनें का C-प्रोग्राम इस प्रकार हैं:

```
/* BISECTION METHOD*/
# include < stdio.h >
# include < conio.h >
# include < math.h >
# define F(x) x*x*x - 2.0*x - 5.0 /* write given
                                     equation*/
void main ( )
{
Float x, x0, x1, x2, y0, y1, y2, e;
clser ( );
PrintF ("Input two Initial starting roots \n");
```

```
Scanf ("%f", &x0, &x1);
Printf ("Input the tolerance value \n");
Scanf ("%f", & e);
y(o) = F(x0);
y0 = F(x0);
y1 = F(x1);
if (y0 * y1 > 0.0)
{
Printf ("\n Initial values are unsuitable \n");
}
else
{
while ((Fabs (x1 - x0) / x1) > e)
{
  x2 = (x0 + x1) / 2;
  y2 = F(x2);
if (y0 * y2 > 0)
  x0 = x2;
else
  x1 = x2;
}
Printf ("Solution converges to a root \n");
Printf ("\n Root of the given equation is % 8.3f",
                                           x2);
} getch ( );
}
```

Input:

Input two Initial Starting roots

2, 3

Input the tolerance value

0.001

Output:

Solution converge to a root

Root of the given equation is = 2.095.

# Experiment - 4

उपरोक्त C-प्रोग्राम द्वारा समीकरण $x^4 - x - 10 = 0$ का मूल द्विभाजन द्वारा ज्ञात कीजिए।

**Input:**

Input Two Initial Starting roots
1, 2

Input the tolerance value 0.0001

**Output:**

Solution converges to a root

Roots of the given equation is = 1.856

# Experiment - 5

C-प्रोग्राम के द्वारा समी. $x^2 + x - 1 = 0$ का एक मूल जो अन्तराल [0, 1] में स्थित है मिथ्या स्थिति विधि की सहायता से दशमलव के तीन स्थानों तक शुद्ध ज्ञात कीजिए।

हल:- मिथ्या स्थिति विधि का पुनरावृत्ति सूत्र निम्न है -

$$x_{n+1} = \frac{x_{n-1} F(x_n) - x_n F(x_{n-1})}{F(x_n) - F(x_{n-1})}, \quad n = 1, 2, 3 \ldots$$

C-प्रोग्राम इस प्रकार है:-

```
/* Root of an Equation by Regula-Falsi method*/
# include < stdio.h>
# include < conio.h>
# include <math.h>
# define F(X),, X*X+X-1/ *write given equation*/
void main ( )
{
 Float x0, x1, x2, F0, F1, e;
 int max_iteration;
 clrscr ();
 Printf ("Input Two Initial Roots, maximum
                iteration and tolerance value:");
 Scanf ("%F%F%F", &x0, &x1, &max_iteration,
                &e);
 F(0) = F(x0)
 F0 = F(x0);
 F1 = F(x1);
 if (F0*F1 > 0.0)
 {
  Printf ("/\ Initial roots are unsuitable \n");
 }
```

```
else
{
 For(i=1; i<=max_iteration; i++)
 {
  x2=(x0*F1-x1* F0)/(F1-F0);
  F2 = F(x2)
  if (Eabs(F2)<e)
 {
   PrintF ("\ Solution is converge.\n");
PrintF ("\ Root of the given equations is = % 8.4 % \n",x2);
max_iteration=i;
   }
   else
   {
    if (F2*F0<0)
    {
     x1 = x2;
     F1 = F2;
    }
    else
    {
     X0 = X2;
     F0 = F2;
    }
   }
  }
 }
 getch();
}
```

Input:

Input Two initial Roots, maximum iterations and tolerance value;

0 1 10 0.0001

Output:

Solution is converge

Roots of the given equation is = 0.618.

# Experiment - 6

उपरोक्त C-प्रोग्राम की सहायता से मिथ्या स्थिति विधि द्वारा समीकरण $x^6 - x^4 - x^3 - 3 = 0$ का 1.5 तथा 1.6 के मध्य वाला वास्तविक मूल चार दशमलव स्थानों तक ज्ञात कीजिए।

Input:

Input Two Initial Roots, maximum iterations and tolerance value;

1.5 1.6 10 0.00001

Output:

Solution in converge

Root of the given equation is = 1.5018

# Experiment - 7.

C-प्रोग्राम के द्वारा समीकरण $x^3-3x-5=0$ का मूल जो अन्तराल [2,3] मे स्थित हैं न्यूटन-रेफसन विधि द्वारा दशमलव के चार स्थानो तक शुद्ध कीजिए ?

हल:- न्यूटन-रेफसन विधि का पुनरावृति सूत्र:-

$$x_{n+1} = x_n - \frac{F(x_n)}{F'(x_n)}$$

जहाँ $F'(x_n)$, $F(x_n)$ का $x_n$ के सापेक्ष प्रथम अवकलन है।

C-प्रोग्राम:

```
/* Newton - Raphson Method */
# include < stdio.h >
# include < conio.h >
# include < math.h >
# define F(x) x*x*x-3*x-5 /* Given Equation */
# define dF(x) 3*x*x-3* Differential of
                                   Equation */
Void main(*)
{
 long Float x0, F0, F1, Y1, x, e;
 long int i, maxiter;
 clrscr( );
PrintF {"\ Input Initial root, max. iterations
                    and tolerance value: \n")
ScanF ("% lf, %ld %lf" & x0, & maxiter, & e);
For ( i=1; i<=maxiter; i++)
 {
   F0 = F(x0);
   F1 = dF(x0);
   x1 = x0 - (F0/F1);
```

```
if ((Fabs(x1-x0)/x1)<e)
{
 PrintF("\Solution converges to a roots \n");
 PrintF("Number of Iteration = %ld\n", i);
 PrintF("\Root of the given equation is = %8.4lf", x1);
 i=maxiter;
}
else
  {
    x0=x1;
  }
}
 getch( );
}
```

Input:

Input Initial root, max. iterations and tolerance value:

2 10 0.00001

Output:

Solution converges to a root

Number of Iterations = 4

Root of the given equation is = 2.2790.

# Experiment - 8

उपरोक्त C-प्रोग्राम की सहायता से समीकरण $x^3+x^2-1=0$ के मूल का चार दशमलव स्थान तक मान न्यूटन-रेफसन विधि द्वारा ज्ञात कीजिए।

Input:

Input Initial root, max, iterations and tolerance Value:

0.5 10 0.00001

Output:

Solution converges to a root

Number of Iteration = 5

Root of the given equation is = 0.7548

# Experiment - 9.

C-प्रोग्राम की सहायता से आयलर विधी का प्रयोग कर प्रारम्भिक मान समस्या $\frac{dy}{dx} = \frac{y-x}{y+x}$, $y(0) = 1$ का $x = 0.1$ पर $y$ का मान ज्ञात कीजिए, जहाँ पद-अं. $h = 0.02$ है।

हल:- यहाँ $h = 0.02$, $n = \frac{0.1 - 0}{0.02} = 5$, $x_0 = 0$, $y_0 = 1$

तथा $\frac{dy}{dx} = F(x, y) = \frac{y-x}{y+x}$

आॅयलर विधि का सूत्र :

$y_{n+1} = y_n + hF(x_n, y_n)$, $n = 0, 1, 2, \ldots$

आॅयलर विधि का C-प्रोग्राम :

```
/* Euler's method */
# include < Studio.h>
# include < conio.h>
# define F(x,y) (y-x)/(y+x) /* R.H.S. of the
                                equation */
Void main( )
{
  Float x, y, h, xn;
  int i = 0, n;
  clrscr ( );
  printF("Enter the initial values of x and y: ");
  ScanF ("%F %F", & x & y );
  Print F("\n Enter x at which y is required:");
  ScanF("%F, & xn);
  PrintF ("\n Enter Step Size, h: ");
  ScanF ("%F", & h );
```

```
n = (int)((xn - x)/h + 0.5);
Printf ("\n\n Step   x    y \n\n").
while (x < xn)
{
  y = y + h* f(x, y);
  x = x + h;
  i = i+1;
  Printf ("%3d %10.4f %10.4\n", i, x, y);
}
Print ("\n Value of y at x = %f is %f\n\n", x, y);
getch ( );
}
```

Input:

Enter the initial values of x and y : 0, 1

Enter x at which y is required : 0.1

Enter step size, h = 0.02

Output:

| Step | x | y |
|---|---|---|
| 1 | 0.020 | 1.020 |
| 2 | 0.040 | 1.0392 |

| Step | x | y |
|---|---|---|
| 3 | 0.060 | 1.0577 |
| 4 | 0.080 | 1.0756 |
| 5 | 0.10 | 1.0928 |

Value of y at x = 0.10000 is 1.0928.

# Experiment - 10.

उपरोक्त C-प्रोग्राम के द्वारा पद लं. 0.1 लेते हुए ऑयलर विधि का प्रयोग कर निम्न समी. से $y(0.5)$ का मान ज्ञात कीजिए :

$$\frac{dy}{dx} = x^2 + y^2, \quad y(0) = 0$$

हल: Input:

Enter the initial values of x and y:

0.0 0.0

Enter x at which y is required: 0.5

Enter step-size, h = 0.1

Output:

| Step | X | Y |
|---|---|---|
| 1 | 0.1 | 0.00 |
| 2 | 0.2 | 0.001 |
| 3 | 0.3 | 0.005 |
| 4 | 0.4 | 0.014 |
| 5 | 0.5 | 0.0302 |

Value of y at x = 0.50000 is 0.0302.

## Experiment - 11.

C-प्रोग्राम को लिखते हुए रूंगे-कुट्टा विधि (चतुर्थ कोटि) का प्रयोग कर $x = 1.5$ पर $y$ का सन्निकट मान ज्ञात कीजिए, यदि $\frac{dy}{dx} = xy^2 - y/x$ जबकि $x = 1$ पर $y = 1$ एवं पद-लम्बाई 0.1 है।

हल:- यहाँ $f(x, y) = xy^2 - y/x$,

$x_0 = 1$, $y_0 = 1$ तथा $h = 0.1$

रूंगे-कुट्टा विधि (चतुर्थ कोटि) का सूत्र:-

$$k_1 = hf(x_n, y_n)$$

$$k_2 = hf\left(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right)$$

$$k_3 = hf\left(x_n + \frac{h}{2}, y_n + \frac{k_2}{2}\right)$$

$$k_4 = hf(x_n + h, y_n + k_3)$$

$$k = \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$y_{n+1} = y_n + k, \quad n = 0, 1, 2, 3 \ldots$$

रूंगे-कुट्टा विधि का C-प्रोग्राम:

```
/* Runge-kutta methed of fourth order */
# include <studio.h>
# include <canio.h>
vaidmain ( )
{
 Float F(Float x, Float y); /* Function
                               Prototype */
```

```
Float x0, y0, h, xn, k1, k2, k3, k4, k; x, y:
int i, n:
clrscr ( );
Printf ("Enter the initial values of x and y:");
Scanf ("%F %F", & x0, & y0);
Printf ("\n Enter x at which y is required:");
Scanf ("% F", & xn);
Printf ("\n Enter step Size :");
Scanf ("% F", & h);
n(int) = ((xn - x0)/h + 0.5);
x = x0;
y = y0;
Print ("\n\n step x    y \n\n");
For (i=1; i<n; i++)
{
   k1 = h*F(x,y); /* Function call */
   k2 = h*F(x+h/2.0, y+k1/2.0); /* Function call */
   k3 = h*F(x+h/2.0, y + k2/2.0); /* Function call */
   k4 = h*F (x+h, y+k3); /* Function call */
   k = (k1+2.0* (k2 + k3) + k4)/6.0;
   x = x + h
   y = y + k;
   PrintF ("%3d %10.3F % 10.3 F\n", i, x, y);
}
Print F ("\n value of y at x = %F is %F\n\n", x, y);
getch ( );
}
```

```
/* Function definition F( )*/
Float F (Float x, Float y)
{
    return(x*y*y - y/x)/* R.H.S. of the given
                                equation */
}
```

Input:

Enter the initial values of x and y: 1, 1

Enter x at which y is required: 1.5

Enter step size: 0.1

Output:

| Step | x | y |
|---|---|---|
| 1 | 1.1 | 1.0101 |
| 2 | 1.2 | 1.0417 |
| 3 | 1.3 | 1.0989 |
| 4 | 1.4 | 1.1905 |
| 5 | 1.5 | 1.3333 |

Value of y at x = 1.5000 is 1.3334.

# Experiment-12.

उपरोक्त C-प्रोग्राम द्वारा रुंगे-कुट्टा विधि का प्रयोग से समीकरण $\frac{dy}{dx} = x + y$, जहाँ $y(0) = 1$ को हल कर $x = 0.1$ से $x = 0.4$ तक का मान ज्ञात कीजिए जबकि $h = 0.1$ है।

हल:- Input:

Enter the initial values of x and y : 0,1

Enter x at which y is required : 0.4

Enter step size : 0.1

Output:

| Step | x | y |
|---|---|---|
| 1 | 0.1 | 1.1103 |
| 2 | 0.2 | 1.2428 |
| 3 | 0.3 | 1.3997 |
| 4 | 0.4 | 1.5836 |

Value of y at x = 0.4000 is 1.5836.

# Experiment - 13.

C-प्रोग्राम को लिखते हुए समाकलन $\int_0^6 \frac{dx}{1+x^2}$ का मान ट्रेपिजोइडल नियम द्वारा ज्ञात कीजिए ?

हल: ट्रेपिजोइडल नियम का सूत्र:

$$\int_{x_0}^{x_n} y\,dx = h\left[\frac{1}{2}(y_0 + y_n) + (y_1 + y_2 + y_3 + \cdots + y_{n-1})\right]$$

where, $h = \frac{x_n - x_0}{n}$ या $n = \frac{x_n - x_0}{h}$

ट्रेपिजोइडल का C-प्रोग्राम:

```
/* Trapezoidal Rule */
# include < studio.h >
# include < conio.h >
Void main ( )
{
Float F(Float); /* Function Prototype */
Float x0, xn, h, Sum, result;
int i, n;
clrscr ( );
Print (" Enter the lower and upper limits of
                                  Integral: ");
Scanf ("%F %F", & x0, & xn);
PrintF ("\n Enter the segment width: ");
ScanF ("%F", &h);
  n = int (xn - x0) / h;
Sum = (F (x0) + F (xn)) / 2.0 /* Function call */
```

```
For (i=1; i<n; i++)
{
  sum = sum + F(x0 + i*h); /* Function call*/
}
result = sum*h;
printF("\n\n Value of integral using trapezoidal
                    rule is = %F\n\n", result);
getch();
}
/* Function definition F() */
Float F(Float x)
{
  return (1.0/(1.0+x*x)); /* Integrand */
}
```

Input:

Enter the lower and upper limits of
Integral : 0.0 6.0

Enter the segment width : 1.0

Output:

Value of integral using trapezoidal rule
is = 1.410799

# Experiment - 14.

उपरोक्त C-प्रोग्राम द्वारा समाकलन $\int_0^5 \frac{dx}{4x+5}$ का मान ट्रेपिज़ोइडल नियम द्वारा ज्ञात कीजिए ?

हल: Input:

Enter the lower and upper limits of Integral:

0.0 5.0

Enter the segment width : 0.5

Output:

Value of integral using trapezoidal rule is = 0.405510

# Experiment- 15.

C-प्रोग्राम की सहायता से सिम्पसन का एक तिहाई (1/3) नियम द्वारा $\int_0^6 \frac{dx}{1+x^2}$ का मान ज्ञात कीजिए ?

हल:- सिम्पसन का एक -तिहाई नियम का सूत्र :

$$\int_{x_0}^{x_n} y\,dx = \frac{h}{3}\left[(y_0 + y_n) + 2(y_2 + y_4 + y_6 + \ldots + y_{n-2}) + 4(y_1 + y_3 + y_5 + \ldots + y_{n-1})\right]$$

जहाँ n एक सम संख्या है तथा $h = (x_n - x_0)/n$

सिम्पसन का एक -तिहाई नियम का C- प्रोग्राम:

```
/* Simpson's (1/3) Rule */
# include < Studio.h>
# include < canio.h>
Void main ( )
{
Float F(Float x); /* Function prototype */
Float a, b, h Sum = 0.0, result;
int i, n;
clrscr ( );
PrintF ("Enter lower limit of the integral:");
ScanF ("% F', & a);
PrintF ("\n Enter Upper limit of the integral:");
ScanF ("% F', & b);
Print F ("\n Enter the number of Segments:");
ScanF ("% d", & n);
```

```
if (n% 2 != 0)
    Print ("\n Number of Segments not even
                              number \n");
else
{
    h = (b-a) / n;
  Sum = F(a) + F(b) /* Function call */
  For (i = 1; i < n; i++)
  {
    if (i % 2 == 0)
     Sum = Sum + 2.0 * F(a + i*h); /* Function call */
    else
     sum = Sum + 4.0 * F(a + i * h); /* Function call */
  }
  result = (h/3.0) * sum;
  PrintF ("\n\n Value of integral is = % F\n\n", result);
}
  getch ( );
}
 /* Function definition F( ) */
  Float F ( Float x )
 {
   return (1.0/(1.0 + x * x)); /* Integrand */
 }
```

Input:

Enter lower limit of the integral : 0.0

Enter upper limit of the integral : 6.0

Enter the number of segments : 6

Output:

Value of integral is = 1.366174

# Experiment - 16.

उपरोक्त C- प्रोग्राम द्वारा समाकलन $\int_0^3 \frac{x^2}{1+x^3} dx$ का मान सिम्पसन एक-तिहाई ($1/3$) नियम द्वारा ज्ञात कीजिए।

हल:-

Input:

Enter lower limit of the integral : 0.0

Enter upper limit of the integral : 3.0

Enter the number of segments : 12

Output:

Value of integral is = 1.110716.