AD Soyad: Anıl Kutay Uçan
Numara: 20011025
Ders: Sayısal Analiz

Yapılan Yöntemler:

1. Bisection
2. Regula Falsi
3. Newton-Raphson
4. NxN'lik bir matrisin tersi
5. Gauss Eliminasyon
6. Gauss Seidal
7. Sayısal Türev(merkezi,ilerş,geri)
8. Simpson Yöntemi
9. Trapez Yöntemi
10. Değişken Dönüşümsüz Gregory Newton Enterpolasyonu

NOT: Bütün Yöntemler yapılmıştır. Sunumda Regula Falsi ve Gauss Seidal yöntemleri gösterilmiştir. Gösterilen yöntemlerin kodları ve çalıştırıldığında alınan ekran görüntüleri aşağıdadır.

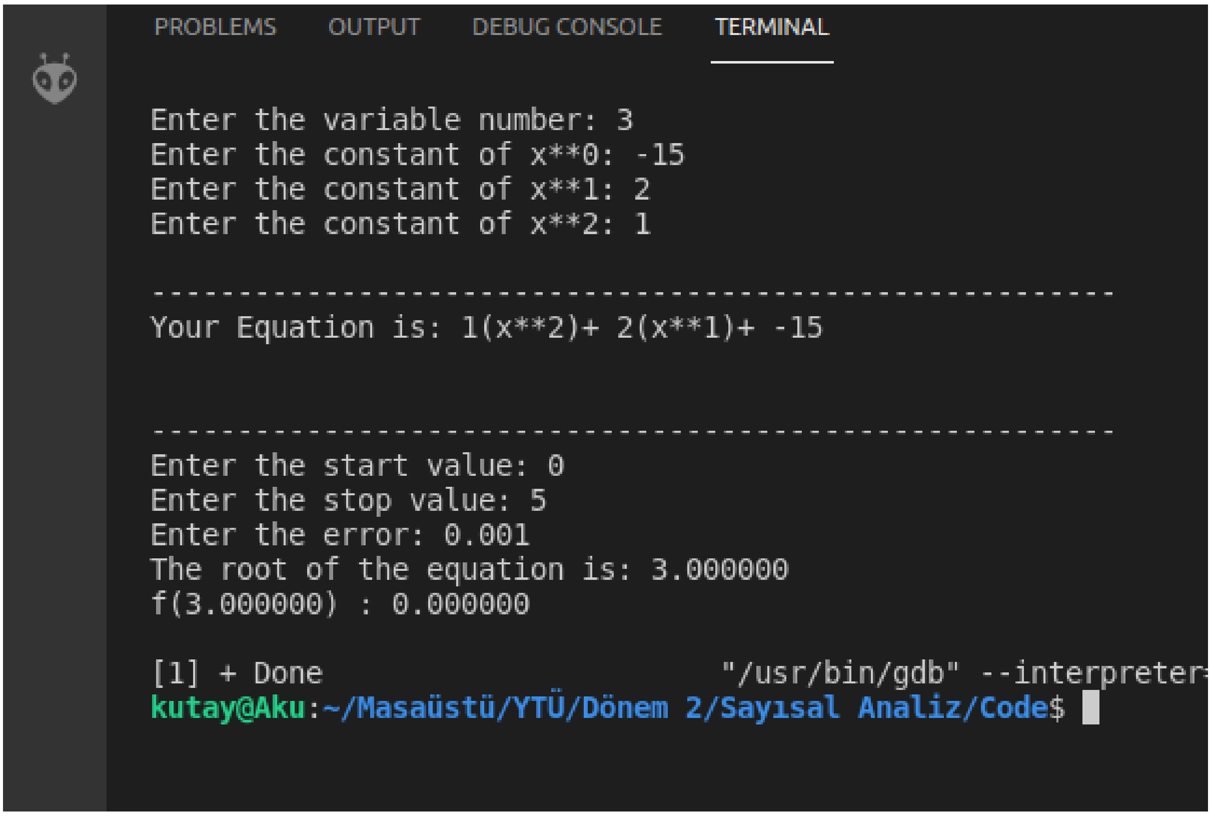
REGULA FALSİ:

Code:

```
#include <stdio.h>

void regulaFalsi(int *variable_number,int *equation);
void getEquation(int *variable_number,int *equation);
void printEquation(int *variable_number,int *equation);
float solveEquation(int *variable_number,int *equation,float x);


int main(){
    int equation[10], variable_number;
    getEquation(&variable_number,&equation[0]);
    printEquation(&variable_number,&equation[0]);
    regulaFalsi(&variable_number,&equation[0]);

}

void regulaFalsi(int *variable_number,int *equation){
    float start,stop,mid,error,start_result,stop_result,mid_result;
    int max_iteration = 20,iteration=0;
    printf("\n-------------------------------------------------------------\n");
    printf("Enter the start value: ");
    scanf("%f",&start);
    printf("Enter the stop value: ");
    scanf("%f",&stop);
    printf("Enter the error: ");
    scanf("%f",&error);

    start_result = solveEquation(variable_number,equation,start);
    stop_result = solveEquation(variable_number,equation,stop);

    if(stop_result * start_result < 0){

        while(stop-start > error && iteration < max_iteration){

            mid = ((start*stop_result) - (stop*start_result)) / (stop_result - start_result);
            mid_result = solveEquation(variable_number,equation,mid);
            if(mid_result * start_result < 0){
                stop_result = mid_result;
                stop = mid;
            }
            else{
                start_result = mid_result;
                start = mid;
            }
            iteration++;

        }
        printf("The root of the equation is: %f \n",mid);
        printf("f(%f) : %f\n\n",mid,mid_result);
    }
    else if(start_result==0){
        printf("The root of the equation is: %f \n",start);
        if(stop_result==0){
            printf("The root of the equation is: %f \n",stop);
        }
    }
    else if(stop_result==0){
        printf("The root of the equation is: %f \n",stop);
    }
    else{
        printf("There is no root");
    }


}


void getEquation(int *variable_number,int *equation){
    int i;
    printf("Enter the variable number: ");
    scanf("%d",variable_number);

    for (i=0;i<*variable_number;i++){
        printf("Enter the constant of x**%d: ",i);
        scanf("%d",equation);
        equation ++;
    }
}

void printEquation(int *variable_number,int *equation){
    int i;
    equation += *variable_number-1;
    printf("\n-------------------------------------------------------\n");
    printf("Your Equation is: ");

    for (i=*variable_number-1;i>0;i--){
        printf("%d(x**%d)+ ",*equation,i);
        equation --;
    }

    printf("%d \n\n",*equation);
}
```

```
float solveEquation(int *variable_number,int *equation,float x){
  int i;
  float result = 0,x_value=1;

  result += *equation;
  equation++;


  for(i=1;i<*variable_number;i++){
     x_value *= x;
     result += *equation * x_value;
     equation++;
  }
  //printf("%d \n\n",result);
  return result;
}
```

Ekran Görüntüsü:



```
Enter the variable number: 3
Enter the constant of x**0: -15
Enter the constant of x**1: 2
Enter the constant of x**2: 1

---------------------------------------------------------
Your Equation is: 1(x**2)+ 2(x**1)+ -15


---------------------------------------------------------
Enter the start value: 0
Enter the stop value: 5
Enter the error: 0.001
The root of the equation is: 3.000000
f(3.000000) : 0.000000

[1] + Done                       "/usr/bin/gdb" --interpreter
kutay@Aku:~/Masaüstü/YTÜ/Dönem 2/Sayısal Analiz/Code$
```

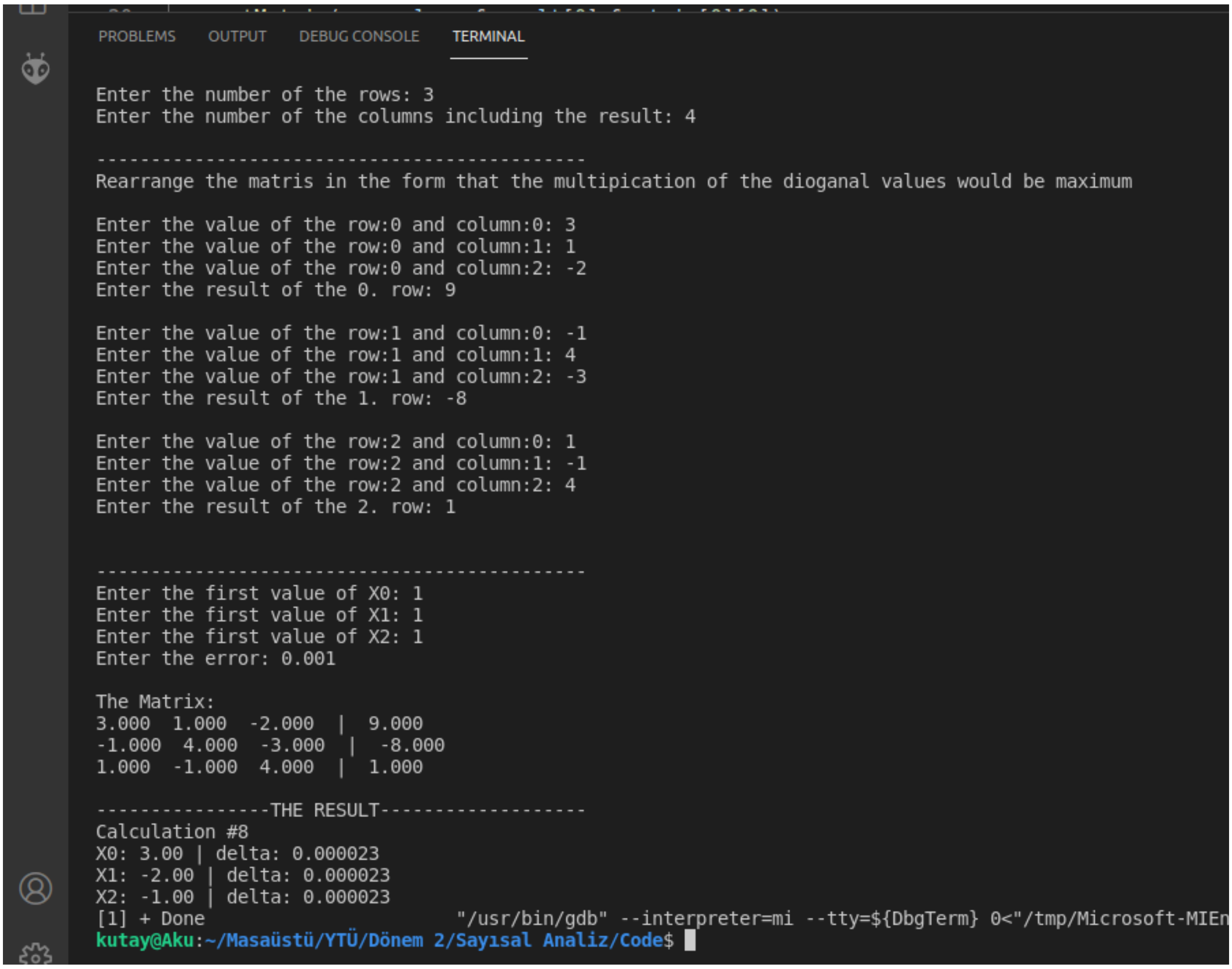
GAUSS SEIDAL:

Code:

```
#include "stdio.h"

#define ROW 10
#define COLUMN 10

void getMatrix(int row,int column,float *result, float *matrix);
void printMatrix(int row,int column, float *matrix);
void gaussSeidal(int row,int column, float *result,float *matrix);

int main(){
  int row,column;
  float result[ROW],matrix[ROW][COLUMN];


  printf("Enter the number of the rows: ");
  scanf("%d",&row);
  printf("Enter the number of the columns including the result: ");
  scanf("%d",&column);

  getMatrix(row,column,&result[0],&matrix[0][0]);


  gaussSeidal(row,column,&result[0],&matrix[0][0]);

}

void gaussSeidal(int row,int column, float *result,float *matrix){
  int i,j;
  int maxIter=20,iter=0;
  float newResult[ROW],maxDelta=10,delta,error;

  printf("Enter the error: ");
  scanf("%f",&error);

  printMatrix(row,column,matrix);

  for(i=0;i<row;i++){
    newResult[i] = result[i];
  }

  while(iter<maxIter && maxDelta > error){
    //CALCULATE THE VALUE OF THE VARABLES
    for(i=0;i<row;i++){
      newResult[i] = matrix[i*COLUMN+column-1];
      for(j=0;j<column-1;j++){
        if(i != j){
          newResult[i] -= matrix[i*COLUMN+j] * newResult[j];
        }
      }
      newResult[i] /= matrix[i*COLUMN+i];
    }


    //FIND THE MAXIMUM DELTA
    maxDelta = 0;
    for(i=0;i<row;i++){
      delta = result[i] - newResult[i];
      if(delta < 0)
        delta *= -1;


      if(delta > maxDelta)
        maxDelta = delta;

      result[i] = newResult[i];
    }
    iter++;
  }

  printf("\n----------------THE RESULT-------------------\n");
  printf("Calculation #%d\n",iter);
  for(i=0;i<row;i++){
    printf("X%d: %.2f | delta: %f\n",i,newResult[i],delta);
  }
}


void getMatrix(int row, int column,float *result,float *matrix){
  int i,j;

  printf("\n---------------------------------------------\n");
  printf("Rearrange the matris in the form that the multipication of the dioganal values would
be maximum\n\n");

  for(i=0;i<row;i++){
    for(j=0;j<column-1;j++){
      printf("Enter the value of the row:%d and column:%d: ",i,j);
      scanf("%f",&matrix[i*COLUMN+j]);
    }
    printf("Enter the result of the %d. row: ",i);
    scanf("%f",&matrix[i*COLUMN+j]);
    printf("\n");
  }
```

```
    printf("\n---------------------------------------------\n");
    for(i=0;i<row;i++){
        printf("Enter the first value of X%d: ",i);
        scanf("%f",&result[i]);
    }
}

void printMatrix(int row,int column,float *matrix){
    int i=0,j=1;

    printf("\nThe Matrix: \n");
    for(i=0;i<row;i++){
        for(j=0;j<column-1;j++){
            printf("%.3f  ",matrix[i*COLUMN+j]);
        }
        printf("|  %.3f",matrix[i*COLUMN+j]);
        printf("\n");
    }
}
```

Ekran Görüntüsü:



```
Enter the number of the rows: 3
Enter the number of the columns including the result: 4

---------------------------------------------
Rearrange the matris in the form that the multipication of the dioganal values would be maximum

Enter the value of the row:0 and column:0: 3
Enter the value of the row:0 and column:1: 1
Enter the value of the row:0 and column:2: -2
Enter the result of the 0. row: 9

Enter the value of the row:1 and column:0: -1
Enter the value of the row:1 and column:1: 4
Enter the value of the row:1 and column:2: -3
Enter the result of the 1. row: -8

Enter the value of the row:2 and column:0: 1
Enter the value of the row:2 and column:1: -1
Enter the value of the row:2 and column:2: 4
Enter the result of the 2. row: 1


---------------------------------------------
Enter the first value of X0: 1
Enter the first value of X1: 1
Enter the first value of X2: 1
Enter the error: 0.001

The Matrix:
3.000  1.000  -2.000  |  9.000
-1.000  4.000  -3.000  |  -8.000
1.000  -1.000  4.000  |  1.000

---------------THE RESULT------------------
Calculation #8
X0: 3.00 | delta: 0.000023
X1: -2.00 | delta: 0.000023
X2: -1.00 | delta: 0.000023
[1] + Done                       "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEn
kutay@Aku:~/Masaüstü/YTÜ/Dönem 2/Sayısal Analiz/Code$ 
```