

Assignment 1

Scientific Programming

Isha Borgaonkar
Student Number: 24209758

Here I coded, compiled and run both programs in C on Sciprog. I am attaching output of both programs, screenshot of code which is done on Sciprog as well as code of .c File.

Output of Program Q1.stub and Q2.stub

The input for Q1.stub file is the data points which are in the Assignment1 file.

```
Windows PowerShell
Last login: Thu Oct 31 12:06:08 2024 from 37.228.210.253
[sp157@sciprog ~]$ gcc -o Q1_stub Q1_stub.c -lm
[sp157@sciprog ~]$ ./Q1_stub
Enter the number of data points (n):
4
Enter the x and y values:
Hit the enter after entering a value
x[0]: 1
y[0]: 2
x[1]: 2
y[1]: 4
x[2]: 3
y[2]: 3
x[3]: 4
y[3]: 5
The equation of the best-fit line is:  $y = 0.80x + 1.50$ 
[sp157@sciprog ~]$ gcc -o Q2_stub Q2_stub.c -lm
[sp157@sciprog ~]$ ./Q2_stub
4x4 Hilbert Matrix:
1.0000 0.5000 0.3333 0.2500
0.5000 0.3333 0.2500 0.2000
0.3333 0.2500 0.2000 0.1667
0.2500 0.2000 0.1667 0.1429
Determinant of the 4x4 Hilbert matrix: 0.0000
[sp157@sciprog ~]$ client_loop: send disconnect: Connection reset
PS C:\Users\ISHA>
```

Q1.Stub.c File sciprog

```
sp157@sciprog:~  
GNU nano 5.6.1 Q1_stub.c  
#include <stdio.h>  
#include <math.h>  
  
int main() {  
    /*Declaring the variable n and taking input for N  
    int n;  
    printf("Enter the number of data points (n):\n ");  
    scanf("%d", &n);  
  
    /*Enter the number of data points  
    /*Enter (xi, yi) values i=1, 2, ..., n Note that C array indices start from 0  
    float x[n], y[n];  
  
    /*Taking input for x and y values  
    printf("Enter the x and y values:\n");  
    printf("Hit the enter after entering a value\n ");  
    for (int i = 0; i < n; i++) {  
        printf("x[%d]: ", i);  
        scanf("%f", &x[i]);  
        printf("y[%d]: ", i);  
        scanf("%f", &y[i]);  
    }  
  
    /* Calculate alpha and beta formulas  
    use the pow() function for the calculation of power of 2 in the formula.  
    check and exit the code if the denominator for a is equal to zero.*/  
  
    /* Initializing the variables to compute sums required for alpha and beta calculations  
    float sum_x = 0, sum_y = 0, sum_x2 = 0, sum_xy = 0;  
  
    /* Calculate the sums  
    for (int i = 0; i < n; i++) {  
        sum_x += x[i];  
        sum_y += y[i];  
        sum_x2 += pow(x[i], 2);  
        sum_xy += x[i] * y[i];  
    }  
}
```

Help Write Out Where Is Cut Execute Location Undo Set Mark To Bracket Previous Back
Exit Read File Replace Paste Justify Go To Line Redo Copy Where Was Next Forward

Type here to search 13°C 16:10 28-10-2024

```
sp157@sciprog:~
GNU nano 5.6.1 Q1_stub.c

scanf("%f", &y[i]);
}

/* Calculate alpha and beta formulas
use the pow() function for the calculation of power of 2 in the formula.
check and exit the code if the denominator for a is equal to zero.*/

/* Initializing the variables to compute sums required for alpha and beta calculations
float sum_x = 0, sum_y = 0, sum_x2 = 0, sum_xy = 0;

/* Calculate the sums
for (int i = 0; i < n; i++) {
    sum_x += x[i];
    sum_y += y[i];
    sum_x2 += pow(x[i], 2);
    sum_xy += x[i] * y[i];
}

/*Print the equation of the line
float denominator = (n * sum_x2) - pow(sum_x, 2);

/*Checking whether the denominator is zero just to avoid division by zero error
if (denominator == 0) {
    printf("Error: Denominator is zero. Cannot compute alpha and beta.\n");
    1;
}

/*Calculating alpha and beta
float alpha = (n * sum_xy - sum_x * sum_y) / denominator;
float beta = (sum_y - alpha * sum_x) / n;

/*Printing the equation of the line
printf("The equation of the best-fit line is: y = %.2fx + %.2f\n", alpha, beta);

0;
}

Help Write Out Where Is Cut Execute Location M-U Undo M-A Set Mark M-T To Bracket M-Q Previous AB Back
Exit Read File Replace Paste Justify Go To Line M-E Redo M-G Copy M-W Where Was M-W Next AB Forward
```

Q1.Stub.c File on sciprog

```
sp157@sciprog:~
GNU nano 5.6.1 Q2_stub.c
#include <stdio.h>
#include <math.h>

/* function to calculate the determinant of a 3x3 matrix
float getdet(float minor[3][3]) {
    minor[0][0] * (minor[1][1] * minor[2][2] - minor[1][2] * minor[2][1])
    - minor[0][1] * (minor[1][0] * minor[2][2] - minor[1][2] * minor[2][0])
    + minor[0][2] * (minor[1][0] * minor[2][1] - minor[1][1] * minor[2][0]);
}

int main( int argc, char *argv[]) {
//Complete variable declarations
float hmat[4][4], minor[3][3];

//Initialize and print the 4x4 Hilbert matrix formatted to four decimal places
printf("4x4 Hilbert Matrix:\n");
for (int i = 0; i < 4; i++) {
    for (int j = 0; j < 4; j++) {
        hmat[i][j] = 1.0 / (i + j + 1);
        printf("%0.4f ", hmat[i][j]);
    }
    printf("\n");
}

// Calculate the determinant of the 4x4 Hilbert matrix using cramer's rule
float determinant = 0.0;

// Applying cramer's rule
// a(matrix) * det(minor matrix of b, c, d)
for (int i = 1; i < 4; i++) {
    for (int j = 1; j < 4; j++) {
        minor[i - 1][j - 1] = hmat[i][j];
    }
}

Read 83 lines (converted from DOS format)
Help Write Out Where Is Cut Execute Location M-U Undo M-A Set Mark M-T To Bracket M-Q Previous AB Back
Exit Read File Replace Paste Justify Go To Line M-E Redo M-G Copy M-W Where Was M-W Next AB Forward
```

```

determinant += hmat[0][0] * getdet(minor);
// -b(matrix) * det(minor matrix of a, c, d)
for (int i = 1; i < 4; i++) {
    for (int j = 0; j < 4; j++) {
        if (j < 1)
            minor[i - 1][j] = hmat[i][j];
        else
            minor[i - 1][j - 1] = hmat[i][j];
    }
}
determinant -= hmat[0][1] * getdet(minor);
// -c(matrix) * det(minor matrix of a, b, d)
for (int i = 1; i < 4; i++) {
    for (int j = 0; j < 4; j++) {
        if (j < 2)
            minor[i - 1][j] = hmat[i][j];
        else
            minor[i - 1][j - 1] = hmat[i][j];
    }
}
determinant += hmat[0][2] * getdet(minor);
// -d * det(minor matrix of a, b, c)
for (int i = 1; i < 4; i++) {
    for (int j = 0; j < 3; j++) {
        minor[i - 1][j] = hmat[i][j];
    }
}
determinant -= hmat[0][3] * getdet(minor);

// Print the calculated determinant
printf("Determinant of the 4x4 Hilbert matrix: %.4f\n", determinant);

```

0;

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location ^U Undo ^M Set Mark ^I To Bracket ^O Previous ^B Back
 ^X Exit ^R Read File ^N Replace ^P Paste ^J Justify ^G Go To Line ^E Redo ^G Copy ^I Where Was ^N Next ^F Forward