

1. Write a C program to find the sum and average of three numbers.

Program:-

```
#include<stdio.h>
int main() {
    float num1, num2, num3, sum, average;

    /* Ask for input */
    printf("Enter first number: ");
    scanf("%f", &num1);
    printf("Enter second number: ");
    scanf("%f", &num2);
    printf("Enter third number: ");
    scanf("%f", &num3);

    /* Calculate sum */
    sum = num1 + num2 + num3;

    /* Calculate average */
    average = sum / 3.0;

    /* Display results */
    printf("Sum = %.2f\n", sum);
    printf("Average = %.2f\n", average);

    return 0;
}
```

Output:-

```
Enter first number: 40
Enter second number: 30
Enter third number: 56
Sum = 126.00
Average = 42.00

-----
Process exited after 5.085 seconds with return value 0
Press any key to continue . . .
```

2. Write a C program to find the sum of individual digits of a given positive integer.

Program:-

```
#include <stdio.h>
int sumOfDigits(int num) {
    int sum = 0;
    while (num != 0) {
        sum += num % 10;
        num /= 10;
    }
    return sum;
}

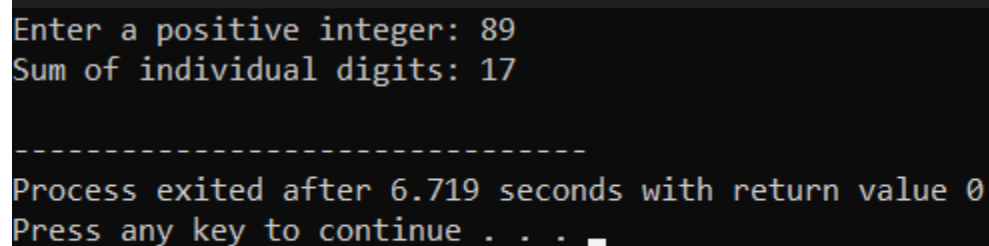
int main() {
    int num;

    printf("Enter a positive integer: ");
    scanf("%d", &num);

    printf("Sum of individual digits: %d\n", sumOfDigits(num));

    return 0;
}
```

Output:-

A screenshot of a terminal window showing the execution of the C program. The user enters the number 89, and the program outputs the sum of its digits as 17. Below the output, a separator line is shown, followed by a message indicating the process exited after 6.719 seconds with a return value of 0, and a prompt to press any key to continue.

```
Enter a positive integer: 89
Sum of individual digits: 17

-----
Process exited after 6.719 seconds with return value 0
Press any key to continue . . .
```

3. Write a C program to generate the first n terms of the Fibonacci sequence.

Program:-

```
#include <stdio.h>

void printFibonacci(int n) {
    int t1 = 0, t2 = 1, nextTerm = 0;

    for (int i = 1; i <= n; ++i) {
        printf("%d, ", t1);
        nextTerm = t1 + t2;
        t1 = t2;
        t2 = nextTerm;
    }
}

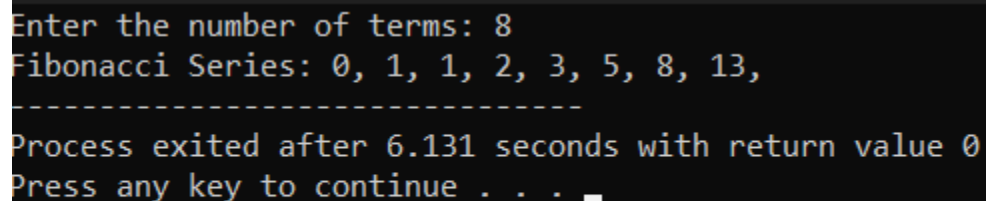
int main() {
    int n;

    printf("Enter the number of terms: ");
    scanf("%d", &n);

    printf("Fibonacci Series: ");
    printFibonacci(n);

    return 0;
}
```

Output:-



```
Enter the number of terms: 8
Fibonacci Series: 0, 1, 1, 2, 3, 5, 8, 13,
-----
Process exited after 6.131 seconds with return value 0
Press any key to continue . . .
```

4. Write a C program to generate prime numbers between 1 to n.

Program:-

```
#include <stdio.h>
#include <stdbool.h>
bool isPrime(int num) {
    if (num <= 1) {
        return false;
    }
    for (int i = 2; i * i <= num; ++i) {
        if (num % i == 0) {
            return false;
        }
    }
    return true;
}

void printPrimes(int n) {
    for (int i = 2; i <= n; ++i) {
        if (isPrime(i)) {
            printf("%d ", i);
        }
    }
}

int main() {
    int n;

    printf("Enter a positive integer: ");
    scanf("%d", &n);

    printf("Prime numbers between 1 to %d: ", n);
    printPrimes(n);

    return 0;
}
```

Output:-

```
Enter a positive integer: 20
Prime numbers between 1 to 20: 2 3 5 7 11 13 17 19
-----
Process exited after 6.719 seconds with return value 0
Press any key to continue . . .
```

5. Write a C program to check whether a given number is an Armstrong number or not.

Program:-

```
#include <stdio.h>
#include <math.h>
int main() {
    int num, originalNum, remainder, n = 0;
    float result = 0.0;

    printf("Enter a number: ");
    scanf("%d", &num);

    originalNum = num;

    // store the number of digits of num in n
    for (;originalNum != 0; n++) {
        originalNum /= 10;
    }

    originalNum = num;

    // compute the value of each digit raised to the power of n and add it to result
    for (;originalNum != 0; originalNum /= 10) {
        remainder = originalNum % 10;
        result += pow(remainder, n);
    }

    // check if result is same as original number
    if ((int)result == num)
        printf("%d is an Armstrong number.", num);
    else
        printf("%d is not an Armstrong number.", num);

    return 0;
}
```

Output:-

```
Enter a number: 521
521 is not an Armstrong number.
-----
Process exited after 3.927 seconds with return value 0
Press any key to continue . . . ■
```

6. Write a C program to evaluate the algebraic expression $(ax+b)/(ax-b)$.

Program:-

```
#include <stdio.h>
int main() {
    float a, x, b, result;

    printf("Enter the value of a: ");
    scanf("%f", &a);

    printf("Enter the value of x: ");
    scanf("%f", &x);

    printf("Enter the value of b: ");
    scanf("%f", &b);

    if (a*x - b == 0) {
        printf("Error! Division by zero is not allowed.\n");
        return 1;
    }
    result = (a*x + b) / (a*x - b);
    printf("The result of the expression (ax+b)/(ax-b) is: %f\n", result);
    return 0;
}
```

Output:-

```
Enter the value of a: 1
Enter the value of x: 23
Enter the value of b: 4
The result of the expression (ax+b)/(ax-b) is: 1.421053

-----
Process exited after 5.229 seconds with return value 0
Press any key to continue . . .
```

7. Write a C program to check if the given number is a perfect number.

Program:-

```
#include <stdio.h>

int main() {
    int num, sum = 0, i;

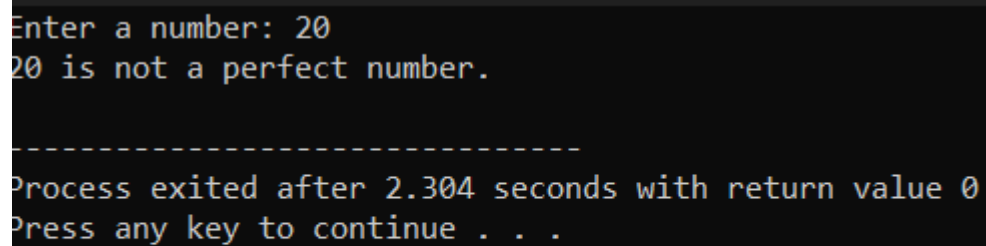
    printf("Enter a number: ");
    scanf("%d", &num);

    for (i = 1; i < num; i++) {
        if (num % i == 0) {
            sum += i;
        }
    }

    if (sum == num) {
        printf("%d is a perfect number.\n", num);
    } else {
        printf("%d is not a perfect number.\n", num);
    }

    return 0;
}
```

Output:-

A screenshot of a terminal window showing the execution of the C program. The user enters '20' when prompted 'Enter a number:'. The program outputs '20 is not a perfect number.' followed by a dashed line separator. Below the separator, it shows 'Process exited after 2.304 seconds with return value 0' and 'Press any key to continue . . .' on the next line.

```
Enter a number: 20
20 is not a perfect number.
-----
Process exited after 2.304 seconds with return value 0
Press any key to continue . . .
```


8. Write a C program to check if a given number is a strong number.

Program:-

```
#include <stdio.h>
#include <math.h>
int factorial(int n) {
    int fact = 1;
    for (int i = 2; i <= n; i++) {
        fact *= i;
    }
    return fact;
}

int main() {
    int num, sum = 0, temp, digit;

    printf("Enter a number: ");
    scanf("%d", &num);

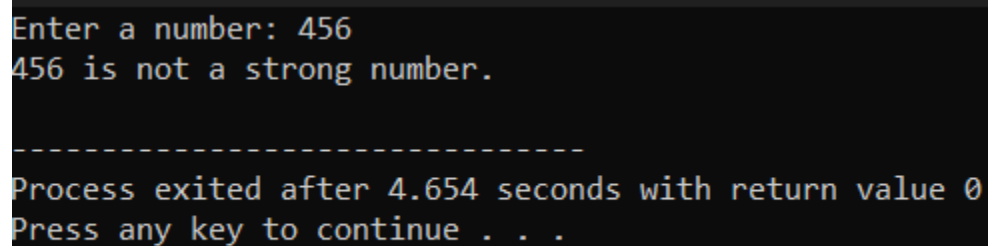
    temp = num;

    while (temp > 0) {
        digit = temp % 10;
        sum += factorial(digit);
        temp /= 10;
    }

    if (sum == num) {
        printf("%d is a strong number.\n", num);
    } else {
        printf("%d is not a strong number.\n", num);
    }

    return 0;
}
```

Output:-

A screenshot of a terminal window showing the execution of the C program. The user enters '456' when prompted 'Enter a number:'. The program outputs '456 is not a strong number.'. Below this, a separator line of dashes is shown, followed by the message 'Process exited after 4.654 seconds with return value 0' and 'Press any key to continue . . .'.

```
Enter a number: 456
456 is not a strong number.

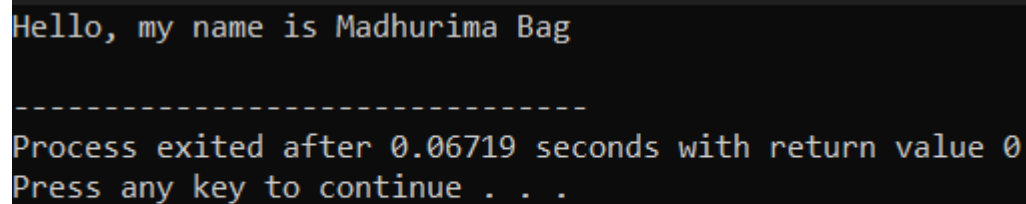
-----
Process exited after 4.654 seconds with return value 0
Press any key to continue . . .
```

9. Write a program to print your name without using any semicolons in the program.

Program:-

```
#include <stdio.h>
int main()
{
    if(printf("Hello, my name is Madhurima Bag\n"))
    {
        return 0;
    }
}
```

Output:-

A screenshot of a terminal window showing the output of the program. The first line is "Hello, my name is Madhurima Bag". The second line is a dashed line "-----". The third line is "Process exited after 0.06719 seconds with return value 0". The fourth line is "Press any key to continue . . .".

```
Hello, my name is Madhurima Bag
-----
Process exited after 0.06719 seconds with return value 0
Press any key to continue . . .
```

10. Write a program to convert temperatures in Celsius to Fahrenheit and vice-versa.

Program:-

```
#include <stdio.h>

void celsiusToFahrenheit(float celsius) {
    float fahrenheit = (celsius * 9.0 / 5.0) + 32.0;
    printf("%.2f Celsius is equal to %.2f Fahrenheit\n", celsius, fahrenheit);
}

void fahrenheitToCelsius(float fahrenheit) {
    float celsius = (fahrenheit - 32.0) * 5.0 / 9.0;
    printf("%.2f Fahrenheit is equal to %.2f Celsius\n", fahrenheit, celsius);
}

int main() {
    int choice;
    float temperature;

    printf("Temperature Conversion Program\n");
    printf("1. Celsius to Fahrenheit\n");
    printf("2. Fahrenheit to Celsius\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

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

    switch (choice) {
        case 1:
            celsiusToFahrenheit(temperature);
            break;
        case 2:
            fahrenheitToCelsius(temperature);
            break;
        default:
            printf("Invalid choice\n");
    }

    return 0;
}
```

Output:-

```
Temperature Conversion Program
1. Celsius to Fahrenheit
2. Fahrenheit to Celsius
Enter your choice: 1
Enter the temperature: 45
45.00 Celsius is equal to 113.00 Fahrenheit

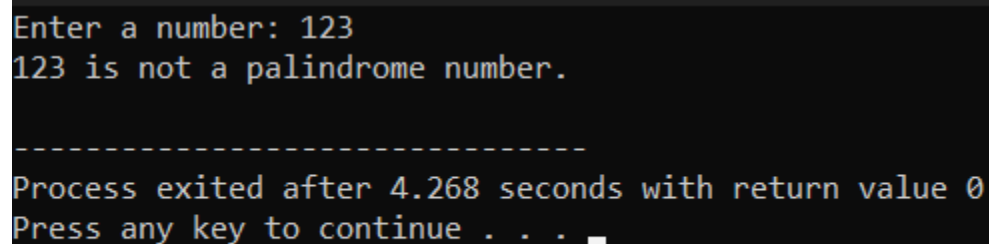
-----
Process exited after 7.774 seconds with return value 0
Press any key to continue . . .
```

11. Write a C program to check whether a number is a palindrome or not.

Program:-

```
#include <stdio.h>
int reverse(int num) {
    int reversed = 0;
    while (num != 0) {
        int digit = num % 10;
        reversed = reversed * 10 + digit;
        num /= 10;
    }
    return reversed;
}
int isPalindrome(int num) {
    return num == reverse(num);
}
int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);
    if (isPalindrome(num)) {
        printf("%d is a palindrome number.\n", num);
    } else {
        printf("%d is not a palindrome number.\n", num);
    }
    return 0;
}
```

Output:-

A screenshot of a terminal window showing the execution of the C program. The user enters the number 123. The program outputs "123 is not a palindrome number." followed by a dashed line separator. Below the separator, it shows "Process exited after 4.268 seconds with return value 0" and "Press any key to continue . . .".

```
Enter a number: 123
123 is not a palindrome number.
-----
Process exited after 4.268 seconds with return value 0
Press any key to continue . . .
```

12. Write a C program to find the maximum between two numbers.

Program:-

```
#include <stdio.h>
int main() {
    int num1, num2;

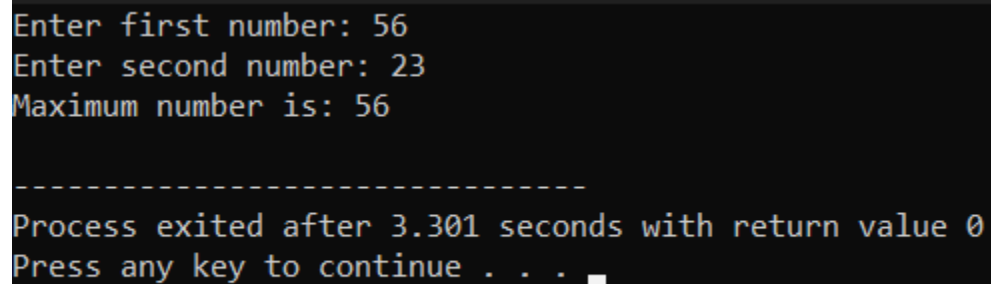
    printf("Enter first number: ");
    scanf("%d", &num1);

    printf("Enter second number: ");
    scanf("%d", &num2);

    if (num1 > num2) {
        printf("Maximum number is: %d\n", num1);
    } else {
        printf("Maximum number is: %d\n", num2);
    }

    return 0;
}
```

Output:-

A screenshot of a terminal window showing the execution of the C program. The user enters 56 for the first number and 23 for the second number. The program outputs "Maximum number is: 56". Below this, a separator line of dashes is shown, followed by the message "Process exited after 3.301 seconds with return value 0" and "Press any key to continue . . .".

```
Enter first number: 56
Enter second number: 23
Maximum number is: 56

-----
Process exited after 3.301 seconds with return value 0
Press any key to continue . . .
```

13. Write a C program to find the maximum between three numbers.

Program:-

```
#include <stdio.h>
int main() {
    int num1, num2, num3;

    printf("Enter first number: ");
    scanf("%d", &num1);

    printf("Enter second number: ");
    scanf("%d", &num2);

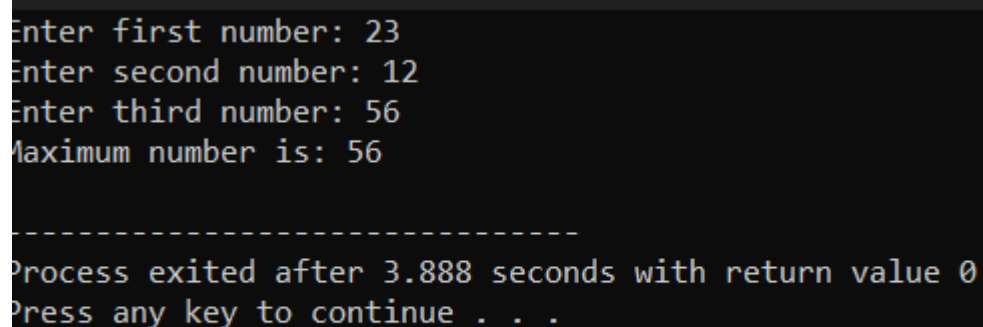
    printf("Enter third number: ");
    scanf("%d", &num3);

    int max = (num1 > num2) ? num1 : num2;
    max = (max > num3) ? max : num3;

    printf("Maximum number is: %d\n", max);

    return 0;
}
```

Output:-

A screenshot of a terminal window showing the execution of the C program. The user enters three numbers: 23, 12, and 56. The program outputs that the maximum number is 56. Below this, a dashed line separates the program output from the system message: "Process exited after 3.888 seconds with return value 0" and "Press any key to continue . . .".

```
Enter first number: 23
Enter second number: 12
Enter third number: 56
Maximum number is: 56

-----
Process exited after 3.888 seconds with return value 0
Press any key to continue . . .
```

14. Write a C program to check whether a number is negative, positive, or zero.

Program:-

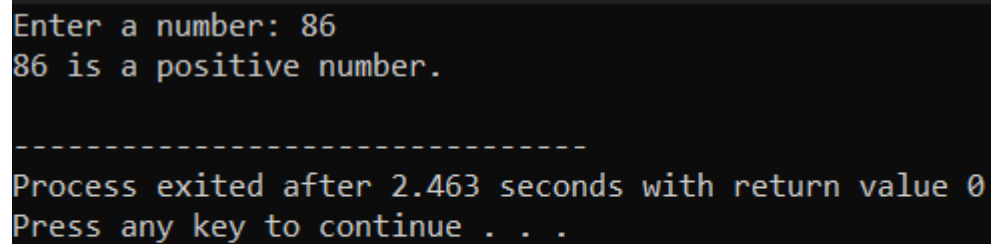
```
#include <stdio.h>
int main() {
    int num;

    printf("Enter a number: ");
    scanf("%d", &num);

    printf("%d is %s.\n", num, (num > 0) ? "a positive number" : (num < 0) ? "a negative number" : "zero");

    return 0;
}
```

Output:-

A screenshot of a terminal window showing the output of the C program. The text is as follows:

```
Enter a number: 86
86 is a positive number.

-----
Process exited after 2.463 seconds with return value 0
Press any key to continue . . .
```


15. Write a C program to check whether a number is divisible by 5 and 11 or not within the range of 100 to 500.

Program:-

```
#include <stdio.h>
int is_divisible(int num) {
    return (num % 5 == 0 && num % 11 == 0);
}

int main() {
    int i;

    printf("Numbers between 100 and 500 that are divisible by 5 and 11:\n");

    for (i = 100; i <= 500; i++) {
        if (is_divisible(i)) {
            printf("%d\n", i);
        }
    }

    return 0;
}
```

Output:-

```
Numbers between 100 and 500 that are divisible by 5 and 11:
110
165
220
275
330
385
440
495

-----
Process exited after 0.0759 seconds with return value 0
Press any key to continue . . .
```

16. Write a C program to check whether a number is even or odd.

Program:-

```
#include <stdio.h>

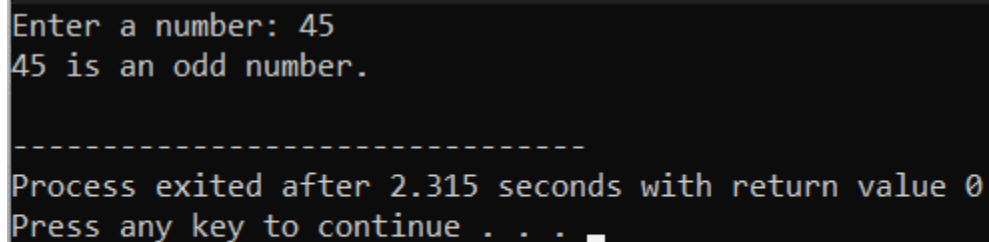
int main() {
    int num;

    printf("Enter a number: ");
    scanf("%d", &num);

    if (num % 2 == 0) {
        printf("%d is an even number.\n", num);
    } else {
        printf("%d is an odd number.\n", num);
    }

    return 0;
}
```

Output:-

A screenshot of a terminal window showing the output of the C program. The text is as follows:

```
Enter a number: 45
45 is an odd number.

-----
Process exited after 2.315 seconds with return value 0
Press any key to continue . . .
```

17. Write a C program to check whether a year is a leap year or not.

Program:-

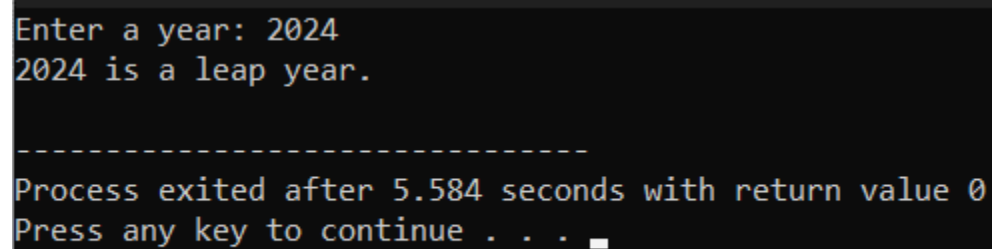
```
#include <stdio.h>
int main() {
    int year;

    printf("Enter a year: ");
    scanf("%d", &year);

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

    return 0;
}
```

Output:-

A screenshot of a terminal window showing the output of the C program. The text is as follows:

```
Enter a year: 2024
2024 is a leap year.

-----
Process exited after 5.584 seconds with return value 0
Press any key to continue . . .
```

18. Write a C program to check whether a character is alphabet or not.

Program:-

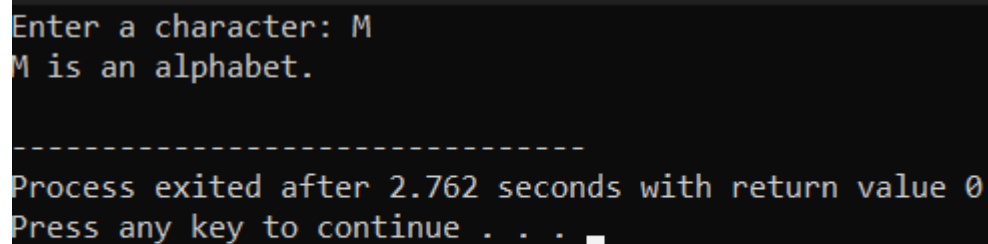
```
#include <stdio.h>
int main() {
    char ch;

    printf("Enter a character: ");
    scanf(" %c", &ch);

    if ((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z')) {
        printf("%c is an alphabet.\n", ch);
    } else {
        printf("%c is not an alphabet.\n", ch);
    }

    return 0;
}
```

Output:-

A screenshot of a terminal window showing the execution of the C program. The user enters 'M' as a character. The program outputs 'M is an alphabet.' followed by a dashed line separator. Below the separator, it shows 'Process exited after 2.762 seconds with return value 0' and 'Press any key to continue . . .'.

```
Enter a character: M
M is an alphabet.
-----
Process exited after 2.762 seconds with return value 0
Press any key to continue . . .
```

19. Write a C program to input any alphabet and check whether it is a vowel or consonant.

Program:-

```
#include <stdio.h>
#include <ctype.h>
int main() {
    char c;
    int isLowercaseVowel, isUppercaseVowel;

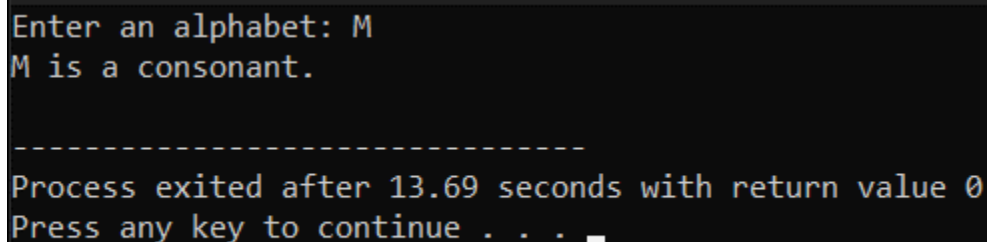
    printf("Enter an alphabet: ");
    scanf("%c", &c);

    // Check if the character is a lowercase vowel
    isLowercaseVowel = (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u');
    // Check if the character is an uppercase vowel
    isUppercaseVowel = (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U');

    // Check if the character is an alphabet
    if (!isalpha(c)) {
        printf("Error! Non-alphabetic character.\n");
    } else if (isLowercaseVowel || isUppercaseVowel) {
        printf("%c is a vowel.\n", c);
    } else {
        printf("%c is a consonant.\n", c);
    }

    return 0;
}
```

Output:-

A screenshot of a terminal window showing the execution of the C program. The user enters 'M' as an alphabet, and the program outputs 'M is a consonant.'. Below this, a separator line of dashes is shown, followed by the message 'Process exited after 13.69 seconds with return value 0' and 'Press any key to continue . . .'.

```
Enter an alphabet: M
M is a consonant.

-----
Process exited after 13.69 seconds with return value 0
Press any key to continue . . .
```

20. Write a C program to input any character and check whether it is an alphabet, digit, or special character.

Program:-

```
#include <stdio.h>
#include <ctype.h>

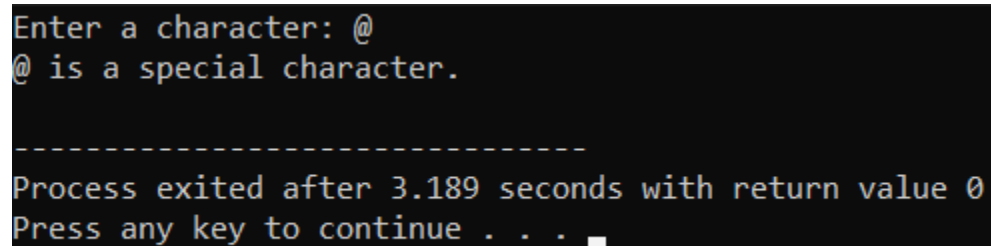
int main() {
    char ch;

    printf("Enter a character: ");
    scanf(" %c", &ch);

    if (isalpha(ch)) {
        printf("%c is an alphabet.\n", ch);
    } else if (isdigit(ch)) {
        printf("%c is a digit.\n", ch);
    } else {
        printf("%c is a special character.\n", ch);
    }

    return 0;
}
```

Output:-



```
Enter a character: @
@ is a special character.

-----
Process exited after 3.189 seconds with return value 0
Press any key to continue . . .
```