1. Time Conversion (seconds, minutes, hours)

```
def time_conversion(value, from_unit, to_unit):
    conversion_factors = {
        "sec": 1,
        "min": 60,
        "hour": 3600,
    }

    if from_unit not in conversion_factors or to_unit not in conversion_factors:
        return "Invalid units"

    # Convert the value to seconds first
    value_in_seconds = value * conversion_factors[from_unit]
    # Convert from seconds to the target unit
    return value_in_seconds / conversion_factors[to_unit]

# Example usage
print(time_conversion(3600, "sec", "hour"))
```

Expected Output: 1.0

2. Reverse the digits in a number

```
def reverse_number(number):
    sign = -1 if number < 0 else 1
    reversed_num = int(str(abs(number))[::-1])
    return sign * reversed_num

# Example usage
print(reverse_number(12345))
```

Expected Output: 54321

3. Find the LCM of 2 Numbers

```
public class LCM {

    public static int gcd(int a, int b) {
        if (b == 0) return a;
        return gcd(b, a % b);
    }

    public static int lcm(int a, int b) {
        return (a * b) / gcd(a, b);
    }

    public static void main(String[] args) {
        int a = 12, b = 15;
        System.out.println(lcm(a, b));
    }
}
```

Expected Output: 60

4. Cross Product of Two 3D Vectors

```
public class VectorOperations {

    public static int[] crossProduct(int[] a, int[] b) {
        if (a.length != 3 || b.length != 3) {
            throw new IllegalArgumentException("Cross product is only defined for 3D vectors");
        }
        int[] crossProduct = new int[3];
        crossProduct[0] = a[1] * b[2] - a[2] * b[1];
        crossProduct[1] = a[2] * b[0] - a[0] * b[2];
        crossProduct[2] = a[0] * b[1] - a[1] * b[0];
        return crossProduct;
```

```
    }

    public static void main(String[] args) {
        int[] vectorA = {1, 2, 3};
        int[] vectorB = {4, 5, 6};
        int[] result = crossProduct(vectorA, vectorB);
        System.out.println("Cross Product: [" + result[0] + ", " + result[1] + ", " + result[2] + "]");
    }
}
```

Expected Output:

Dot Product: 32

5. Check if Two Numbers are Equal

```
#include <stdio.h>

int main() {
    // Default values
    int num1 = 10;
    int num2 = 20;

    // Check if the numbers are equal
    if (num1 == num2) {
        printf("The numbers are equal.\n");
    } else {
        printf("The numbers are not equal.\n");
    }

    return 0;
}
```

Expected Output:

The numbers are not equal.

6. count the Armstrong numbers in the interval 1 to 100

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

// Function to check if a number is an Armstrong number
int isArmstrong(int num) {
    int originalNum = num;
    int sum = 0;
    int digit;

    // For 1-digit and 2-digit numbers, sum the powers of the digits
    while (num != 0) {
        digit = num % 10;    // Extract the last digit
        sum += digit * digit; // Square the digit and add to sum
        num /= 10;           // Remove the last digit
    }

    // Return true if the sum equals the original number
    return (sum == originalNum);
}

int main() {
    int count = 0;

    // Loop through the range 1 to 100
    for (int i = 1; i < 100; i++) {
        if (isArmstrong(i)) {
            count++;
            printf("%d is an Armstrong number.\n", i);
        }
    }

    // Display the count of Armstrong numbers
```

```
    printf("\nTotal Armstrong numbers between 1 and 100: %d\n", count);
    return 0;
}
```

Expected Output:

1 is an Armstrong number.

Total Armstrong numbers between 1 and 100: 1