Programming Fundamentals – ENSF 337

Lab 1

M. Moussavi

Jaxon Braun

B01

Submitted on September 21, 2021

Exercise B. Variables and Basic Arithmetic

Source Code:

```
/*
   File Name: lab1_exe_B.c
   Assignment: Lab 1 Exercise B
   Lab Section: B01
   Completed By: Jaxon Braun
   Submission Date: Sept 20, 2021
*/

#include <stdio.h>
int main()
{
   double num1 = -34.5;
   double num2 = 98.7;

   double sum;             // sum of num1 and num2
   double sumSquared;      // the square of num1 plus num2

   sum = num1 + num2;

   sumSquared = sum * sum;

   printf("The sum squared is %f \n", sumSquared);

   sumSquared *= 2;

   printf("The sum squared now is: %f \n", sumSquared);
```

```
    return 0;
}
```

Output:

```
The sum squared is 4121.640000
The sum squared now is: 8243.280000
```

Exercise C. Operator Precedence

a) z = 2.5 + 4 * -1.5 – (2.5 + 4) * -1.5
   z = 2.5 + 4 * -1.5 – 6.5 * -1.5
   z = 2.5 + -6 – -9.75
   z = 6.25

b) z = 18 / 4 + 18 % 4
   z = 4 + 2
   z = 6

c) z = 4 / 18 + 4 % 18
   z = 0 + 4
   z = 4

d) z = 5 * 2.5 – 4 / 5
   z = 12.5 – 0
   z = 12.5

e) z = 1 – (1 – (1 – (1 – (1 – 4))))
   z = 1 – (1 – (1 – (1 – (-3))))
   z = 1 – (1 – (1 – (4)))
   z = 1 – (1 – (-3))
   z = 1 – (4)
   z = -3

f) z = sqrt(sqrt((double)4)
   z = sqrt(2)
   z = 1.414214

Exercise D: Mathematical Expressions

Source code:

```
/*
   File Name: lab1_exe_D.c
   Assignment: Lab 1 Exercise D
   Lab Section: B01
   Completed By: Jaxon Braun
   Submission Date: Sept 20, 2021
*/

#include <stdio.h>
#include <math.h>

int main()
{
   float angle;
   printf("Please enter a value for an angle in radians: \n");
   scanf("%f", &angle);

   // using the built in math.h function to calculate sin(x) where x = angle
   float result_sin_function = sin(angle);
   printf("Using the built-in function, sin(%f) = %f \n", angle, result_sin_function);

   // using the taylor series approximation to calcualte sin(x) where x = angle
   float three_fact = 3 * 2 * 1, five_fact = 5 * 4 * 3 * 2 * 1, seven_fact = 7 * 6 * 5 * 4 * 3 * 2 * 1;
   float result_taylor_approx = angle - (pow(angle, 3) / three_fact) + (pow(angle, 5) / five_fact) -(pow(angle, 7) /
seven_fact);
   printf("Using the taylor series approximation, sin(%f) = %f \n", angle, result_taylor_approx);

   return 0;
}
```

Terminal output:

```
Please enter a value for an angle in radians:
1
Using the built-in function, sin(1.000000) = 0.841471
Using the taylor series approximation, sin(1.000000) = 0.841468
```

Angle	**Built-in sin(x)**	**Taylor Approximation of sin(x)**
0	0.000000	0.000000
0.5	0.479426	0.479426
1	0.841471	0.841468
2.5	0.598472	0.588534

Exercise E: Problem Solving

Source Code:

```
/*
   File Name: lab1_exe_E.c
   Assignment: Lab 1 Exercise E
   Lab Section: B01
   Completed By: Jaxon Braun
   Submission Date: Sept 21, 2021
*/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main()
{
   float a, b, c;

   //asking for the inputs of coefficients a b and c

   printf("This program calculates the solutions of a quadratic equation given the coefficients in the general quadratic equation ax^2 + bx + c. \n");
   printf("Please enter a value for coefficient a: \n");
   scanf("%f", &a);
   printf("Please enter a value for coefficient b: \n");
   scanf("%f", &b);
   printf("Please enter a value for coefficient c: \n");
   scanf("%f", &c);
```

```
    //computing the discriminant of the equation to determine if the equation will have complex solutions or not


    float discriminant = (pow(b, 2)) - (4 * a * c);


    //separating the quadratic into two ratios in order to simplify calculation. Using the absolute value of the discriminant for now, but will consider its true value later in order to determine if we need to represent complex solutions or not


    float ratio_1 = (b * -1) / (2 * a);
    float ratio_2 = (sqrt(abs(discriminant))) / (2 * a);


    //if the discriminant is 0 or greater, we calculate the solutions using the quadratic equation normally. If the discriminant is less than zero, we must represent the complex solutions, which we can do by simply adding an i to ratio_2 and not add together ratio_1 and ratio_2. We also need to print the results here, as the formatting is different depending on if we need complex solutions or not.


    if (discriminant >= 0){
        float x_1 = ratio_1 + ratio_2;
        float x_2 = ratio_1 - ratio_2;
        printf("The solutions to the quadratic are: \n x_1 = %f and x_2 = %f \n", x_1, x_2);
    }
    else {
        ratio_2 = abs(ratio_2);
        printf("The solutions to the quadratic are: \n x_1 = %f + %fi and x_2 = %f - %fi \n", ratio_1, ratio_2, ratio_1, ratio_2);
    }


    return 0;
}
```

Terminal Outputs:

```
This program calculates the solutions of a quadratic equation given the coefficients in the general
quadratic equation ax^2 + bx + c.
Please enter a value for coefficient a:
1
Please enter a value for coefficient b:
2
Please enter a value for coefficient c:
2
The solutions to the quadratic are:
 x_1 = -1.000000 + 1.000000i and x_2 = -1.000000 - 1.000000i
```

```
This program calculates the solutions of a quadratic equation given the coefficients in the general
quadratic equation ax^2 + bx + c.
Please enter a value for coefficient a:
1
Please enter a value for coefficient b:
-4
Please enter a value for coefficient c:
3
The solutions to the quadratic are:
 x_1 = 3.000000 and x_2 = 1.000000
```

```
This program calculates the solutions of a quadratic equation given the coefficients in the general
quadratic equation ax^2 + bx + c.
Please enter a value for coefficient a:
2
Please enter a value for coefficient b:
-6
Please enter a value for coefficient c:
3
The solutions to the quadratic are:
 x_1 = 2.366025 and x_2 = 0.633975
```

a	b	c	Root 1	Root 2
1	2	2	-1 + 1i	-1 – 1i
1	-4	3	3	1
2	-6	3	2.366025	0.633975