****

**East West University**

**CSE225 (Numerical Methods), Section: 1**

**Spring 2017**

**Project Report**

***Project Title:* Equation Solver**

-Triangular Factorisation Methods [LU].

***Submitted To:***

**Tamanna Yasmin**

Lecturer,

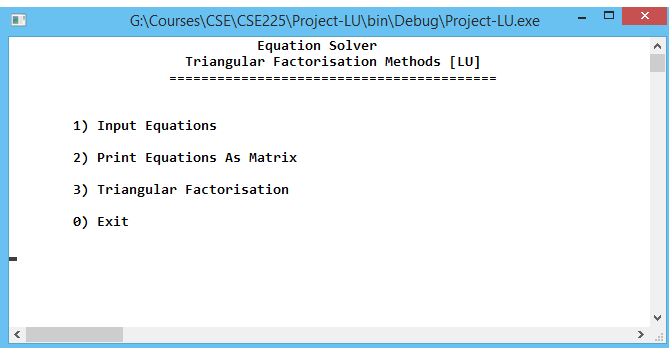
Department of Computer Science and Engineering.

***Submitted By:***

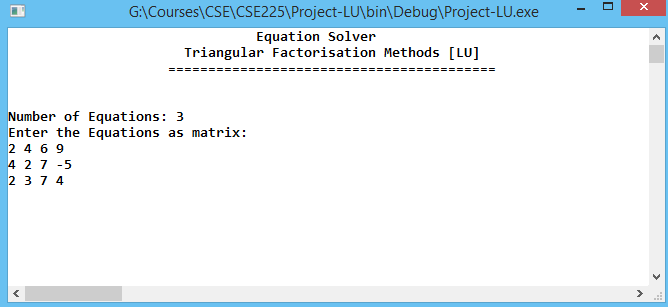
1. **Samia Bari**  
   ID: 2011-3-60-027
2. **Shaykh Siddique**  
   ID: 2016-1-60-053
3. **Nabil Al Raian**   
   ID: 2015-2-60-081

**Introduction:** In our project we have tried to build up a software to calculate the values of linear equations by using **Triangular Factorization Method** of LU decomposition. The **Triangular Factorization Method** isone of the direct solution method of linear equations. In the **Triangular Factorization Method** we used the **Dolittle Algorithm** for the LU decomposition.

**Program Details:** We are up to make a software that calculate and show the result of a linear equations using the LU decomposition method. Firstly we have 4 options. The options are shown below:

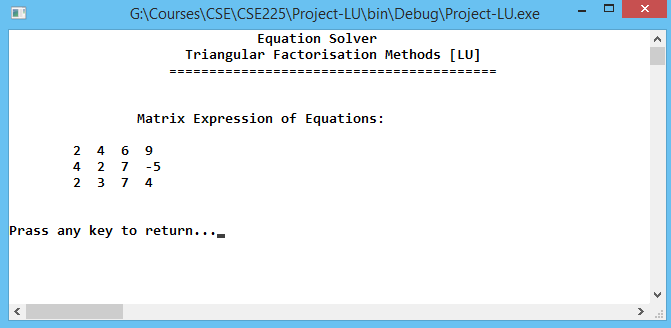


If you select option 1 then there will be another option which will ask the user about how many equations they want to enter.

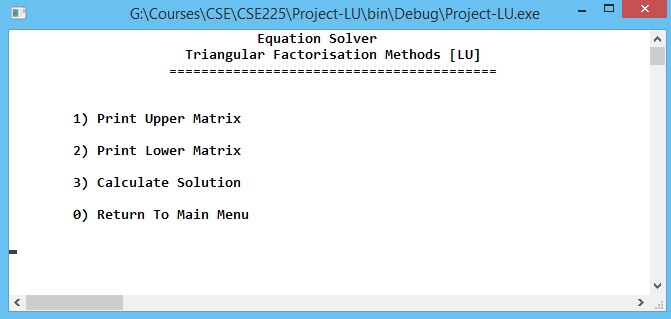


If you put 3 then the program will take input about three equations continuously like this serial:

Now if you select the option 2 it will show the full matrix that was entered previously.



In the option 3 there are four more options for the user. If they want to see the upper matrix they can select the option 1, or if they want to see lower matrix they can select option 2. They can print the value if they select option 3. Else they can return to the main menu by pressing 0.



And lastly there, by pressing 0 they can terminate the program.

Reference: Numerical Methods by E Balagurusamy (Text Book).

Project Codes:

[main.cpp]

1. #include"header.h"
2. #include"variables.h"
3. #include"functions.h"
5. **int** main(){
6. **bool** access=**false**;
7. **while**(**true**){
8. **int** key=nav\_bar();
9. **if**(key==1){
10. //input equation
11. input\_equation();
12. access=**true**;
13. }**else** **if**(key==2){
14. //print\_matrix
15. **if**(!access){
16. cout<<"Please Input Equations...";bl;bl;
17. Sleep(700);
18. **continue**;
19. }
20. print\_matrix();
21. }**else** **if**(key==3){
22. //Triangular Factorisation
23. **if**(!access){
24. cout<<"Please Input Equations...";bl;bl;
25. Sleep(700);
26. **continue**;
27. }
28. cal\_upper\_lower();
29. **while**(**true**){
30. **int** key2=oparate\_menu();
31. **if**(key2==1){
32. //                Print Upper Matrix
33. print\_mat\_U();
34. }**else** **if**(key2==2){
35. //            Print Lower Matrix
36. print\_mat\_L();
37. }**else** **if**(key2==3){
38. //            Calculate Solution
39. calculate\_eqn();
40. }**else** **if**(key2==0){
41. //return to main
42. **break**;
43. }**else**{
44. bl;
45. cout<<"Invalid Input...";bl;
46. Sleep(700);
47. }
48. }
49. }**else** **if**(key==0)
50. **return** 0;
51. **else**{
52. bl;
53. cout<<"Invalid Input...";bl;
54. Sleep(700);
55. }
56. }
57. **return** 0;
58. }

[header.h]

1. #include<bits/stdc++.h>
2. #include<windows.h>
3. #include<conio.h>
5. #define sfi(n) scanf("%d", &n)
6. #define sfl(n) scanf("%lld", &n)
7. #define sfd(n) scanf("%lf", &n)
8. #define sfi2(x, y) scanf("%d %d", &x, &y);
9. #define sfl2(x, y) scanf("%lld %lld", &x, &y);
10. #define sfd2(x, y) scanf("%lf %lf", &x, &y);
11. #define pf printf
12. #define pfc(i) printf("Case %d: ", i)
13. #define pfn(n) printf("%d\n", n)
14. #define pfdn(n) printf("%f\n", n)
15. #define pb push\_back
16. #define bl printf("\n")
17. #define spc putchar(' ')
18. #define fli() freopen("in.txt","r",stdin)
19. #define flo() freopen("out.txt","w",stdout)
20. #define ll long long
21. #define ull unsigned long long
22. #define debug(x) cerr << #x << " : " << x << endl
23. #define find puts("I am here")
24. #define repe(i, n, test) for(int i=n; i<=test; i++)
25. #define rep(i, n, test) for(int i=n; i<test; i++)
26. #define MOD 1000000007
27. #define EPS 1e-7
28. #pragma comment(linker, "/stack:20000000")
29. #define \_CRT\_SECURE\_NO\_WARNINGS
30. #define MAX 1000005
31. **using** **namespace** std;

[variables.h]

1. #define row\_col\_sz 200
3. **double** matrix[row\_col\_sz][row\_col\_sz];
4. **double** mat\_L[row\_col\_sz][row\_col\_sz], mat\_U[row\_col\_sz][row\_col\_sz];
5. **int** eqn\_no=0;
7. **double** X[row\_col\_sz];

[functions.h]

1. **void** bar(){
2. system("cls");
3. cout<<"                               Equation Solver"<<endl;
4. cout<<"                      Triangular Factorisation Methods [LU]"<<endl;
5. cout<<"                    ========================================="<<endl;bl;bl;
6. }
8. **int** nav\_bar(){
9. bar();
10. cout<<"\t1) Input Equations"<<endl;bl;
11. cout<<"\t2) Print Equations As Matrix"<<endl;bl;
12. cout<<"\t3) Triangular Factorisation"<<endl;bl;
13. cout<<"\t0) Exit"<<endl;bl;
14. **int** key;
15. cin>>key;
16. **return** key;
17. }

20. **void** debug\_mat(**double** mat[row\_col\_sz][row\_col\_sz], string str){
21. bar();
22. cout<<"\t"<<str<<" Matrix :"<<endl;bl;
23. rep(ro, 0, eqn\_no){
24. cout<<"\t";
25. rep(co, 0, eqn\_no){
26. pf("%.2f  ", mat[ro][co]);
27. }bl;
28. }bl;bl;
29. }
31. **void** input\_equation(){
32. bar();
33. memset(matrix, 0, **sizeof**(matrix));
34. cout<<"Number of Equations: ";
35. cin>>eqn\_no;
36. //    fli();
37. **if**(eqn\_no){
38. cout<<"Enter the Equations as matrix: \n";
39. rep(ro, 0, eqn\_no){
40. rep(co, 0, eqn\_no+1){
41. cin>>matrix[ro][co];
42. }
43. }
44. }
45. }
47. **void** print\_matrix(){
48. bar();
49. **if**(eqn\_no){
50. cout<<"\t\tMatrix Expression of Equations:";bl;bl;
51. rep(ro, 0, eqn\_no){
52. cout<<"\t";
53. rep(co, 0, eqn\_no+1){
54. cout<<matrix[ro][co]<<"  ";
55. }bl;
56. }bl;bl;}
57. **else**{
58. cout<<"No equations found. Please input equations.";bl;bl;
59. }
60. cout<<"Prass any key to return...";
61. getch();
62. }
64. **int** oparate\_menu(){
65. bar();
66. cout<<"\t1) Print Upper Matrix";bl;bl;
67. cout<<"\t2) Print Lower Matrix";bl;bl;
68. cout<<"\t3) Calculate Solution";bl;bl;
69. cout<<"\t0) Return To Main Menu";bl;bl;
70. **int** key;
71. cin>>key;
72. **return** key;
73. }

76. **void** cal\_upper\_lower(){
77. bar();
78. memset(mat\_U, 0, **sizeof**(mat\_U));
79. memset(mat\_L, 0, **sizeof**(mat\_L));
80. rep(j, 0, eqn\_no){
81. mat\_U[0][j]=matrix[0][j];
82. }
83. rep(i, 0, eqn\_no){
84. mat\_L[i][i]=1;
85. }
86. rep(i, 1, eqn\_no){
87. mat\_L[i][0]=matrix[i][0]/mat\_U[0][0];
88. }

91. rep(j, 1, eqn\_no){
92. rep(i, 1, j+1){
93. **double** sum=0;
94. rep(k, 0, j){
95. sum+=(mat\_L[i][k]\*mat\_U[k][j]);
96. }
97. mat\_U[i][j]=matrix[i][j]-sum;
98. }
99. rep(i, j+1, eqn\_no){
100. **double** sum=0;
101. rep(k, 0, j){
102. sum+=(mat\_L[i][k]\*mat\_U[k][j]);
103. }
104. mat\_L[i][j]=(1/mat\_U[j][j])\*(matrix[i][j]-sum);
105. }
106. }
107. }
109. **void** print\_mat\_U(){
110. debug\_mat(mat\_U, "Upper");
111. cout<<"Prass any key to return...";
112. getch();
113. }
115. **void** print\_mat\_L(){
116. debug\_mat(mat\_L, "Lower");
117. cout<<"Prass any key to return...";
118. getch();
119. }

122. **void** calculate\_eqn(){
123. bar();
124. **double** z[row\_col\_sz];
125. z[0]=matrix[0][eqn\_no];
126. rep(i, 1, eqn\_no){
127. **double** sum=0;
128. rep(j, 0, i){
129. sum+=(mat\_L[i][j]\*z[j]);
130. }
131. z[i]=matrix[i][eqn\_no]-sum;
132. }
133. memset(X, 0, **sizeof**(X));
134. X[eqn\_no-1]=((z[eqn\_no-1])/mat\_U[eqn\_no-1][eqn\_no-1]);
135. **for**(**int** i=eqn\_no-2; i>=0; i--){
136. **double** sum=0;
137. rep(j, 0, eqn\_no){
138. sum+=(mat\_U[i][j]\*X[j]);
139. }
140. X[i]=(z[i]-sum)/mat\_U[i][i];
141. }
142. cout<<"\tEquations :\n";bl;
143. rep(r, 0, eqn\_no){
144. **int** k=1;
145. cout<<"\t";
146. rep(c, 0, eqn\_no+1){
147. **if**(c==eqn\_no){
148. pf("=");
149. pf(" %.2f", matrix[r][c], k);
150. }**else**
151. pf(" %.2f[X%d] ", matrix[r][c], k);
152. k++;
153. }bl;
154. }bl;bl;
155. cout<<"  Values:";bl;bl;
156. rep(i, 0, eqn\_no){
157. cout<<"   X"<<i+1<<" :"<<X[i]<<endl;
158. }
159. bl;bl;bl;
160. cout<<"Prass any key to return...";
161. getch();
162. }

[**Thank You**]