# 2021-10-31

1 )Built-in functions
2)If – else
3)Exercise Assignment 1

# BUILT-IN FUNCTION

## PRINTF

```
/*
 * Format: int printf(const char * format, ... );
 * Purpose: display (format) string
 * Library: stdio
 */
```

| | |
|---|---|
| %c | Character |
| %d | Integer |
| %f | Floating-point |
| %s | String |
| %p | void* (pointer to void) |
| %e | Scientific (exponential) notation |

| | |
|---|---|
| \n | New line |
| \t | Tab |

```
// %c: character
printf ("Characters: %c %c \n", 'a', 65);

// %s: string
printf ("%s \n", "A string");

// %d: signed decimal integer
printf ("Decimals: %d \n", 1977);
```

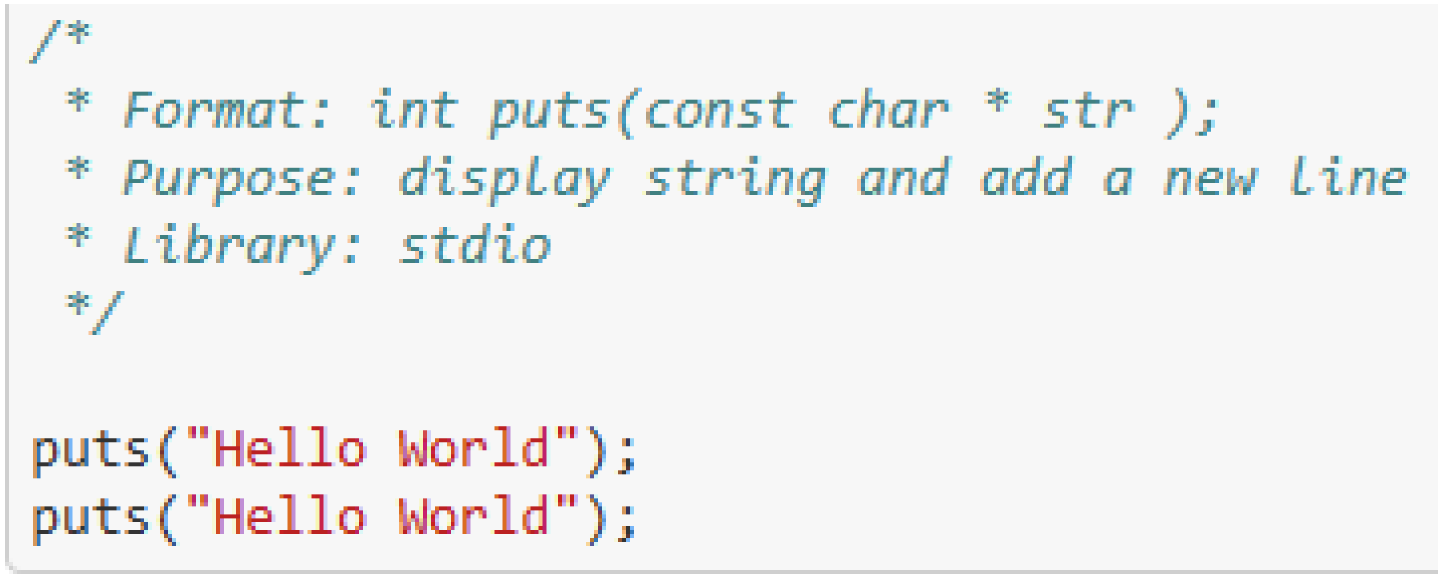
```
Characters: a A
A string
Decimals: 1977
```

```
// %f: float
printf ("Floats: %f \n", 3.1416);

// %._f: float but take _ decumal part (with _ is the number of decumal part)
printf ("Floats: %.2f \n", 3.1416);
```

```
Floats: 3.141600
Floats: 3.14
```

# BUILT-IN FUNCTION

## PUTS



```
/*
 * Format: int puts(const char * str );
 * Purpose: display string and add a new line
 * Library: stdio
 */

puts("Hello World");
puts("Hello World");
```

```
Hello World
Hello World
```

# BUILT-IN FUNCTION

## SCANF

```
/*
 * Format: int scanf(const char * format, ... );
 * Purpose: reads data from stdin -> stores them (with parameter format) into the locations pointed by the additional arguments
 * Library: stdio
 */
```

```c
char name [80];
int age;

printf("Enter your name: ");
scanf("%s", name);

printf("Enter your age: ");
scanf("%d", &age);

printf("\nMr. %s, you are %d years old.\n", name, age);
```

```
Enter your name: Khoa
Enter your age: 20

Mr. Khoa, you are 20 years old.
```

# BUILT-IN FUNCTION

## FGETS

```
/*
 * Format: char * fgets(char *variable, int size, FILE *stream)
 * Purpose: reads data from stdin (with the whitespace) -> stores them into the variable
 * Library: stdio
 */
```

```
char name [80];

printf("Enter your full name: ");
fgets(name, 80, stdin);

printf("\nGood Morning Mr. %s\n", name);
```

```
Enter your full name: Nguyen Khoa

Good Morning Mr. Nguyen Khoa
```

## BUILT-IN FUNCTION

### FGETS

```
/*
 * Format: char * fgets(char *variable, int size, FILE *stream)
 * Purpose: reads data from stdin (with the whitespace) -> stores them into the variable
 * Library: stdio
 */
```

```
char name [80];

printf("Enter your full name: ");
fgets(name, 80, stdin);

printf("\nGood Morning Mr. %s\n", name);
```

```
Enter your full name: Nguyen Khoa

Good Morning Mr. Nguyen Khoa
```

# BUILT-IN FUNCTION

## POW

```
/*
 * Format: double pow(double base, double exponent);
 * Purpose: calculates base raised to the power exponent
 * Library: math
 */
```

```
printf ("7 ^ 3 = %f\n", pow(7.0, 3.0));
printf ("7 ^ 3 = %f\n", pow(7, 3));
printf ("4.73 ^ 12 = %f\n", pow(4.73, 12.0));
printf ("32.01 ^ 1.52 = %f\n", pow(32.01, 1.52));
```

```
7 ^ 3 = 343.000000
7 ^ 3 = 343.000000
4.73 ^ 12 = 125410439.217423
32.01 ^ 1.52 = 194.103884
```

# BUILT-IN FUNCTION

## SQRT

```
/*
 * Format: double sqrt(double x);
 * Purpose: calculates the square root of x
 * Library: math
 */
```

```
printf ("Square root of 2 = %f\n", sqrt(2));
printf ("Square root of 4 = %f\n", sqrt(4));
```

```
Square root of 2 = 1.414214
Square root of 4 = 2.000000
```

# BUILT-IN FUNCTION

## EXP

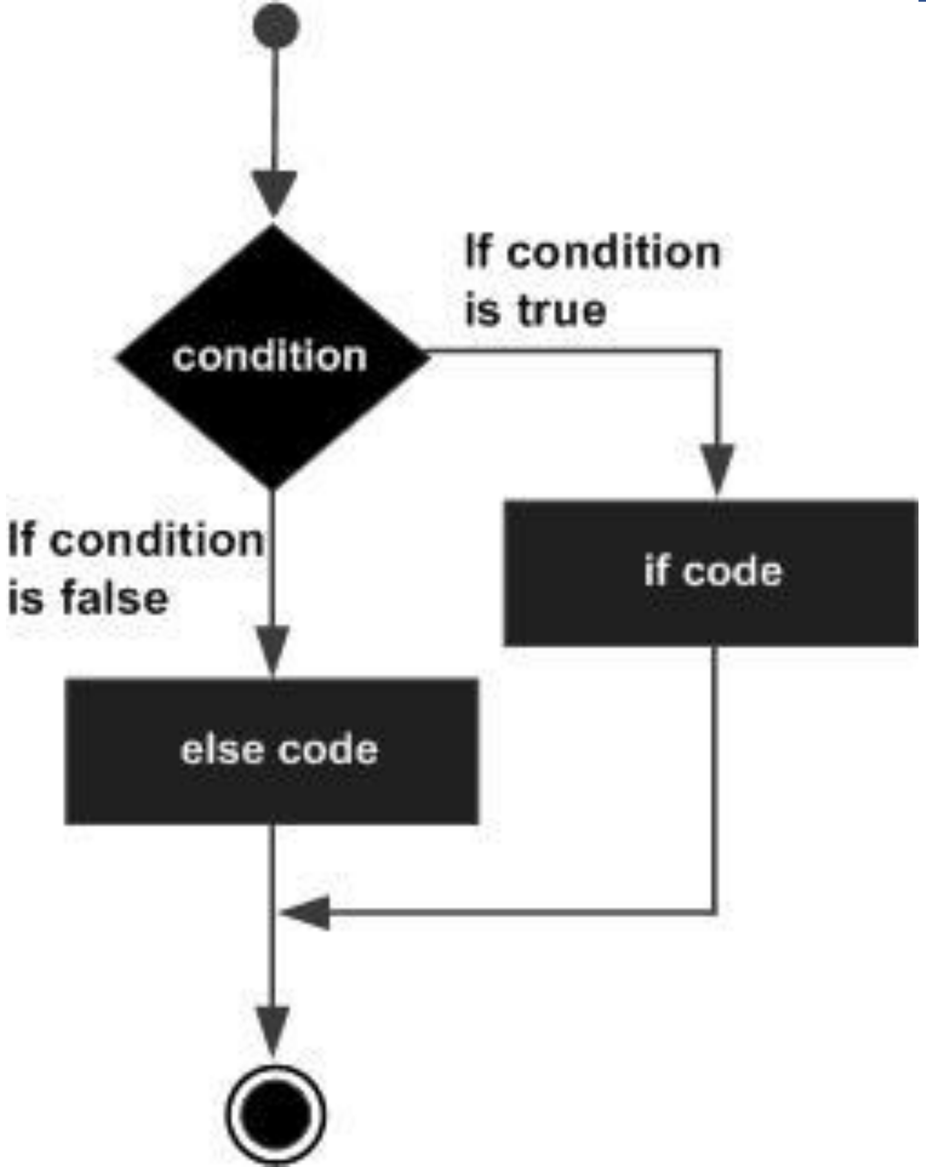
```
/*
 * Format: double exp(double x);
 * Purpose: calculates e raised to the power x
 * Library: math
 */
```

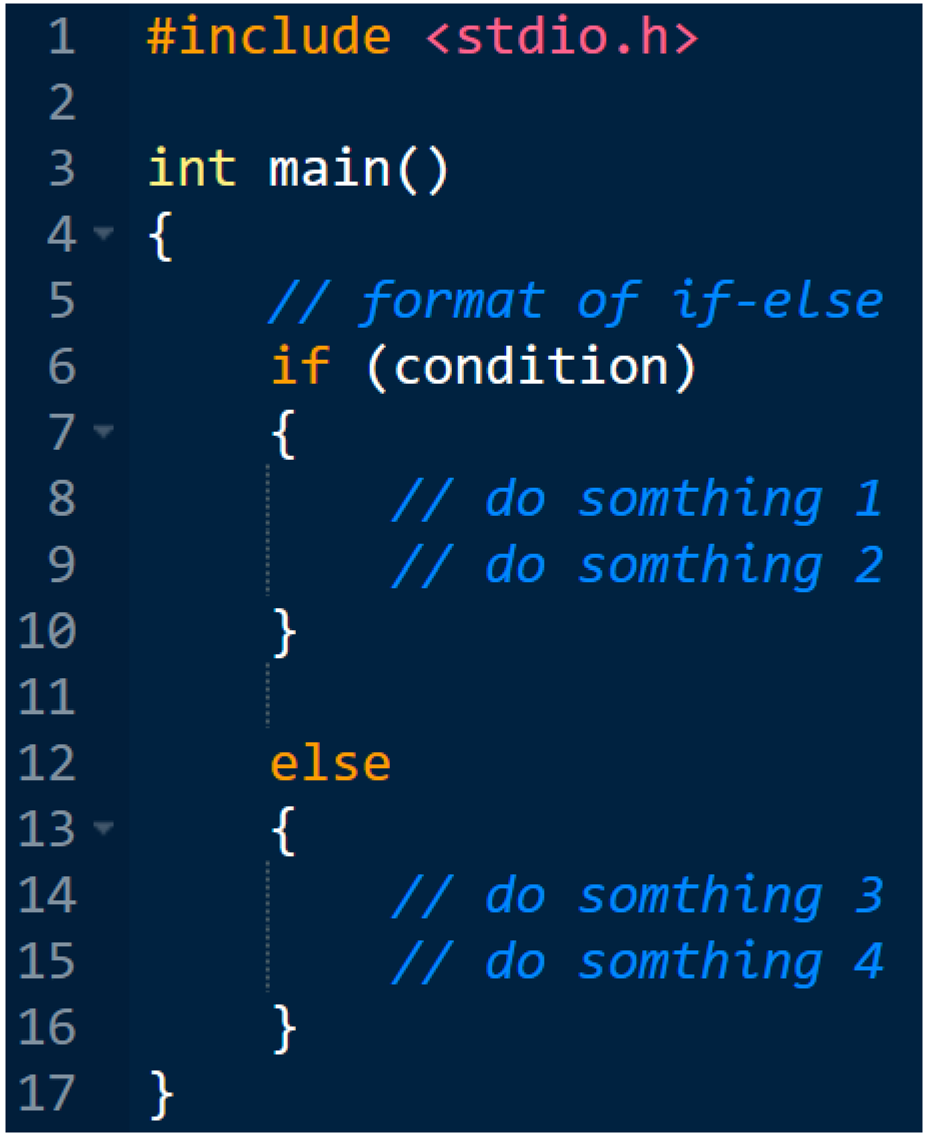
```
printf ("The exponential value of 2 = %f\n", exp(2));
printf ("The exponential value of 5 = %f\n", exp(5));
```

```
The exponential value of 2 = 7.389056
The exponential value of 5 = 148.413159
```

## IF - ELSE

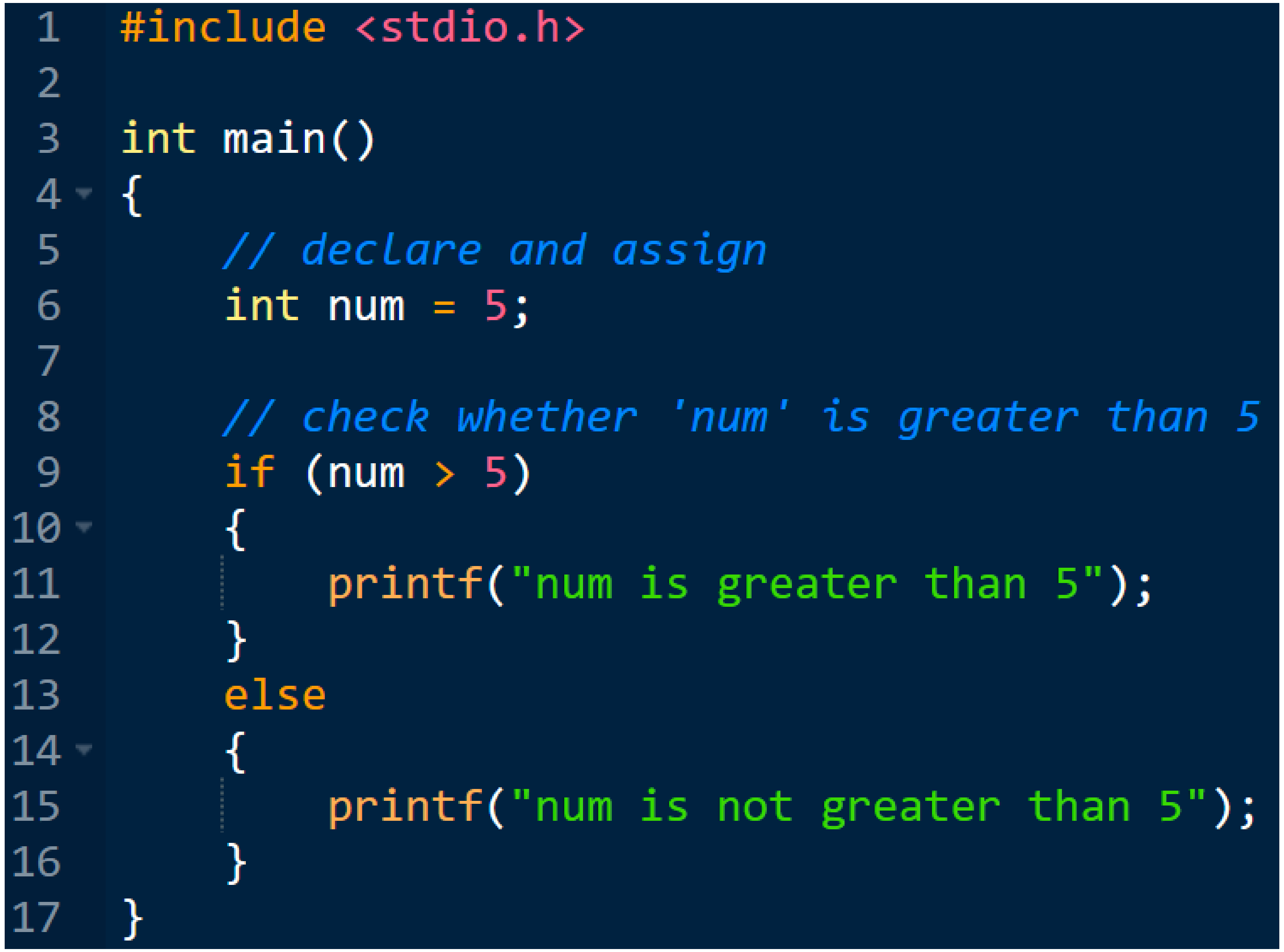
# IF - ELSE



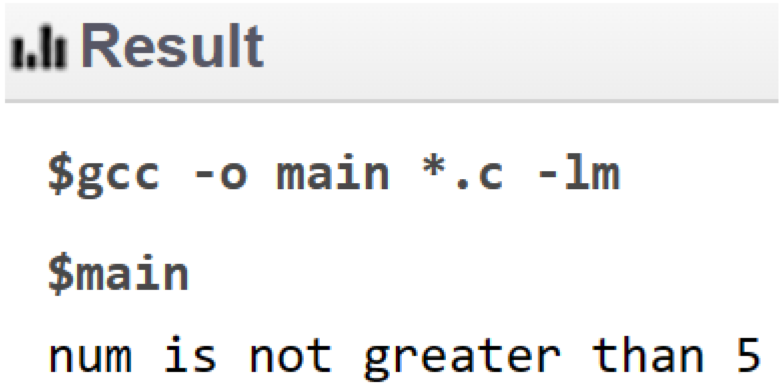


```
#include <stdio.h>

int main()
{
    // format of if-else
    if (condition)
    {
        // do somthing 1
        // do somthing 2
    }

    else
    {
        // do somthing 3
        // do somthing 4
    }
}
```

# IF - ELSE



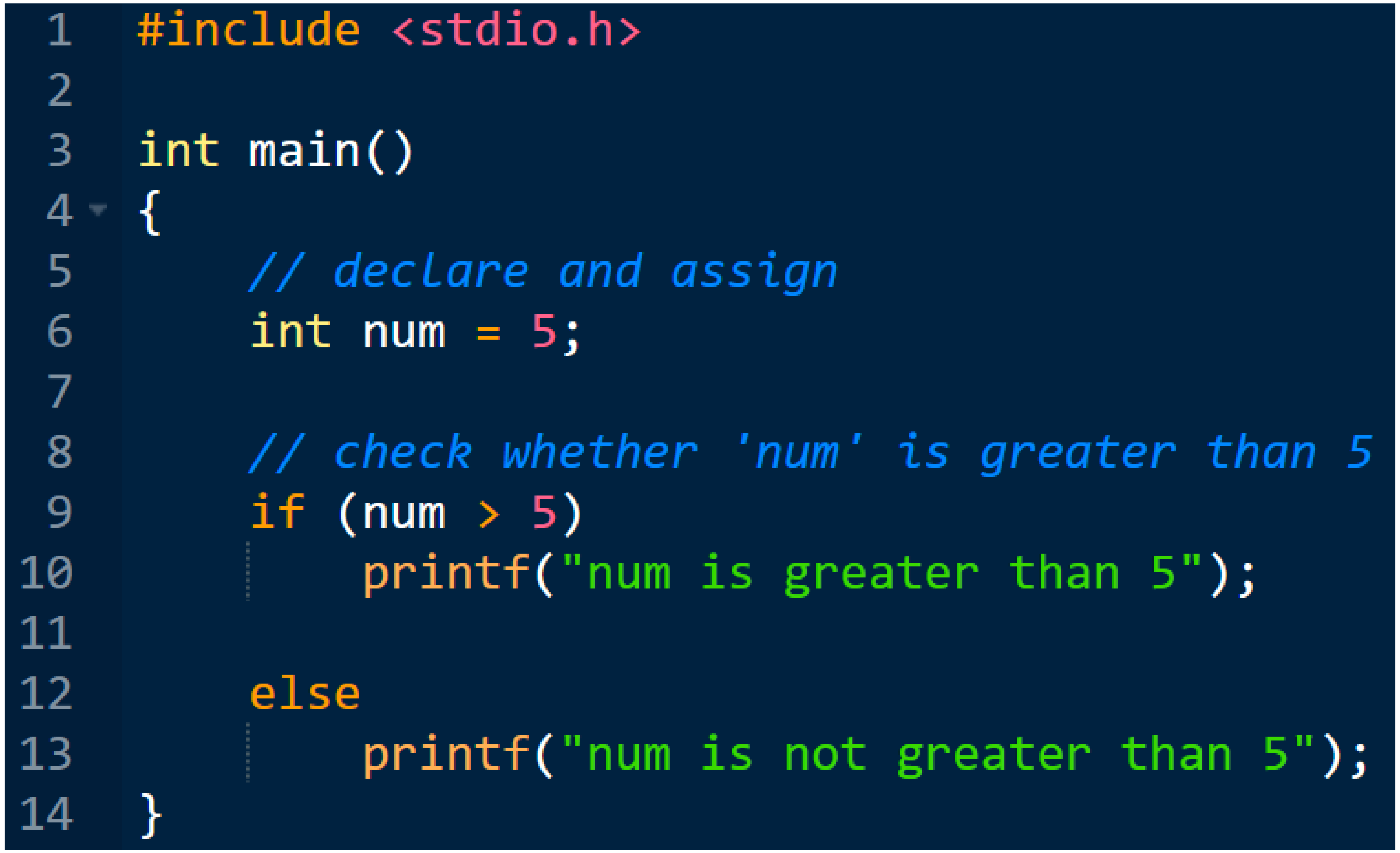
```c
#include <stdio.h>

int main()
{
    // declare and assign
    int num = 5;

    // check whether 'num' is greater than 5
    if (num > 5)
    {
        printf("num is greater than 5");
    }
    else
    {
        printf("num is not greater than 5");
    }
}
```



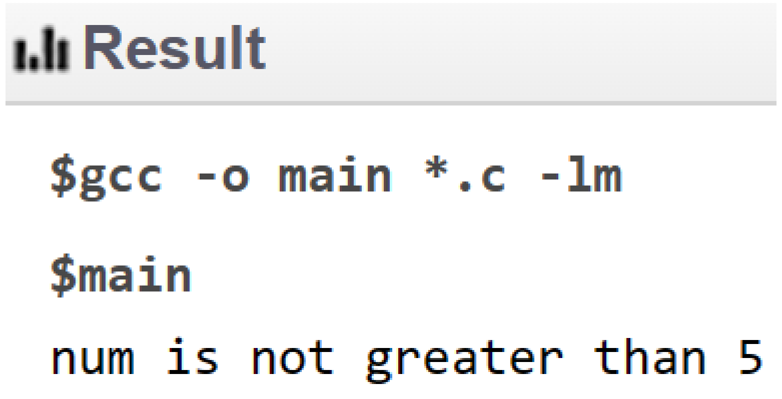
Result

```
$gcc -o main *.c -lm

$main
num is not greater than 5
```

# IF - ELSE

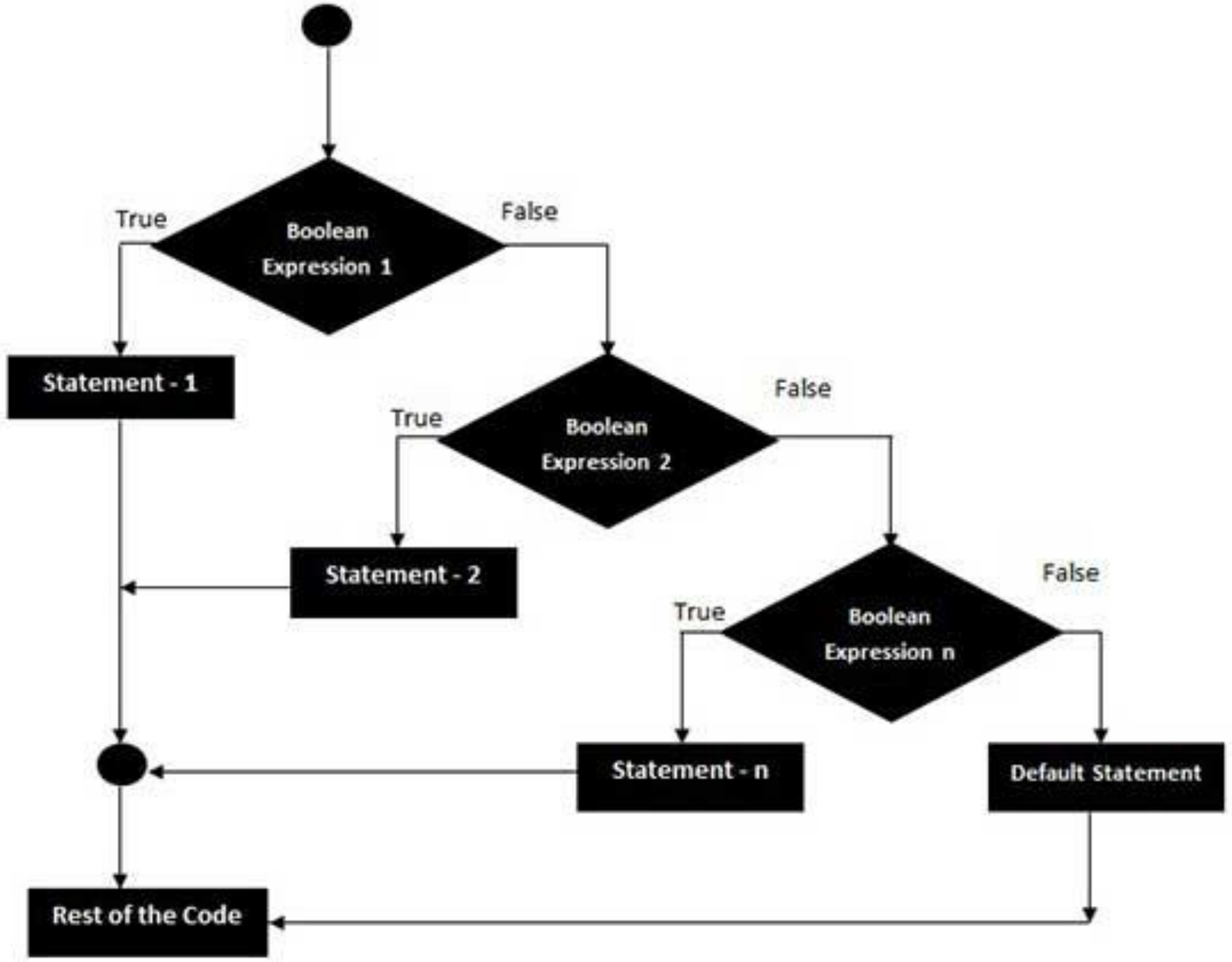


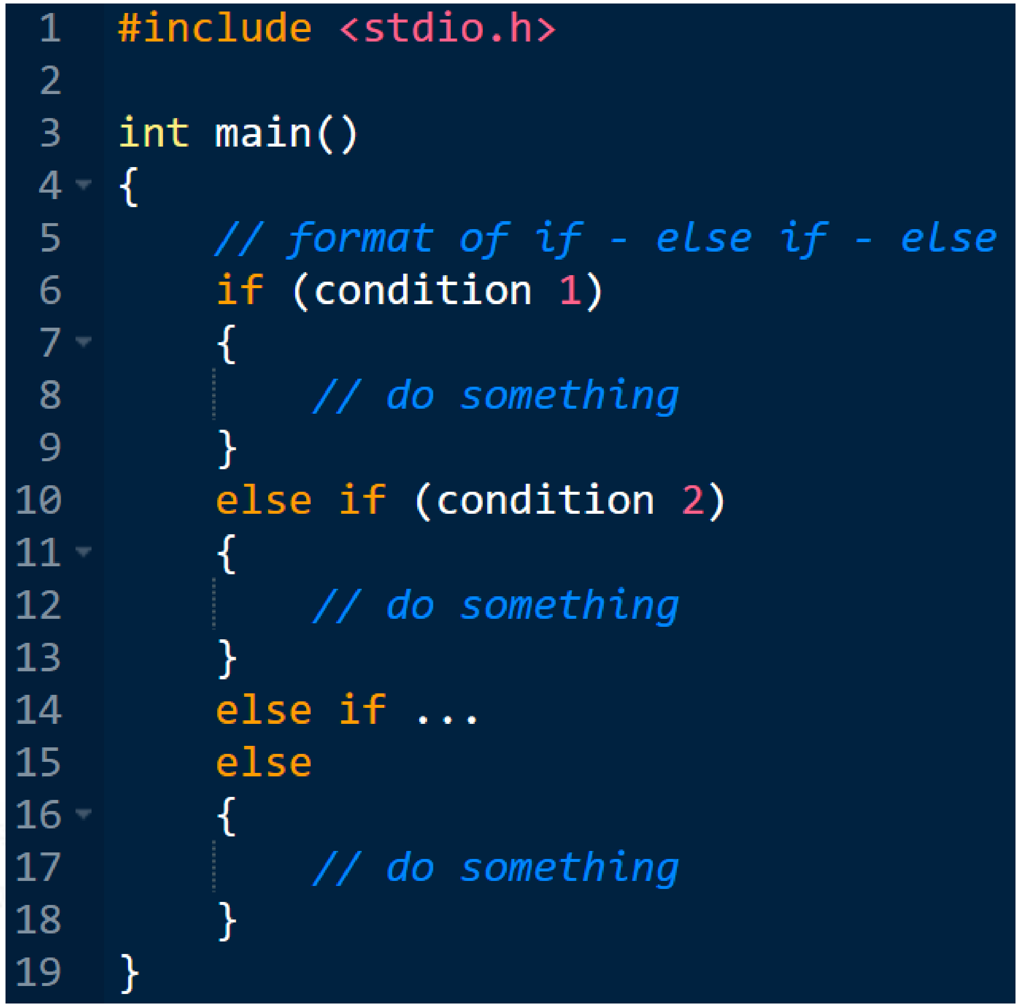
```c
#include <stdio.h>

int main()
{
    // declare and assign
    int num = 5;

    // check whether 'num' is greater than 5
    if (num > 5)
        printf("num is greater than 5");

    else
        printf("num is not greater than 5");
}
```



```
$gcc -o main *.c -lm

$main

num is not greater than 5
```

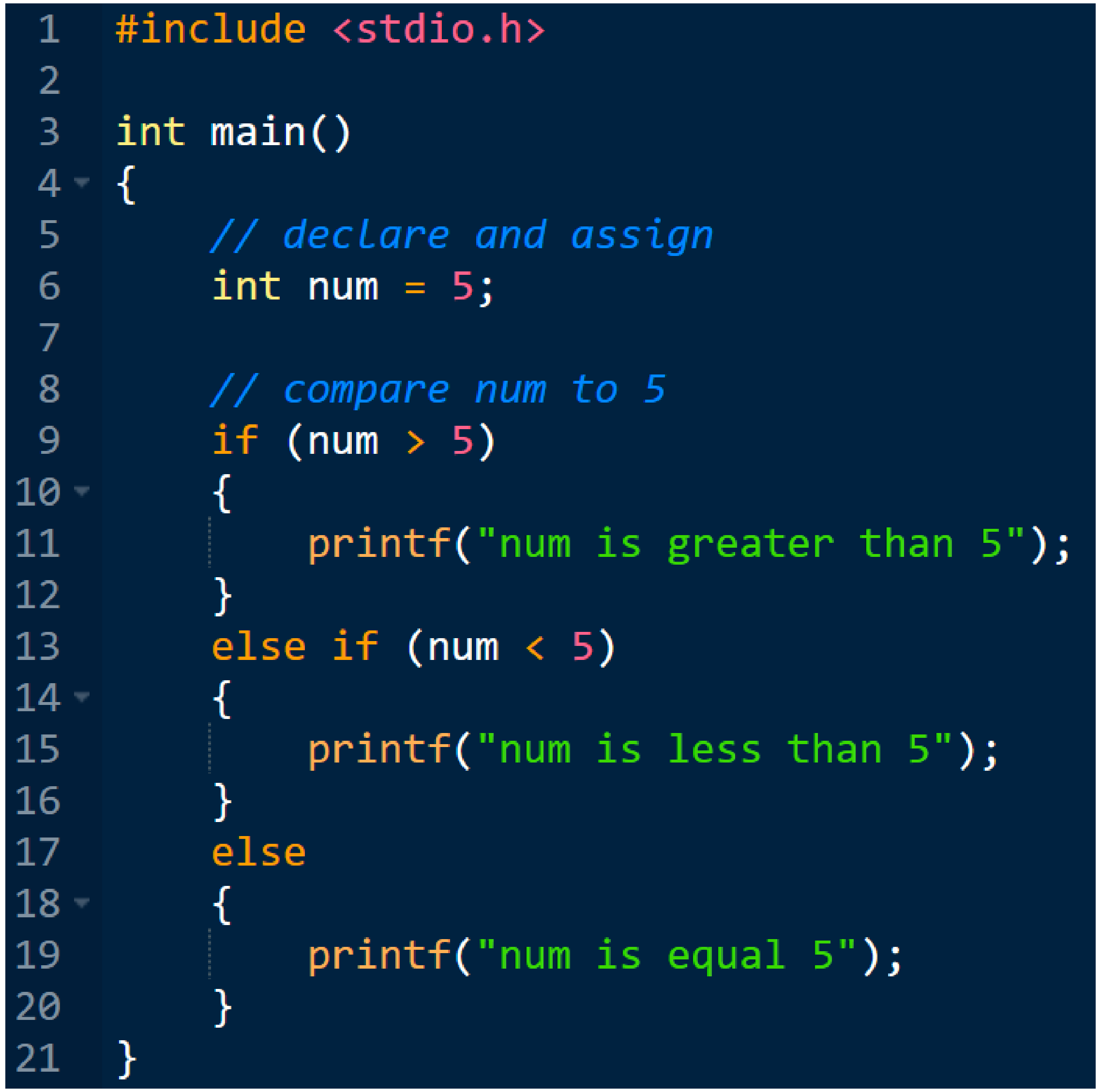
# IF - ELSE



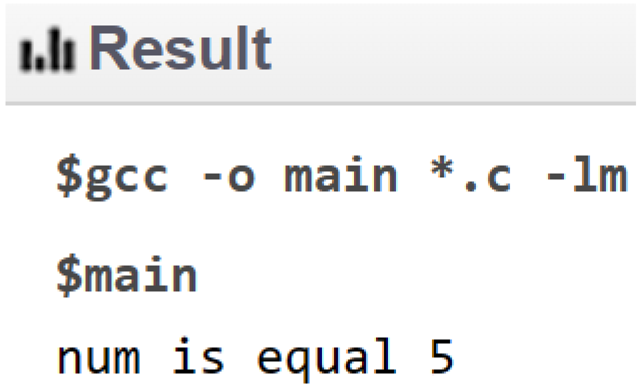


```
#include <stdio.h>

int main()
{
    // format of if - else if - else
    if (condition 1)
    {
        // do something
    }
    else if (condition 2)
    {
        // do something
    }
    else if ...
    else
    {
        // do something
    }
}
```

# IF - ELSE



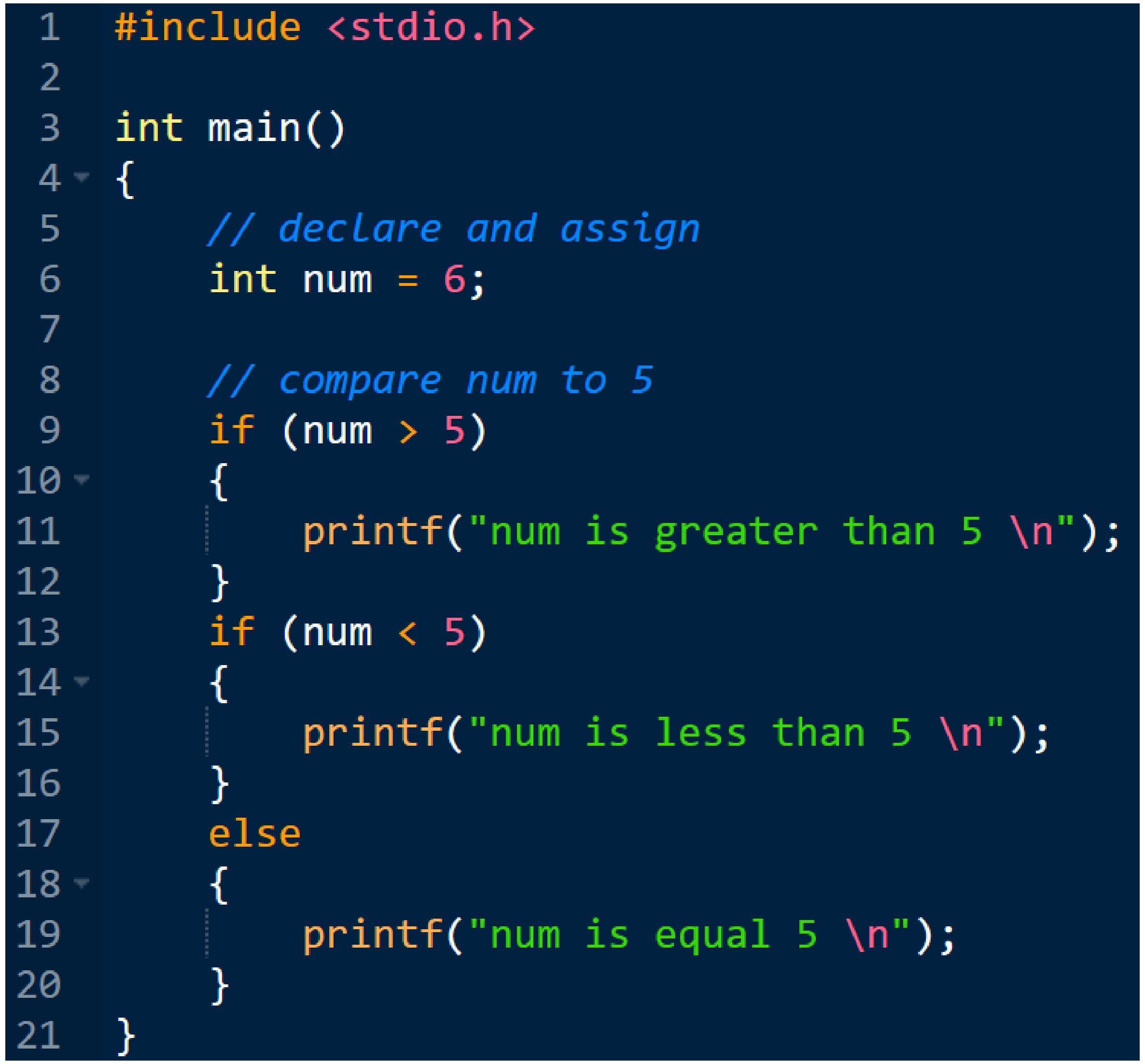
```c
#include <stdio.h>

int main()
{
    // declare and assign
    int num = 5;

    // compare num to 5
    if (num > 5)
    {
        printf("num is greater than 5");
    }
    else if (num < 5)
    {
        printf("num is less than 5");
    }
    else
    {
        printf("num is equal 5");
    }
}
```



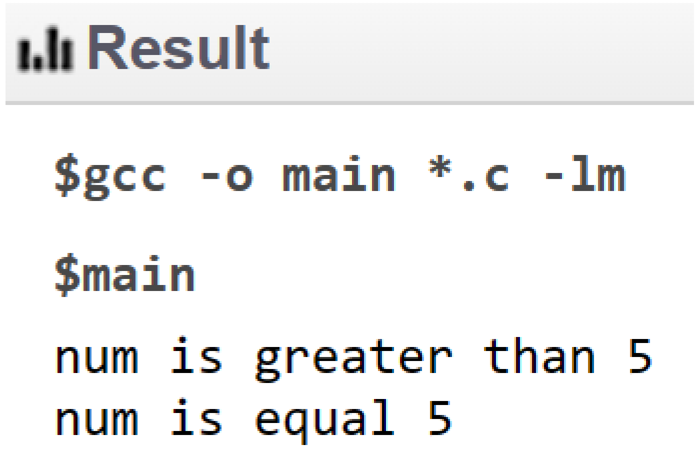
Result

```
$gcc -o main *.c -lm

$main
num is equal 5
```

# IF - ELSE



```c
#include <stdio.h>

int main()
{
    // declare and assign
    int num = 6;

    // compare num to 5
    if (num > 5)
    {
        printf("num is greater than 5 \n");
    }
    if (num < 5)
    {
        printf("num is less than 5 \n");
    }
    else
    {
        printf("num is equal 5 \n");
    }
}
```



```
$gcc -o main *.c -lm

$main

num is greater than 5
num is equal 5
```

# EXERCISE

**Question 1**: Write a program that gets an input number from a user and checks if the input is divisible by 4

gets an input number from a user $\rightarrow$ scanf

```
/*
 * Format: int scanf(const char * format, ... );
 * Purpose: reads data from stdin -> stores them (with parameter format) into the locations pointed by the additional arguments
 * Library: stdio
 */
```

checks if the input is divisible by 4 $\rightarrow$ modulo operator (%)

# EXERCISE

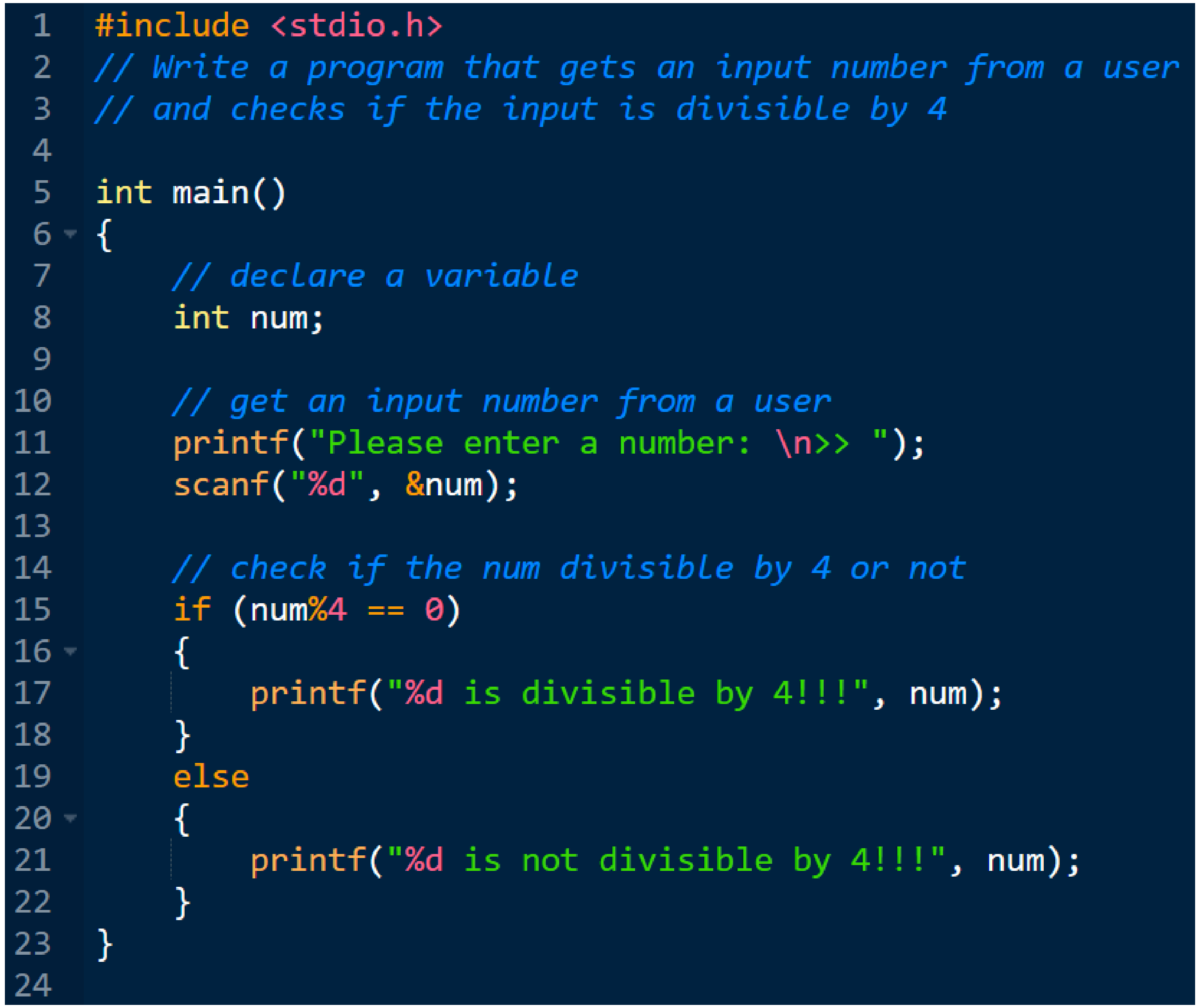
%

```
/* Modulo operation (%)
 * Format: a%b (with a and b are two integers)
 * Purpose: return the remainer of the division
 * Library: stdio
 */
```

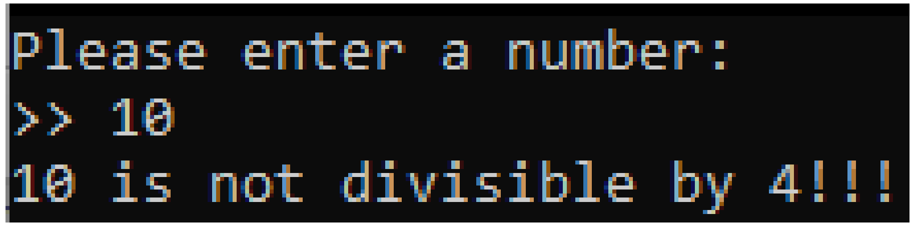
```
printf("5 / 2 = %d, remainer = %d \n", 5/2, 5%2);
printf("100 / 5 = %d, remainer = %d \n", 100/5, 100%5);
```

```
5 / 2 = 2, remainer = 1
100 / 5 = 20, remainer = 0
```

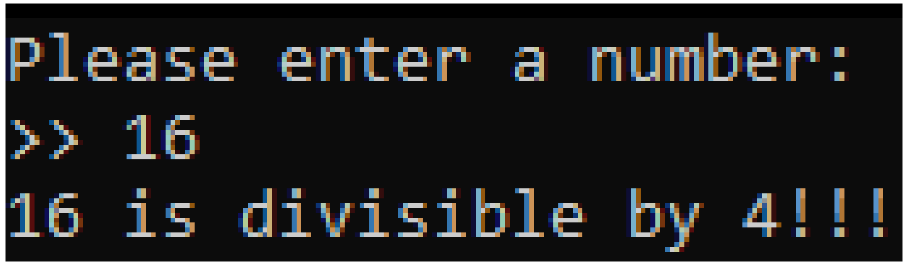
# EXERCISE



```c
#include <stdio.h>
// Write a program that gets an input number from a user
// and checks if the input is divisible by 4

int main()
{
    // declare a variable
    int num;

    // get an input number from a user
    printf("Please enter a number: \n>> ");
    scanf("%d", &num);

    // check if the num divisible by 4 or not
    if (num%4 == 0)
    {
        printf("%d is divisible by 4!!!", num);
    }
    else
    {
        printf("%d is not divisible by 4!!!", num);
    }
}

```



```
Please enter a number:
>> 10
10 is not divisible by 4!!!
```



```
Please enter a number:
>> 16
16 is divisible by 4!!!
```

# EXERCISE

**Question 3**: Write a program to check whether a number (entered from a user) is a multiple of 7s or not.

$$\begin{tikzcd}
\boxed{\text{a number (entered from a user)}} \arrow[r] & \boxed{\text{scanf}}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{\text{multiple of 7s or not.}} \arrow[r] & \boxed{\text{Divisible by 7}} \arrow[r] & \boxed{\%} \arrow[r] & \boxed{\textbf{Question 1}}
\end{tikzcd}$$

# EXERCISE

**Question 4**: Write a program to check whether a number (entered from a user) is even or odd.

$$\begin{tikzcd}
\boxed{\text{a number (entered from a user)}} \arrow[r] & \boxed{\text{scanf}}
\end{tikzcd}$$

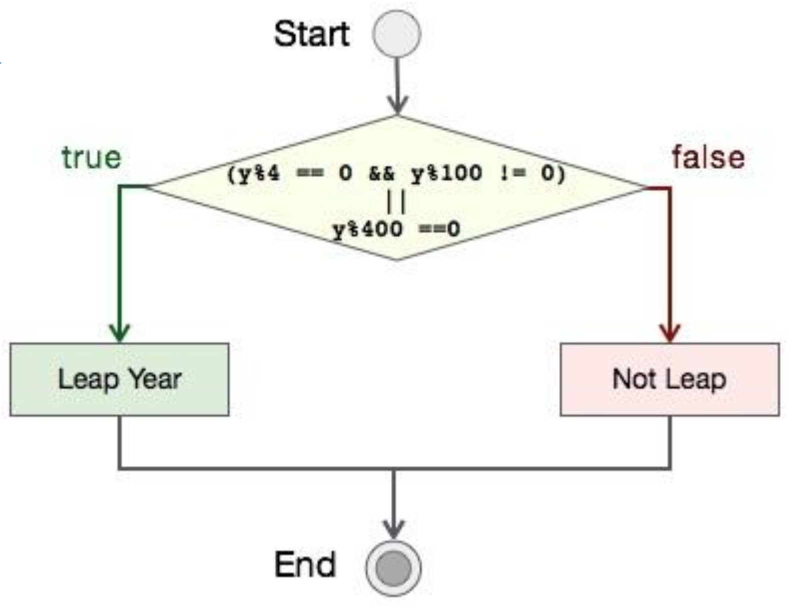
$$\begin{tikzcd}
\boxed{\text{even or odd.}} \arrow[r] & \boxed{\text{Divisible by 2}} \arrow[r] & \boxed{\%} \arrow[r] & \boxed{\textbf{Question 1}}
\end{tikzcd}$$

# EXERCISE

**Question 2**: Write a program to check whether a given year (entered from a user) is a leap year or not

given year (entered from a user) → scanf

leap year or not →



# EXERCISE

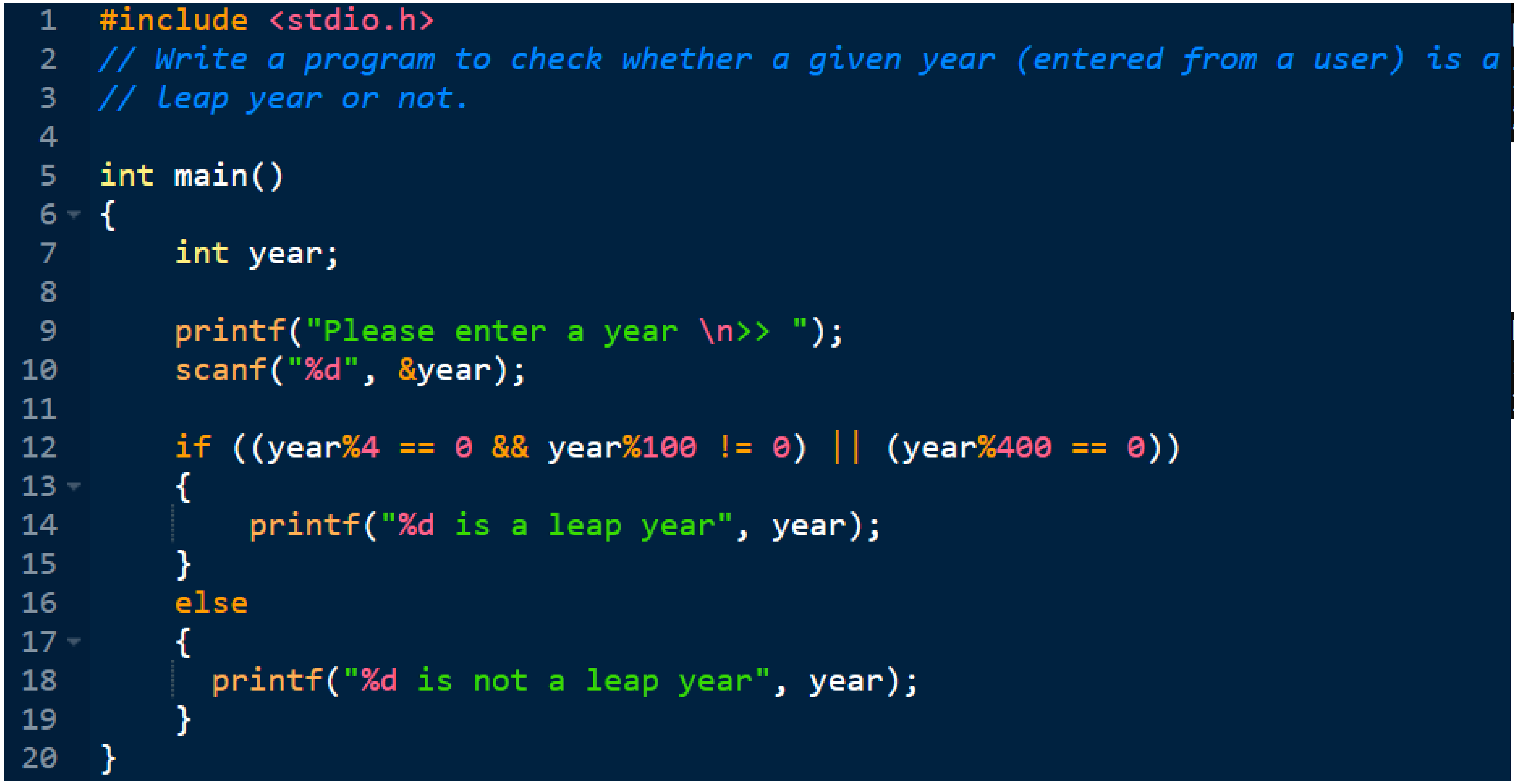
A year (calender) = 365 (days)
A year (physical) = 365.2421875 (days)

4 years (calender) = 4 * 365 = 1460 (days)
4 years (physical) = 4 * 365.2421875 = 1,460.96875 (days)
➔ Difference = 1,460.96875 – 1460 = 0.96875 (days)
➔ Add 1 day into every 4 yeas

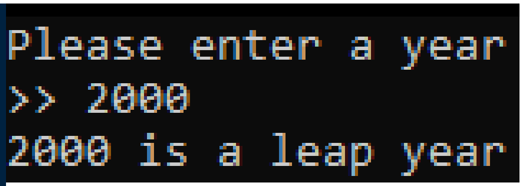
4 years (calender) = 4 * 365 + 1 = 1,461 (days)
➔ Difference = 1,461 - 1,460.96875 = 0.03125 (days)

A century (calender) = 1,461 * 25 = 36,525 (days)
A century (physical) = 1,460.96875 * 25 = 36,524.21875 (days)
➔ Difference = 36,524.21875 - 36,525 = -0.78125 (days)
➔ Difference (4 centuries) = -0.78125 * 4 = -3,125 (days)
➔ Minus 3 days into every 4 centuries
➔ In 400 years, there are 4 years divisible by 4 and 100, but 1 one them divisible by 400

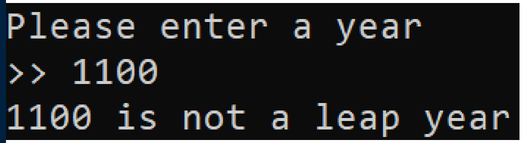
# EXERCISE



```c
#include <stdio.h>
// Write a program to check whether a given year (entered from a user) is a
// leap year or not.

int main()
{
    int year;

    printf("Please enter a year \n>> ");
    scanf("%d", &year);

    if ((year%4 == 0 && year%100 != 0) || (year%400 == 0))
    {
        printf("%d is a leap year", year);
    }
    else
    {
      printf("%d is not a leap year", year);
    }
}
```



```
Please enter a year
>> 2000
2000 is a leap year
```



```
Please enter a year
>> 1100
1100 is not a leap year
```

# EXERCISE

**Question 5**: Write a program to find solutions for a quadratic equation. The program requires a user to enter values for a, b, and c; and uses the printf function to print out its outputs.

find solutions for a quadratic equation.

If the discriminant > 0,

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

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

---

If the discriminant = 0,

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

---

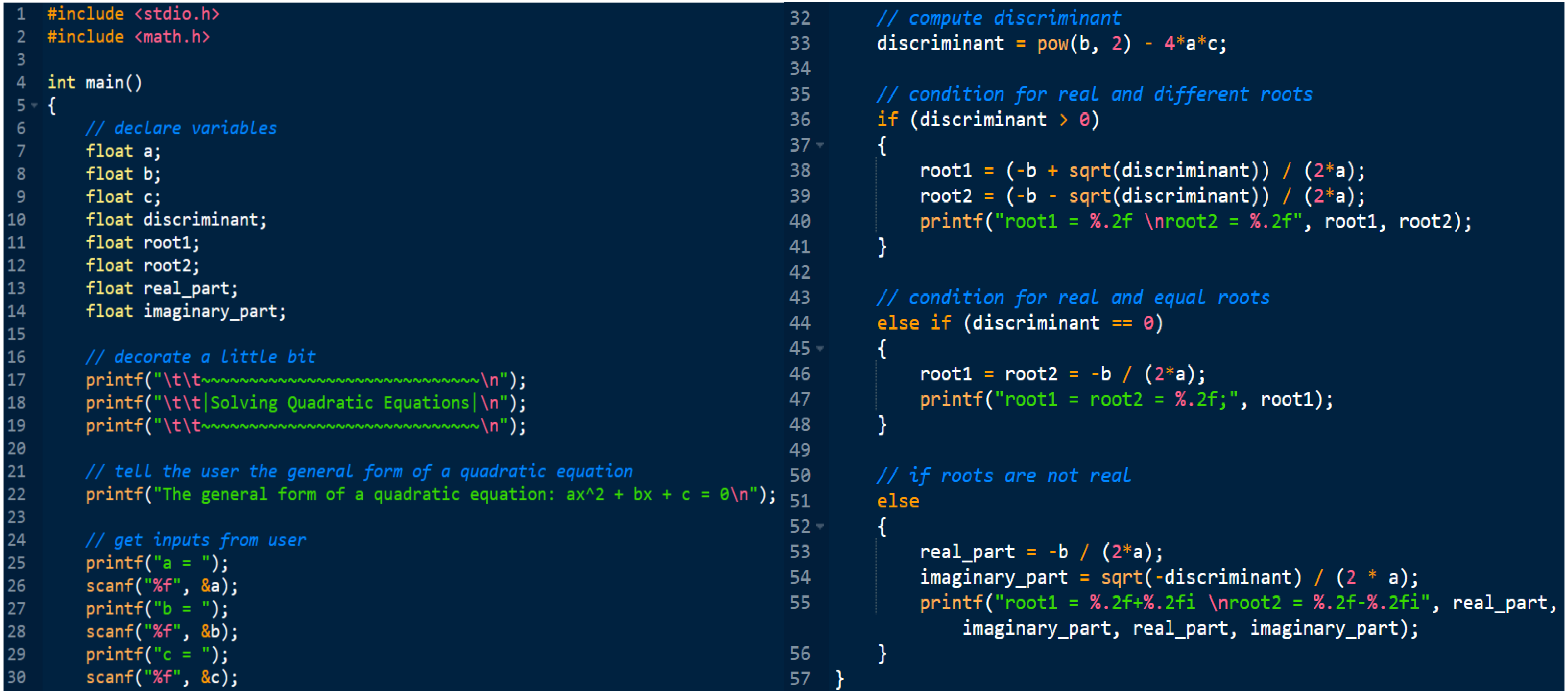
If the discriminant < 0,

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

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

enter values for a, b, and c; → scanf

# EXERCISE



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

int main()
{
    // declare variables
    float a;
    float b;
    float c;
    float discriminant;
    float root1;
    float root2;
    float real_part;
    float imaginary_part;

    // decorate a little bit
    printf("\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");
    printf("\t\t|Solving Quadratic Equations|\n");
    printf("\t\t~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");

    // tell the user the general form of a quadratic equation
    printf("The general form of a quadratic equation: ax^2 + bx + c = 0\n");

    // get inputs from user
    printf("a = ");
    scanf("%f", &a);
    printf("b = ");
    scanf("%f", &b);
    printf("c = ");
    scanf("%f", &c);
    // compute discriminant
    discriminant = pow(b, 2) - 4*a*c;

    // condition for real and different roots
    if (discriminant > 0)
    {
        root1 = (-b + sqrt(discriminant)) / (2*a);
        root2 = (-b - sqrt(discriminant)) / (2*a);
        printf("root1 = %.2f \nroot2 = %.2f", root1, root2);
    }

    // condition for real and equal roots
    else if (discriminant == 0)
    {
        root1 = root2 = -b / (2*a);
        printf("root1 = root2 = %.2f;", root1);
    }

    // if roots are not real
    else
    {
        real_part = -b / (2*a);
        imaginary_part = sqrt(-discriminant) / (2 * a);
        printf("root1 = %.2f+%.2fi \nroot2 = %.2f-%.2fi", real_part,
            imaginary_part, real_part, imaginary_part);
    }
}
```

# EXERCISE

## ❖ Tanh function

$$\tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

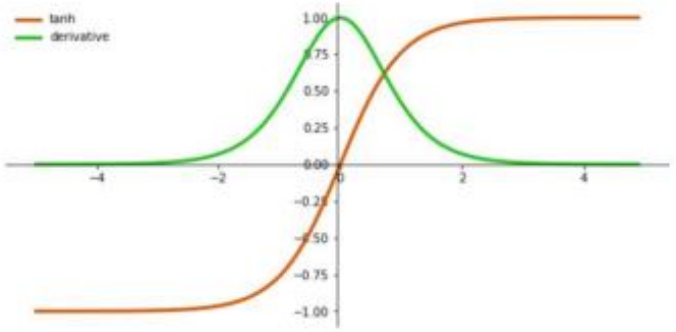
data =

| | | | | |
|---|---|---|---|---|
| 1 | 5 | -4 | 3 | -2 |

data_a = tanh(data)

data_a =

| | | | | |
|---|---|---|---|---|
| 0.761 | 0.999 | -0.999 | 0.995 | -0.964 |



## ❖ Sigmoid function

$$\text{sigmoid}(x) = \frac{1}{1+e^{-x}}$$

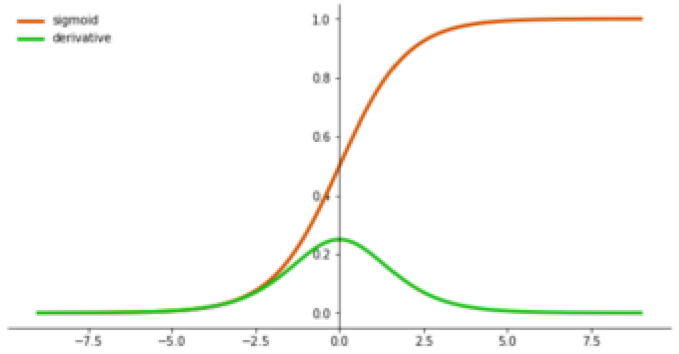
data =

| | | | | |
|---|---|---|---|---|
| 1 | 5 | -4 | 3 | -2 |

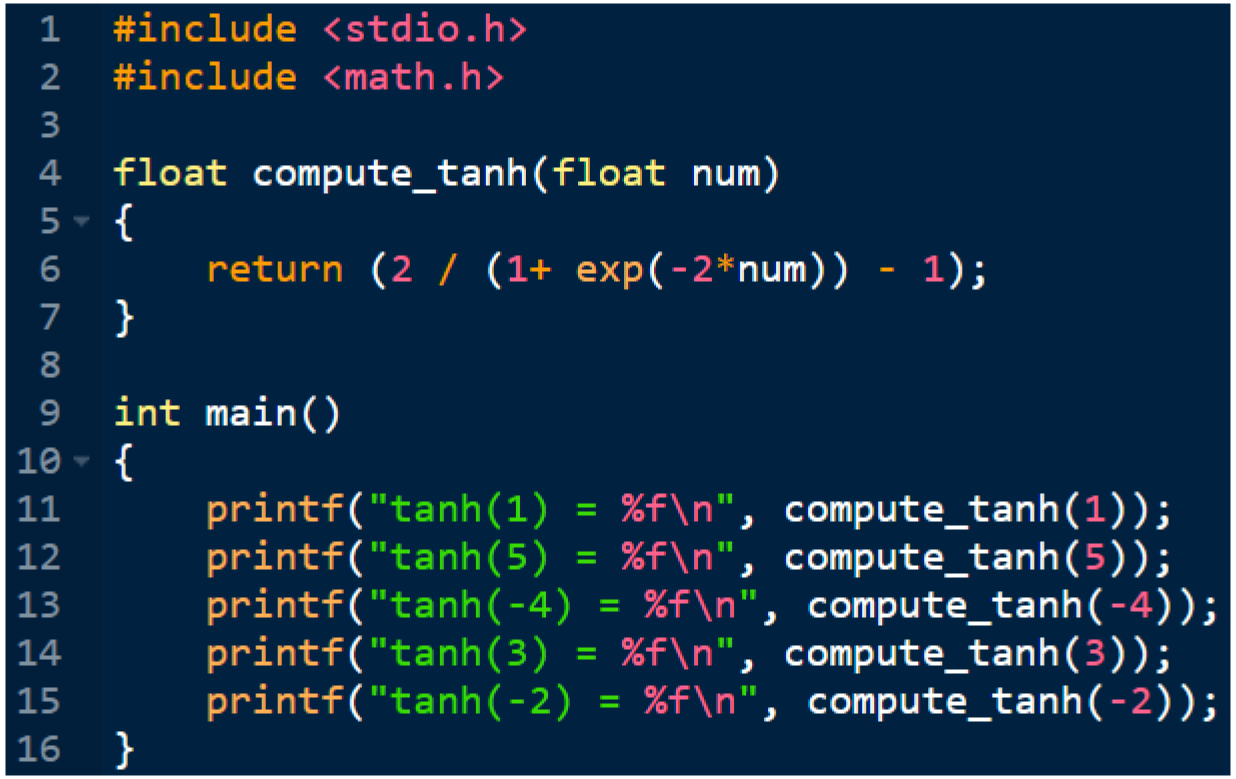
data_a = sigmoid(data)

data_a =

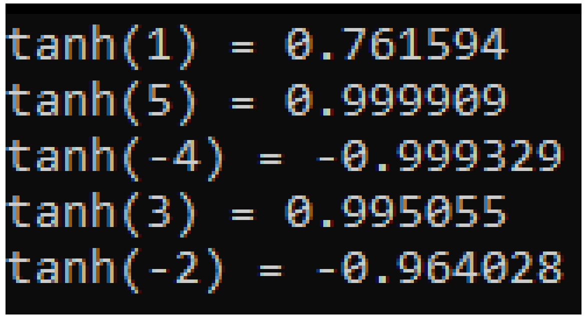
| | | | | |
|---|---|---|---|---|
| 0.731 | 0.993 | 0.017 | 0.95 | 0.119 |



# EXERCISE

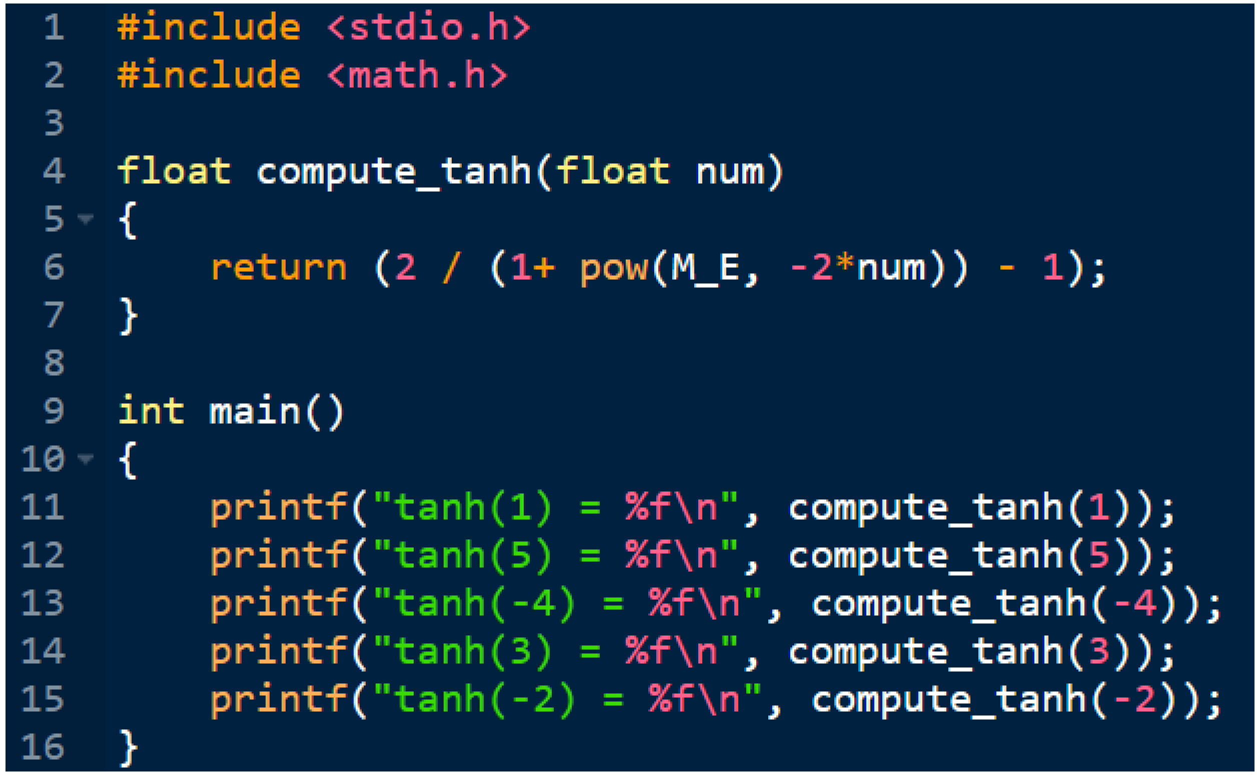


```c
#include <stdio.h>
#include <math.h>

float compute_tanh(float num)
{
    return (2 / (1+ exp(-2*num)) - 1);
}

int main()
{
    printf("tanh(1) = %f\n", compute_tanh(1));
    printf("tanh(5) = %f\n", compute_tanh(5));
    printf("tanh(-4) = %f\n", compute_tanh(-4));
    printf("tanh(3) = %f\n", compute_tanh(3));
    printf("tanh(-2) = %f\n", compute_tanh(-2));
}
```

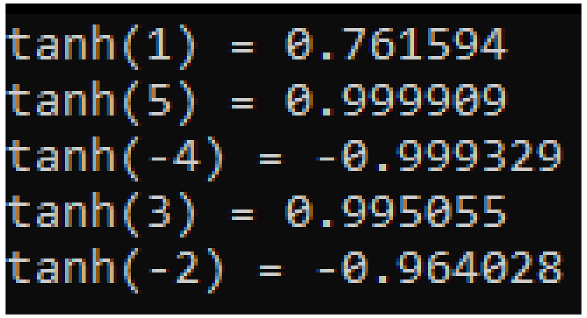


```
tanh(1) = 0.761594
tanh(5) = 0.999909
tanh(-4) = -0.999329
tanh(3) = 0.995055
tanh(-2) = -0.964028
```

# EXERCISE

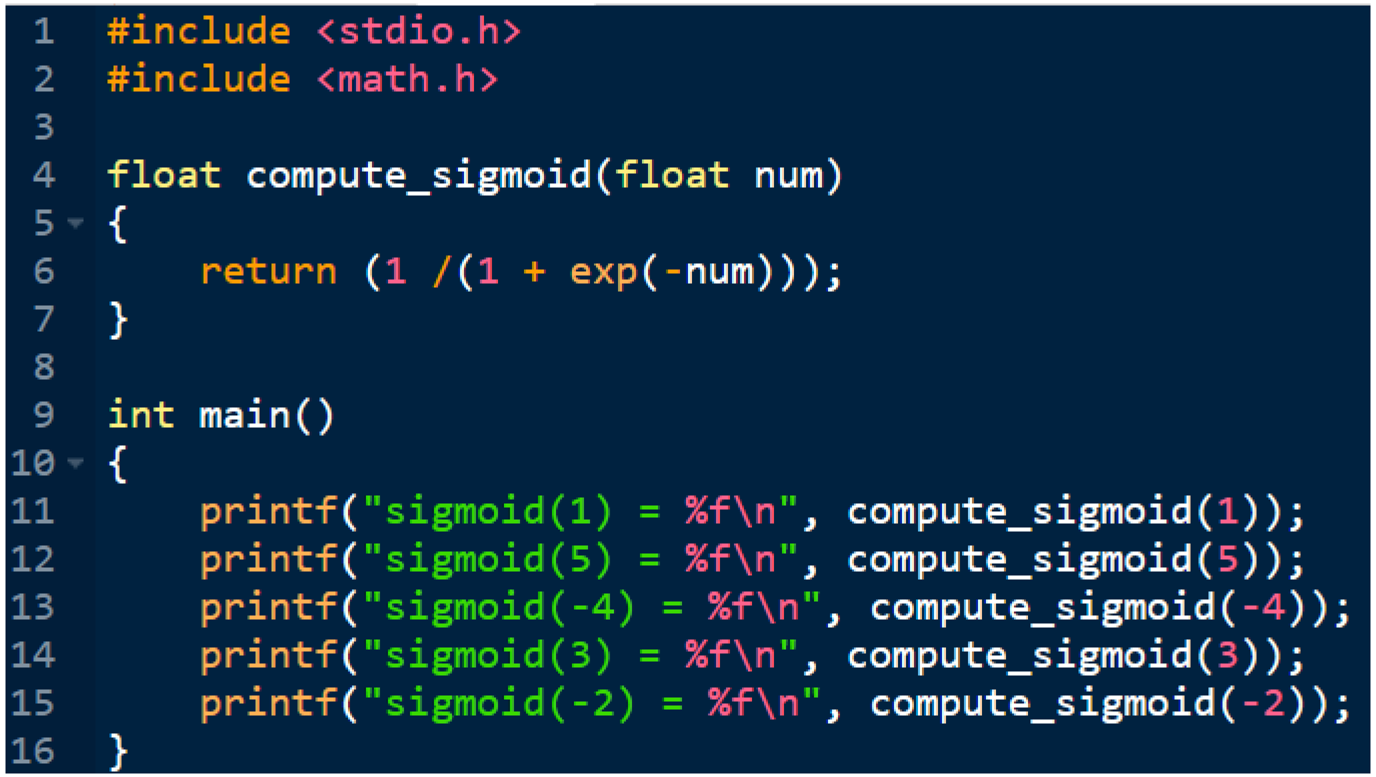


```c
#include <stdio.h>
#include <math.h>

float compute_tanh(float num)
{
    return (2 / (1+ pow(M_E, -2*num)) - 1);
}

int main()
{
    printf("tanh(1) = %f\n", compute_tanh(1));
    printf("tanh(5) = %f\n", compute_tanh(5));
    printf("tanh(-4) = %f\n", compute_tanh(-4));
    printf("tanh(3) = %f\n", compute_tanh(3));
    printf("tanh(-2) = %f\n", compute_tanh(-2));
}
```

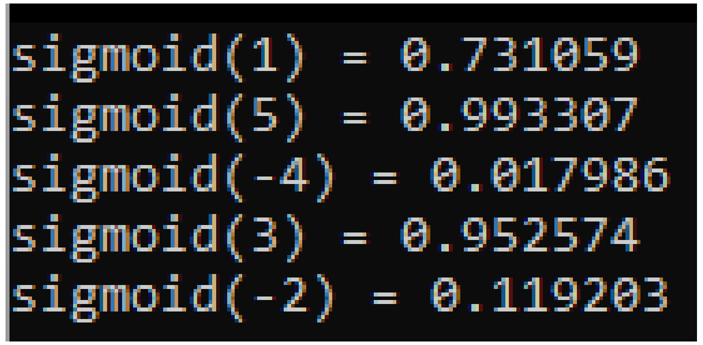


```
tanh(1) = 0.761594
tanh(5) = 0.999909
tanh(-4) = -0.999329
tanh(3) = 0.995055
tanh(-2) = -0.964028
```

# EXERCISE



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

float compute_sigmoid(float num)
{
    return (1 /(1 + exp(-num)));
}

int main()
{
    printf("sigmoid(1) = %f\n", compute_sigmoid(1));
    printf("sigmoid(5) = %f\n", compute_sigmoid(5));
    printf("sigmoid(-4) = %f\n", compute_sigmoid(-4));
    printf("sigmoid(3) = %f\n", compute_sigmoid(3));
    printf("sigmoid(-2) = %f\n", compute_sigmoid(-2));
}
```



```
sigmoid(1) = 0.731059
sigmoid(5) = 0.993307
sigmoid(-4) = 0.017986
sigmoid(3) = 0.952574
sigmoid(-2) = 0.119203
```