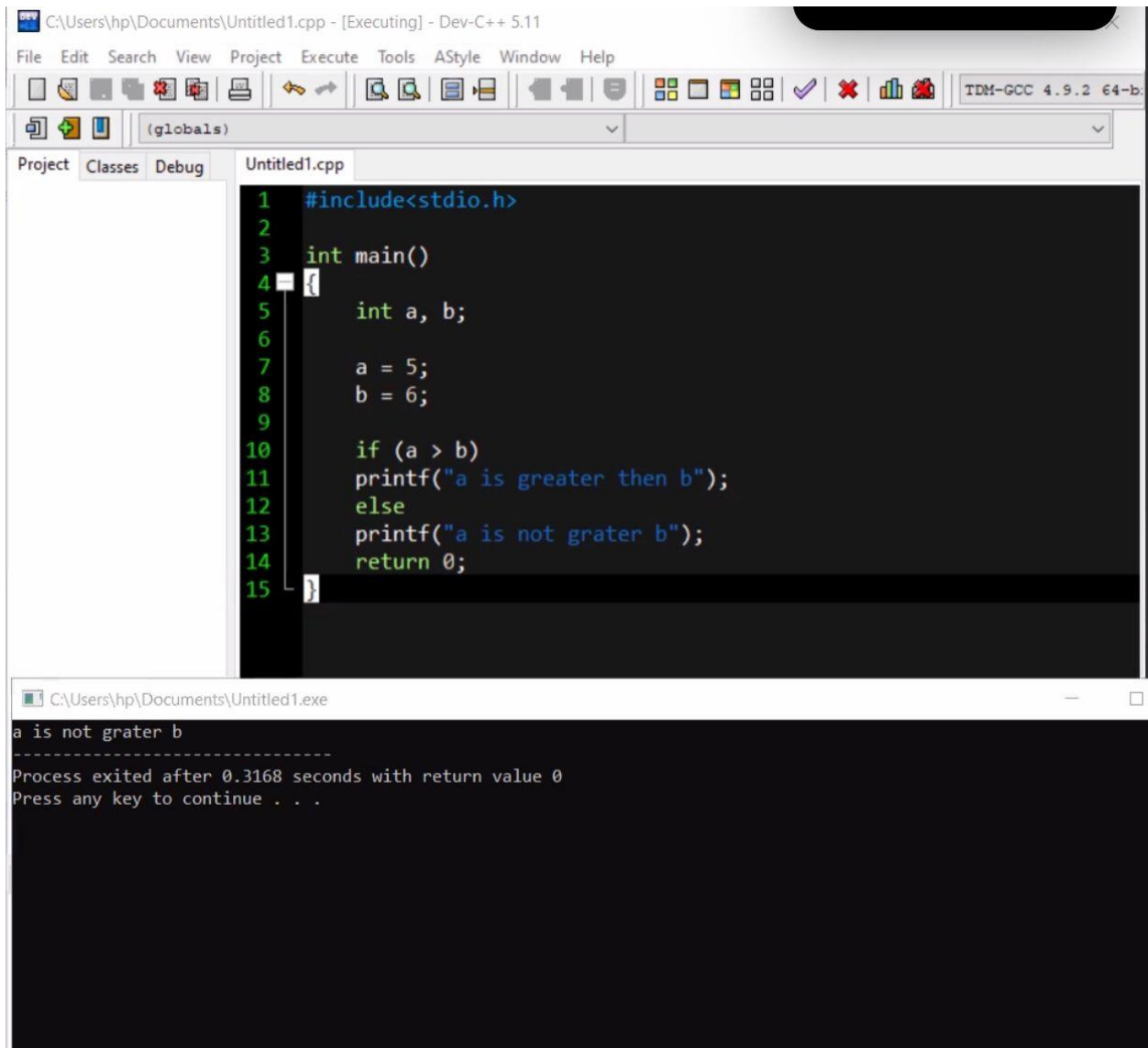


Comparing Two numbers



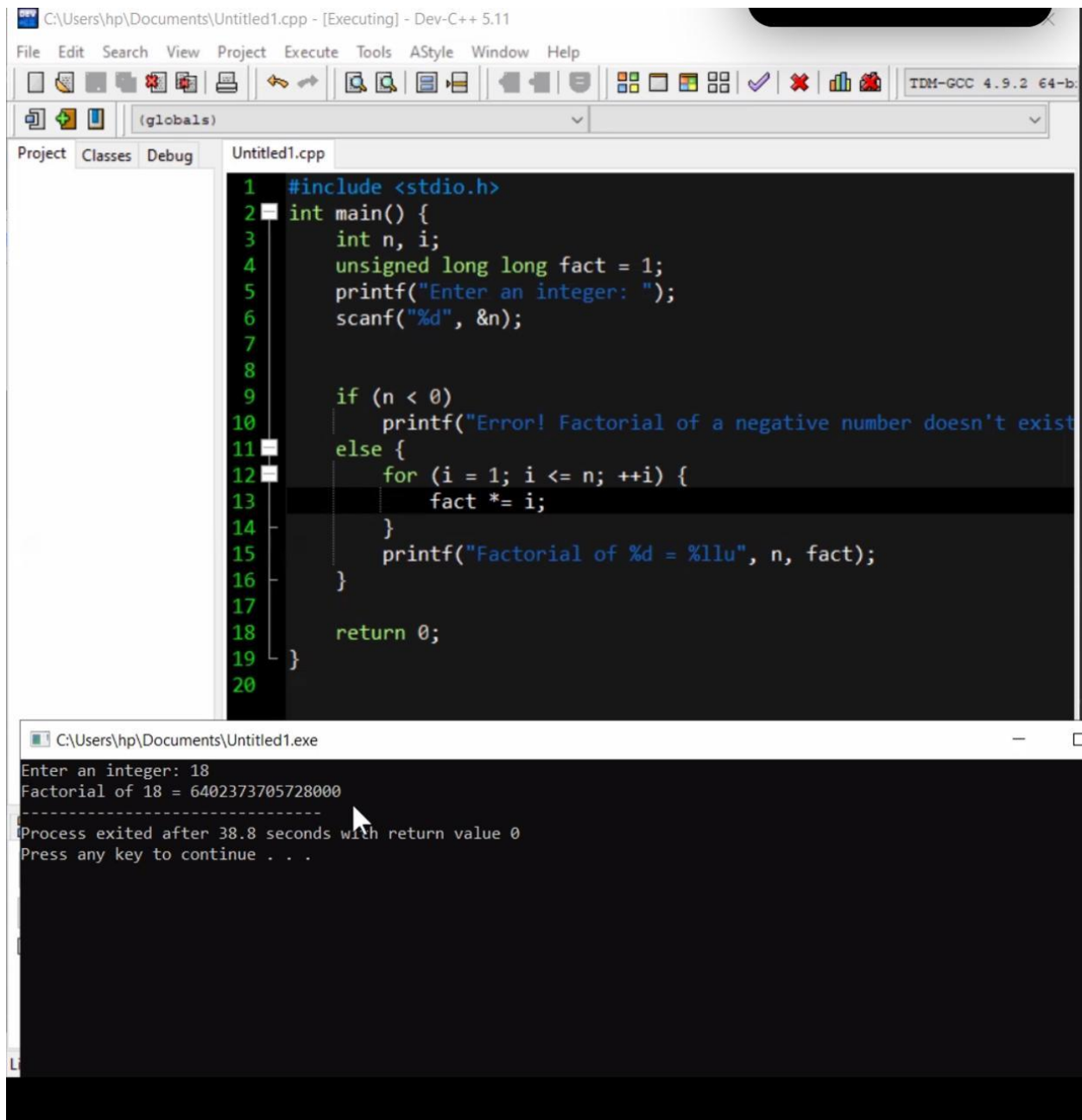
The screenshot shows the Dev-C++ IDE with a C++ program titled "Untitled1.cpp" being executed. The program compares two integers, a and b, and prints a message based on the result. The code is as follows:

```
1  #include<stdio.h>
2
3  int main()
4  {
5      int a, b;
6
7      a = 5;
8      b = 6;
9
10     if (a > b)
11         printf("a is greater then b");
12     else
13         printf("a is not grater b");
14     return 0;
15 }
```

The output window shows the result of the execution:

```
a is not grater b
-----
Process exited after 0.3168 seconds with return value 0
Press any key to continue . . .
```

Finding Factorial of given numbers:



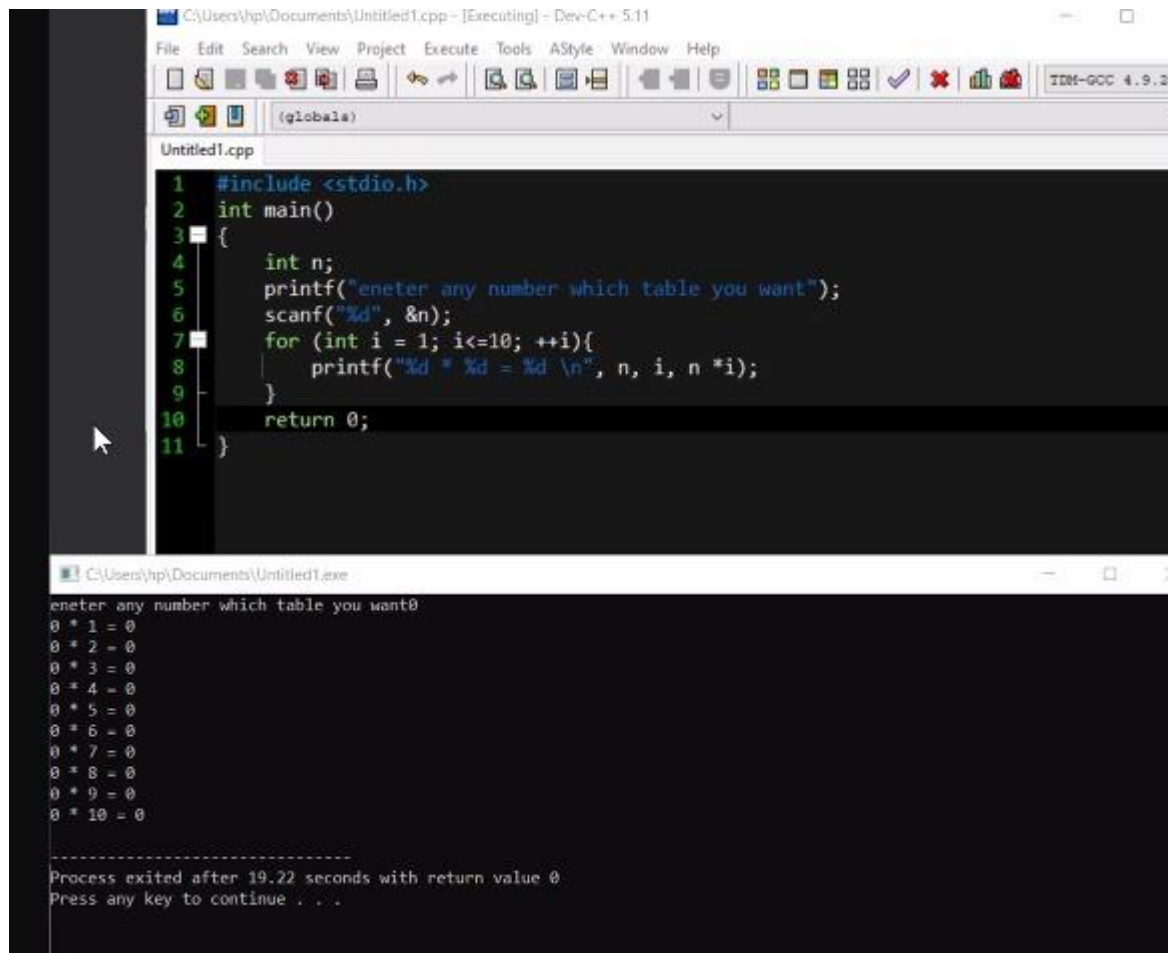
The image shows a screenshot of the Dev-C++ IDE. The top window displays the source code for a C++ program that calculates the factorial of a given integer. The code is as follows:

```
1 #include <stdio.h>
2 int main() {
3     int n, i;
4     unsigned long long fact = 1;
5     printf("Enter an integer: ");
6     scanf("%d", &n);
7
8
9     if (n < 0)
10        printf("Error! Factorial of a negative number doesn't exist");
11    else {
12        for (i = 1; i <= n; ++i) {
13            fact *= i;
14        }
15        printf("Factorial of %d = %llu", n, fact);
16    }
17
18    return 0;
19 }
20
```

The bottom window shows the execution output of the program. It prompts the user to enter an integer, and the user has entered 18. The program then outputs the factorial of 18, which is 6402373705728000. The output also indicates that the process exited after 38.8 seconds with a return value of 0.

```
C:\Users\hp\Documents\Untitled1.exe
Enter an integer: 18
Factorial of 18 = 6402373705728000
-----
Process exited after 38.8 seconds with return value 0
Press any key to continue . . .
```

Finding table of given numbers:



The image shows a screenshot of the Dev-C++ IDE. The top window, titled 'Untitled1.cpp', contains the following C++ code:

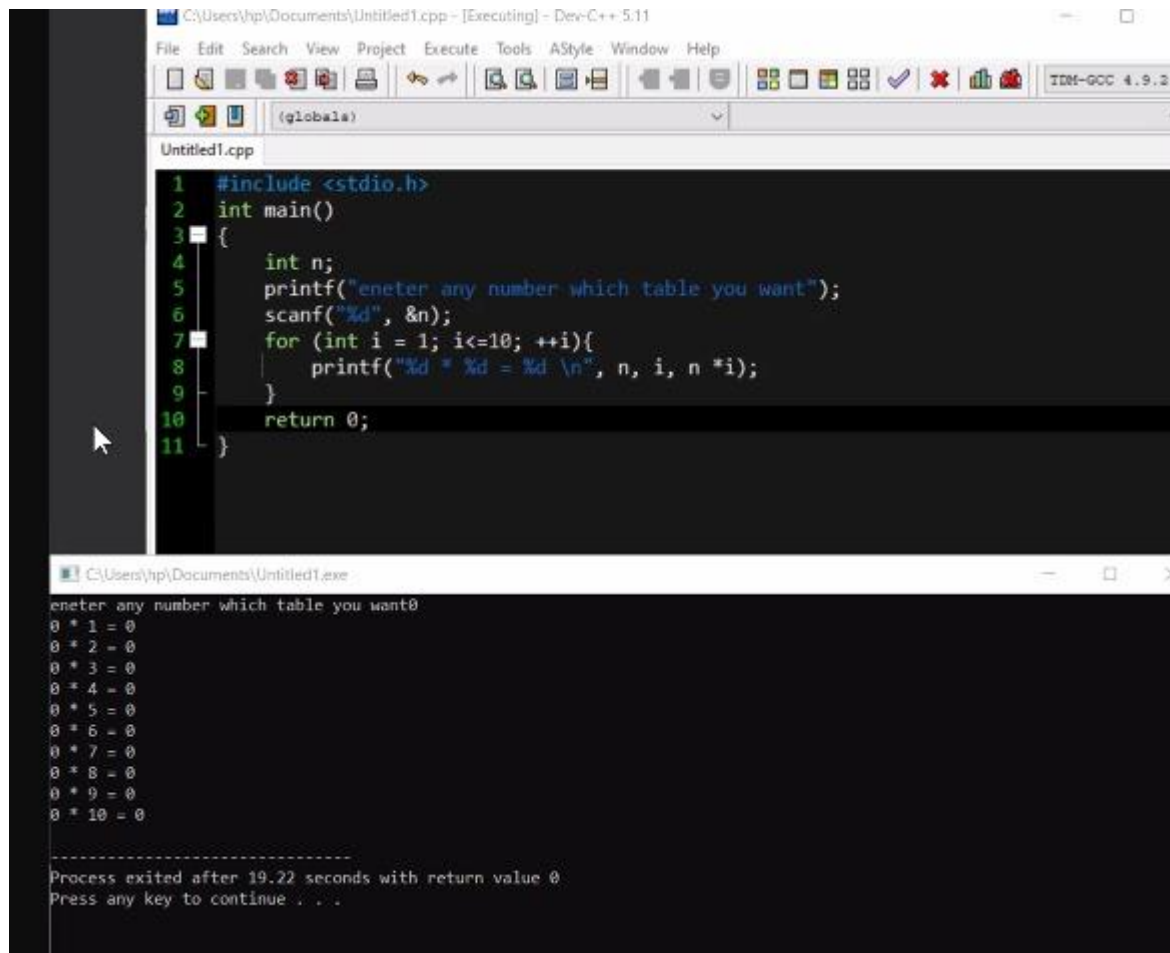
```
1 #include <stdio.h>
2 int main()
3 {
4     int n;
5     printf("enter any number which table you want");
6     scanf("%d", &n);
7     for (int i = 1; i<=10; ++i){
8         printf("%d * %d = %d \n", n, i, n * i);
9     }
10    return 0;
11 }
```

The bottom window, titled 'Untitled1.exe', shows the program's execution. It prompts the user to enter a number, and the output displays the multiplication table for the entered number (0 in this case):

```
enter any number which table you want0
0 * 1 = 0
0 * 2 = 0
0 * 3 = 0
0 * 4 = 0
0 * 5 = 0
0 * 6 = 0
0 * 7 = 0
0 * 8 = 0
0 * 9 = 0
0 * 10 = 0

-----
Process exited after 19.22 seconds with return value 0
Press any key to continue . . .
```

Even and odd finding program:

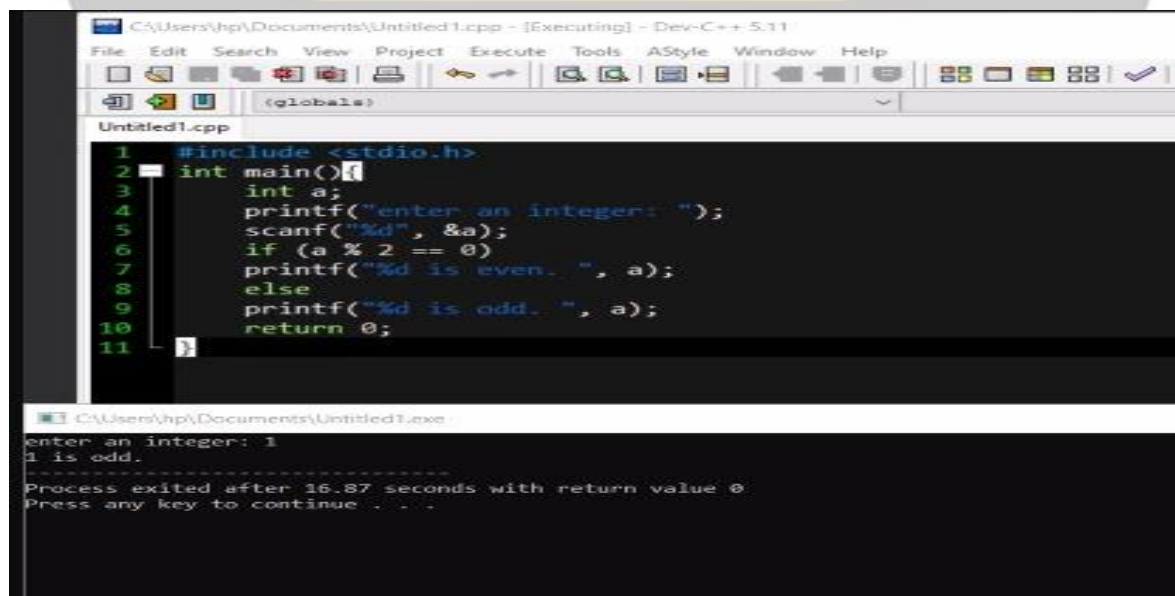


The screenshot shows the Dev-C++ IDE with a C program in the editor and its execution output in the console. The program prompts the user to enter a number and then displays its multiplication table from 1 to 10. The console output shows the user entered 0, resulting in a table of zeros.

```
1 #include <stdio.h>
2 int main()
3 {
4     int n;
5     printf("enter any number which table you want");
6     scanf("%d", &n);
7     for (int i = 1; i<=10; ++i){
8         printf("%d * %d = %d \n", n, i, n *i);
9     }
10    return 0;
11 }
```

```
enter any number which table you want0
0 * 1 = 0
0 * 2 = 0
0 * 3 = 0
0 * 4 = 0
0 * 5 = 0
0 * 6 = 0
0 * 7 = 0
0 * 8 = 0
0 * 9 = 0
0 * 10 = 0

-----
Process exited after 19.22 seconds with return value 0
Press any key to continue . . .
```



The screenshot shows the Dev-C++ IDE with a C program in the editor and its execution output in the console. The program prompts the user to enter an integer and then checks if it is even or odd using a modulo operation. The console output shows the user entered 1, which is odd.

```
1 #include <stdio.h>
2 int main()
3 {
4     int a;
5     printf("enter an integer: ");
6     scanf("%d", &a);
7     if (a % 2 == 0)
8         printf("%d is even. ", a);
9     else
10        printf("%d is odd. ", a);
11    return 0;
12 }
```

```
enter an integer: 1
1 is odd.

-----
Process exited after 16.87 seconds with return value 0
Press any key to continue . . .
```

Generating / Summing of simple series (even / odd)

```
Untitled1.cpp
1 #include<stdio.h>
2 int main()
3 {
4     int n, sum;
5     printf("enter the n value: \n");
6     scanf("%d", &n);
7     sum = 1;
8     printf("%d", sum);
9     for (int i =2; i<=n; i++)
10    {
11        printf("+%d", i);
12        sum += i;
13    }
14    printf("%d", sum);

```

C:\Users\hp\Documents\Untitled1.exe

enter the n value:
10
1+2+3+4+5+6+7+8+9+10=55

Process exited after 15.12 seconds with return value 0
Press any key to continue . . .

Scanf Program example:

C:\Users\hp\Documents\Untitled1.cpp - [Executing] - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

(globals)

Project Classes Debug

```
Untitled1.cpp
1 #include <stdio.h>
2
3 int sum (int a, int b)
4 {
5     return a+b;
6 }
7
8 int main()
9 {
10
11     int num1, num2, num3;
12     printf("Enter first number");
13     scanf("%d", &num1);
14     printf("enter the second");
15     scanf("%d", &num2);
16
17     //calling the function
18     num3 = sum(num1, num2);
19     printf("sum of entered number: %d", num3);
20     return 0;
21
22

```

Compiler Resources Compile Log Debug Find Results Close

Abort Compilation

☐ Shorten compiler paths

Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\hp\Documents\Untitled1.exe
- Output Size: 128.119140625 KiB
- Compilation Time: 0.66s

line: 22 Col: 2 Sel: 0 Lines: 22 Length: 331 Insert Done parsing in 0.031 seconds

Output:

```
C:\Users\hp\Documents\Untitled1.exe
Enter first number5
enter the second5
sum of entered number: 10
-----
Process exited after 59.99 seconds with return value 0
Press any key to continue . . .
```

Solving Arithmetic problem

C:\Users\hp\Documents\Untitled1.cpp - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

(globals)

Untitled1.cpp

```
1 #include<stdio.h>
2
3 int main()
4 {
5     int a = 3.7, b = 5.8, res;
6
7     //printing a and b
8
9     printf("a is %d and b is %d\n", a, b);
10
11     res = a + b; //addition
12     printf("a + b is %d\n", res);
13
14     res = a - b; //subtraction
15     printf("a - b is %d\n", res);
16
17     res = a * b; //multiplication
18     printf("a * b is %d\n", res);
19
20     res = a / b; //division
21     printf("a / b is %d\n", res);
22
23     res = a % b; //average
24     printf("a % b is %d\n", res);
25
26     return 0;
27 }
28
```

Compiler Resources Compile Log Debug Find Results Close

About Compilation

Shorten compiler paths

Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\hp\Documents\Untitled1.exe
- Output Size: 128.431640625 KiB
- Compilation Time: 0.45s

Output:

C:\Users\hp\Documents\Untitled1.exe

```
a is 3 and b is 5
a + b is 8
a - b is -2
a * b is 15
a / b is 0
a % b is 3

-----
Process exited after 0.3102 seconds with return value 0
Press any key to continue . . .
```

Quadratic program in C

$$\text{root1} = \frac{-b + \sqrt{(b^2 - 4ac)}}{2a}$$

If the discriminant > 0 ,

$$\text{root2} = \frac{-b - \sqrt{(b^2 - 4ac)}}{2a}$$

If the discriminant $= 0$,

$$\text{root1} = \text{root2} = \frac{-b}{2a}$$

$$\text{root1} = \frac{-b}{2a} + \frac{i \sqrt{-(b^2 - 4ac)}}{2a}$$

If the discriminant < 0 ,

$$\text{root2} = \frac{-b}{2a} - \frac{i \sqrt{-(b^2 - 4ac)}}{2a}$$




```

1  #include <math.h>
2  #include <stdio.h>
3  int main() {
4      double a, b, c, discriminant, root1, root2, realPart, imagPart;
5      printf("Enter coefficients a, b and c: ");
6      scanf("%lf %lf %lf", &a, &b, &c);
7
8      discriminant = b * b - 4 * a * c;
9
10     // condition for real and different roots
11     if (discriminant > 0) {
12         root1 = (-b + sqrt(discriminant)) / (2 * a);
13         root2 = (-b - sqrt(discriminant)) / (2 * a);
14         printf("root1 = %.2lf and root2 = %.2lf", root1, root2);
15     }
16
17     // condition for real and equal roots
18     else if (discriminant == 0) {
19         root1 = root2 = -b / (2 * a);
20         printf("root1 = root2 = %.2lf;", root1);
21     }
22
23     // if roots are not real
24     else {
25         realPart = -b / (2 * a);
26         imagPart = sqrt(-discriminant) / (2 * a);
27         printf("root1 = %.2lf+%.2lfi and root2 = %.2lf-%.2lfi", realPart, imagPart, realPart, imagPart);
28     }
29
30     return 0;
31 }
32

```

Output:

```

C:\Users\hp\Documents\Untitled1.exe
Enter coefficients a, b and c: 4
5
10
root1 = -0.63+1.45i and root2 = -0.63-1.45i
-----
Process exited after 7.404 seconds with return value 0
Press any key to continue . . .

```