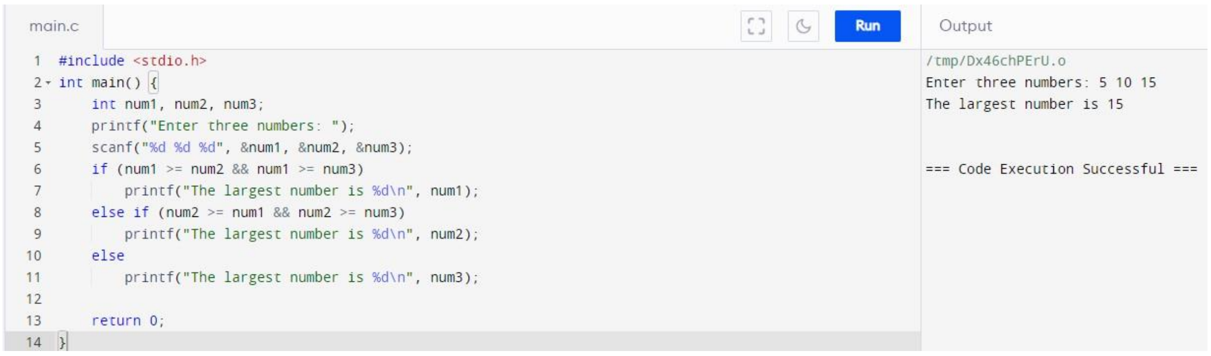
24CSEN2371 - Advanced Coding

C Programming Practice Lab1

R. Jitesh

HU21CSEN0100563

1. Find the largest number among the three numbers.



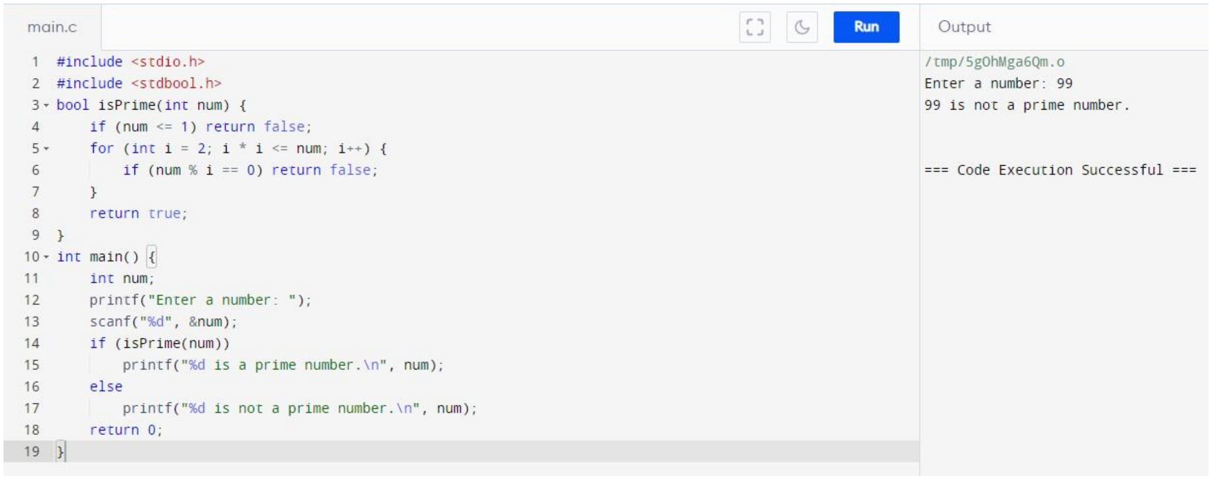
```c
#include <stdio.h>
int main() {
    int num1, num2, num3;
    printf("Enter three numbers: ");
    scanf("%d %d %d", &num1, &num2, &num3);
    if (num1 >= num2 && num1 >= num3)
        printf("The largest number is %d\n", num1);
    else if (num2 >= num1 && num2 >= num3)
        printf("The largest number is %d\n", num2);
    else
        printf("The largest number is %d\n", num3);

    return 0;
}
```

```
/tmp/Dx46chPErU.o
Enter three numbers: 5 10 15
The largest number is 15


=== Code Execution Successful ===
```

2. Write a Program to check whether a number is prime or not.

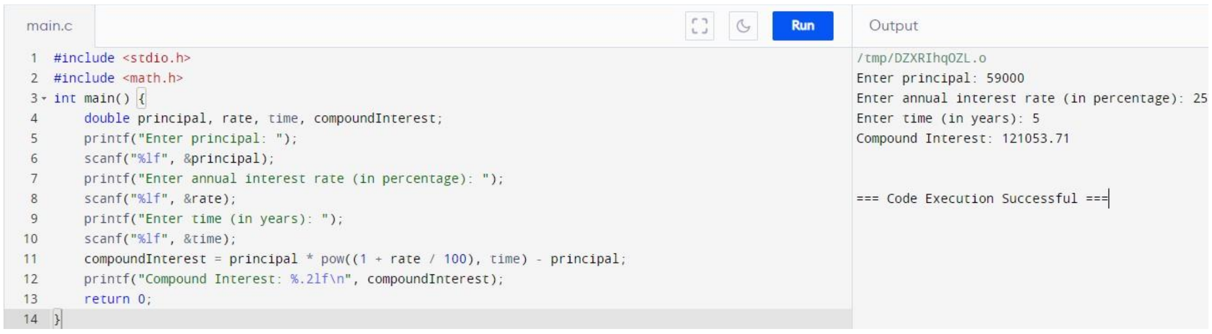


```c
#include <stdio.h>
#include <stdbool.h>
bool isPrime(int num) {
    if (num <= 1) return false;
    for (int i = 2; i * i <= num; i++) {
        if (num % i == 0) return false;
    }
    return true;
}
int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);
    if (isPrime(num))
        printf("%d is a prime number.\n", num);
    else
        printf("%d is not a prime number.\n", num);
    return 0;
}
```

```
/tmp/5gOhMga6Qm.o
Enter a number: 99
99 is not a prime number.


=== Code Execution Successful ===
```

3. Write a C program to calculate Compound Interest.

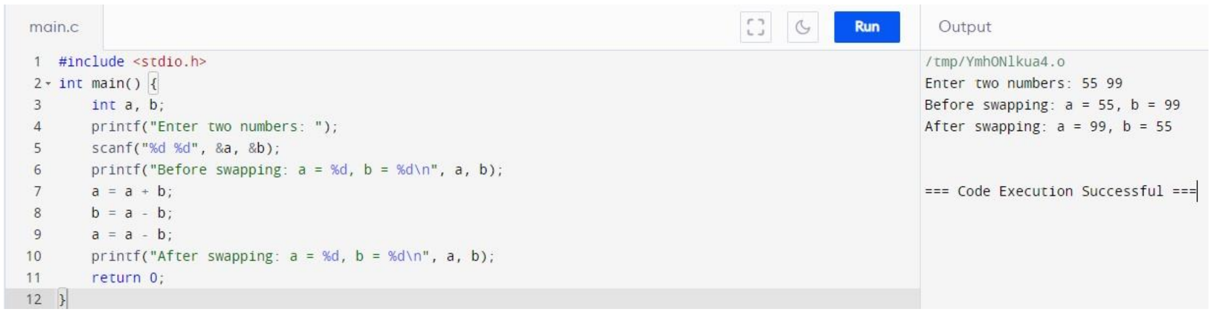


```c
#include <stdio.h>
#include <math.h>
int main() {
    double principal, rate, time, compoundInterest;
    printf("Enter principal: ");
    scanf("%lf", &principal);
    printf("Enter annual interest rate (in percentage): ");
    scanf("%lf", &rate);
    printf("Enter time (in years): ");
    scanf("%lf", &time);
    compoundInterest = principal * pow((1 + rate / 100), time) - principal;
    printf("Compound Interest: %.2lf\n", compoundInterest);
    return 0;
}
```

```
/tmp/DZXRIhqOZL.o
Enter principal: 59000
Enter annual interest rate (in percentage): 25
Enter time (in years): 5
Compound Interest: 121053.71


=== Code Execution Successful ===
```

4. Write a Program in C to Swap the values of two variables without using any extra variable.

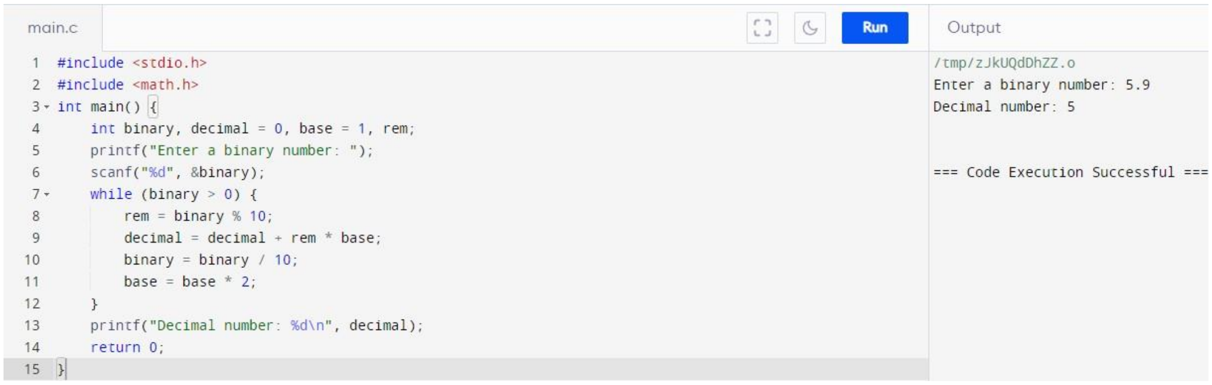


```c
#include <stdio.h>
int main() {
    int a, b;
    printf("Enter two numbers: ");
    scanf("%d %d", &a, &b);
    printf("Before swapping: a = %d, b = %d\n", a, b);
    a = a + b;
    b = a - b;
    a = a - b;
    printf("After swapping: a = %d, b = %d\n", a, b);
    return 0;
}
```

```
/tmp/YmhONlkua4.o
Enter two numbers: 55 99
Before swapping: a = 55, b = 99
After swapping: a = 99, b = 55


=== Code Execution Successful ===
```

5. Write a Program to convert the binary number into a decimal number.

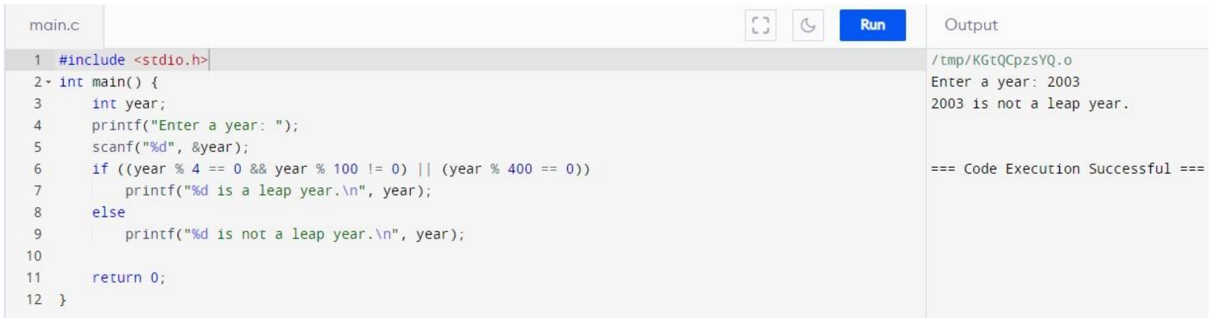


```c
#include <stdio.h>
#include <math.h>
int main() {
    int binary, decimal = 0, base = 1, rem;
    printf("Enter a binary number: ");
    scanf("%d", &binary);
    while (binary > 0) {
        rem = binary % 10;
        decimal = decimal + rem * base;
        binary = binary / 10;
        base = base * 2;
    }
    printf("Decimal number: %d\n", decimal);
    return 0;
}
```

```
/tmp/zJkUQdDhZZ.o
Enter a binary number: 5.9
Decimal number: 5


=== Code Execution Successful ===
```

6. Write a Program to check if the year is a leap year or not.



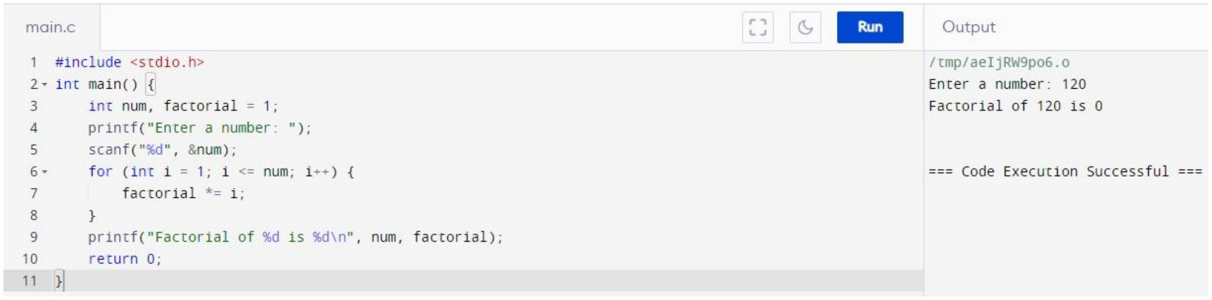
```c
#include <stdio.h>
int main() {
    int year;
    printf("Enter a year: ");
    scanf("%d", &year);
    if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0))
        printf("%d is a leap year.\n", year);
    else
        printf("%d is not a leap year.\n", year);

    return 0;
}
```

```
/tmp/KGtQCpzsYQ.o
Enter a year: 2003
2003 is not a leap year.


=== Code Execution Successful ===
```

7. Write a program to find the Factorial of a Number without recursion.

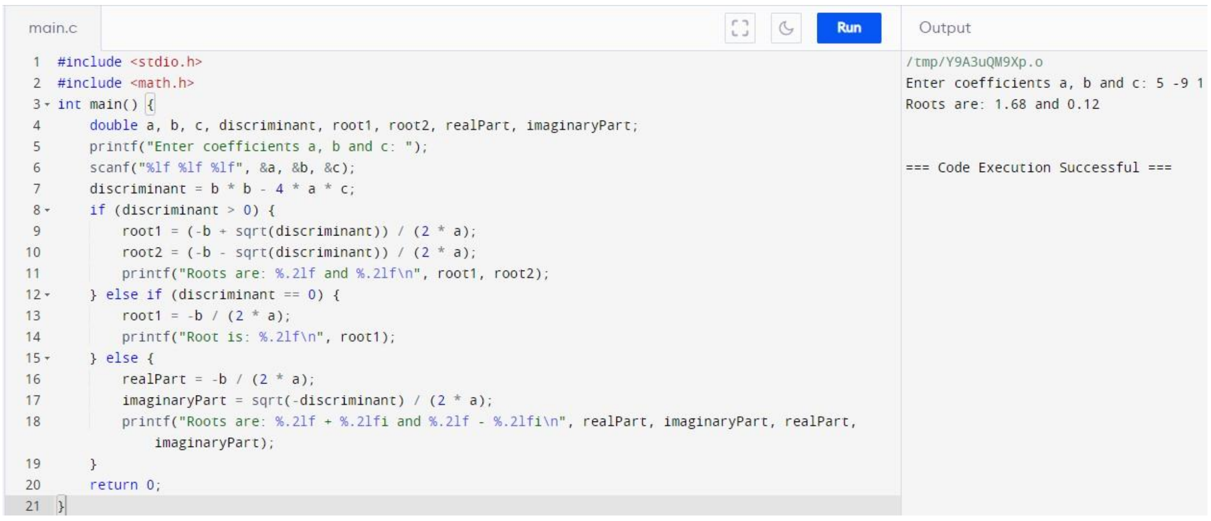


```c
#include <stdio.h>
int main() {
    int num, factorial = 1;
    printf("Enter a number: ");
    scanf("%d", &num);
    for (int i = 1; i <= num; i++) {
        factorial *= i;
    }
    printf("Factorial of %d is %d\n", num, factorial);
    return 0;
}
```

```
/tmp/aeIjRW9po6.o
Enter a number: 120
Factorial of 120 is 0


=== Code Execution Successful ===
```

8. Write a program to Find all the roots of a quadratic equation in C.

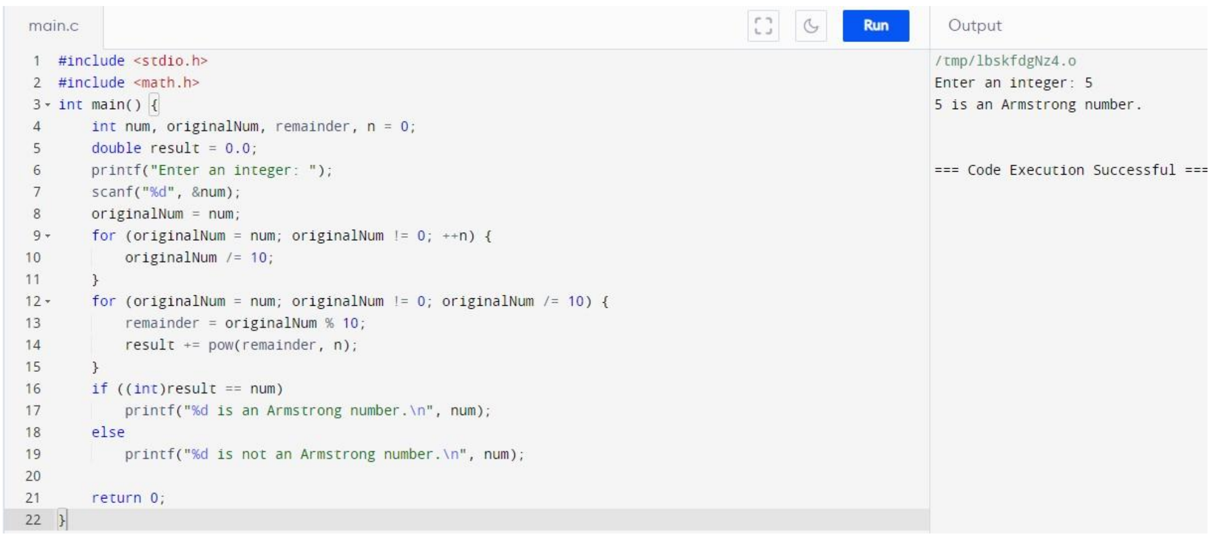


```c
#include <stdio.h>
#include <math.h>
int main() {
    double a, b, c, discriminant, root1, root2, realPart, imaginaryPart;
    printf("Enter coefficients a, b and c: ");
    scanf("%lf %lf %lf", &a, &b, &c);
    discriminant = b * b - 4 * a * c;
    if (discriminant > 0) {
        root1 = (-b + sqrt(discriminant)) / (2 * a);
        root2 = (-b - sqrt(discriminant)) / (2 * a);
        printf("Roots are: %.2lf and %.2lf\n", root1, root2);
    } else if (discriminant == 0) {
        root1 = -b / (2 * a);
        printf("Root is: %.2lf\n", root1);
    } else {
        realPart = -b / (2 * a);
        imaginaryPart = sqrt(-discriminant) / (2 * a);
        printf("Roots are: %.2lf + %.2lfi and %.2lf - %.2lfi\n", realPart, imaginaryPart, realPart,
            imaginaryPart);
    }
    return 0;
}
```

```
/tmp/Y9A3uQM9Xp.o
Enter coefficients a, b and c: 5 -9 1
Roots are: 1.68 and 0.12

=== Code Execution Successful ===
```

9. a Program to Check if a number is an Armstrong number or not.



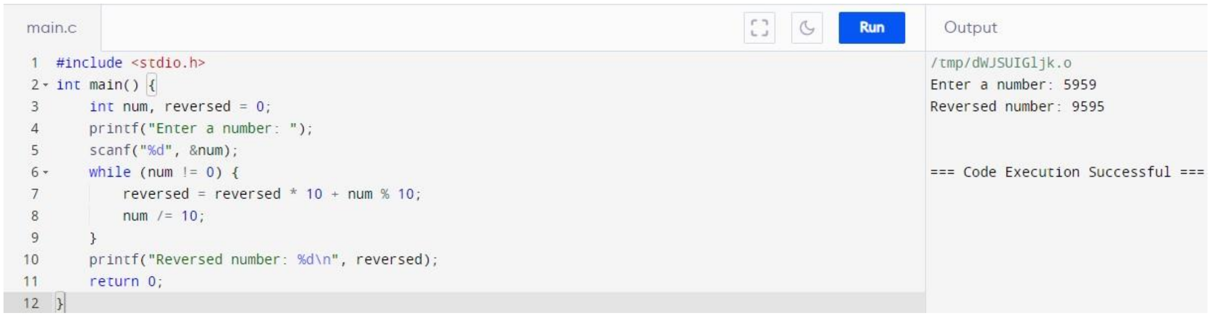
```c
#include <stdio.h>
#include <math.h>
int main() {
    int num, originalNum, remainder, n = 0;
    double result = 0.0;
    printf("Enter an integer: ");
    scanf("%d", &num);
    originalNum = num;
    for (originalNum = num; originalNum != 0; ++n) {
        originalNum /= 10;
    }
    for (originalNum = num; originalNum != 0; originalNum /= 10) {
        remainder = originalNum % 10;
        result += pow(remainder, n);
    }
    if ((int)result == num)
        printf("%d is an Armstrong number.\n", num);
    else
        printf("%d is not an Armstrong number.\n", num);

    return 0;
}
```

```
/tmp/lbskfdgNz4.o
Enter an integer: 5
5 is an Armstrong number.


=== Code Execution Successful ===
```

10. Write a Program to reverse a number.

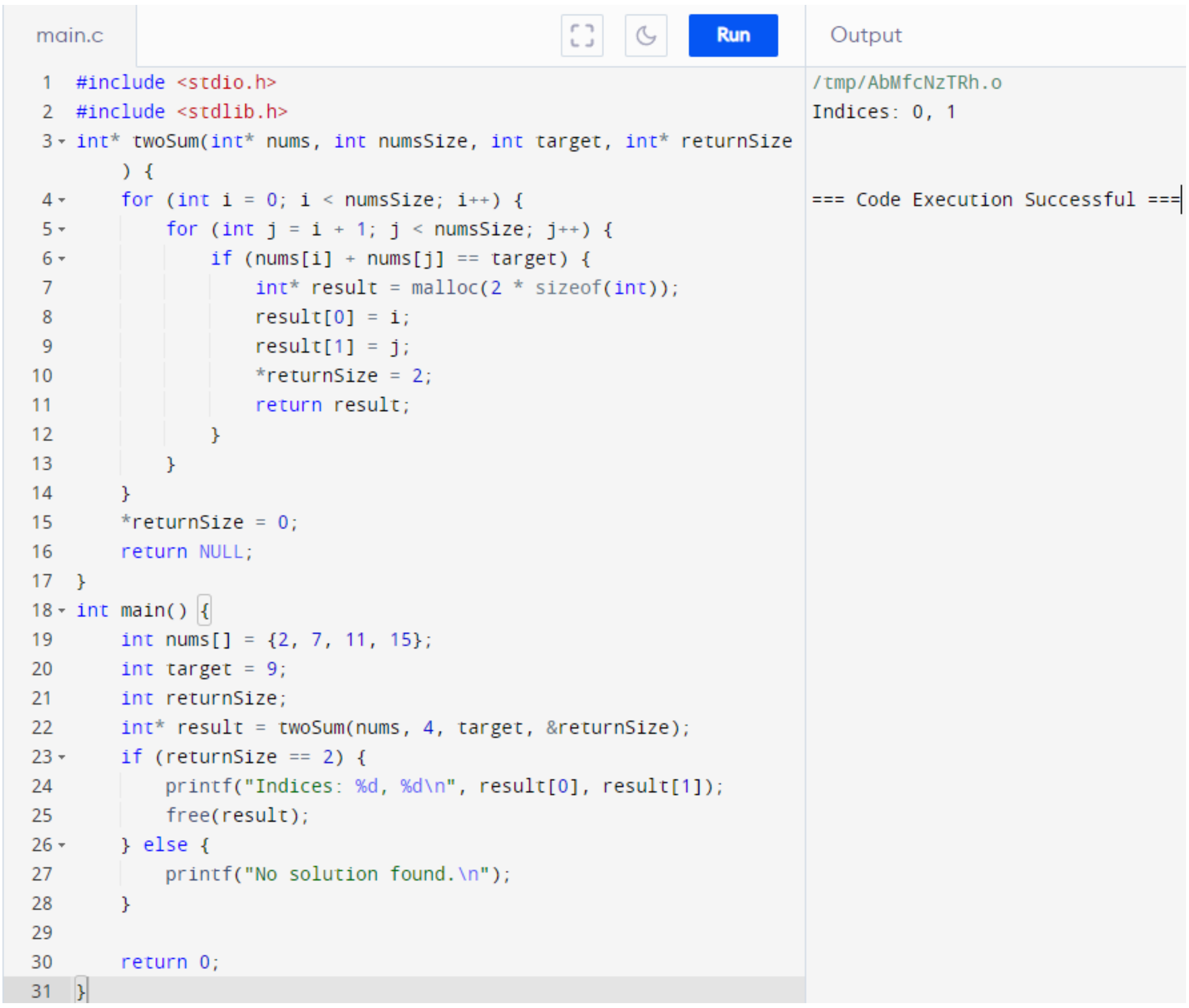


```c
#include <stdio.h>
int main() {
    int num, reversed = 0;
    printf("Enter a number: ");
    scanf("%d", &num);
    while (num != 0) {
        reversed = reversed * 10 + num % 10;
        num /= 10;
    }
    printf("Reversed number: %d\n", reversed);
    return 0;
}
```

```
/tmp/dWJSUIGljk.o
Enter a number: 5959
Reversed number: 9595


=== Code Execution Successful ===
```

LeetCode: Given an array of integers nums and an integer target, return indices of the two numbers such that they add up to target.



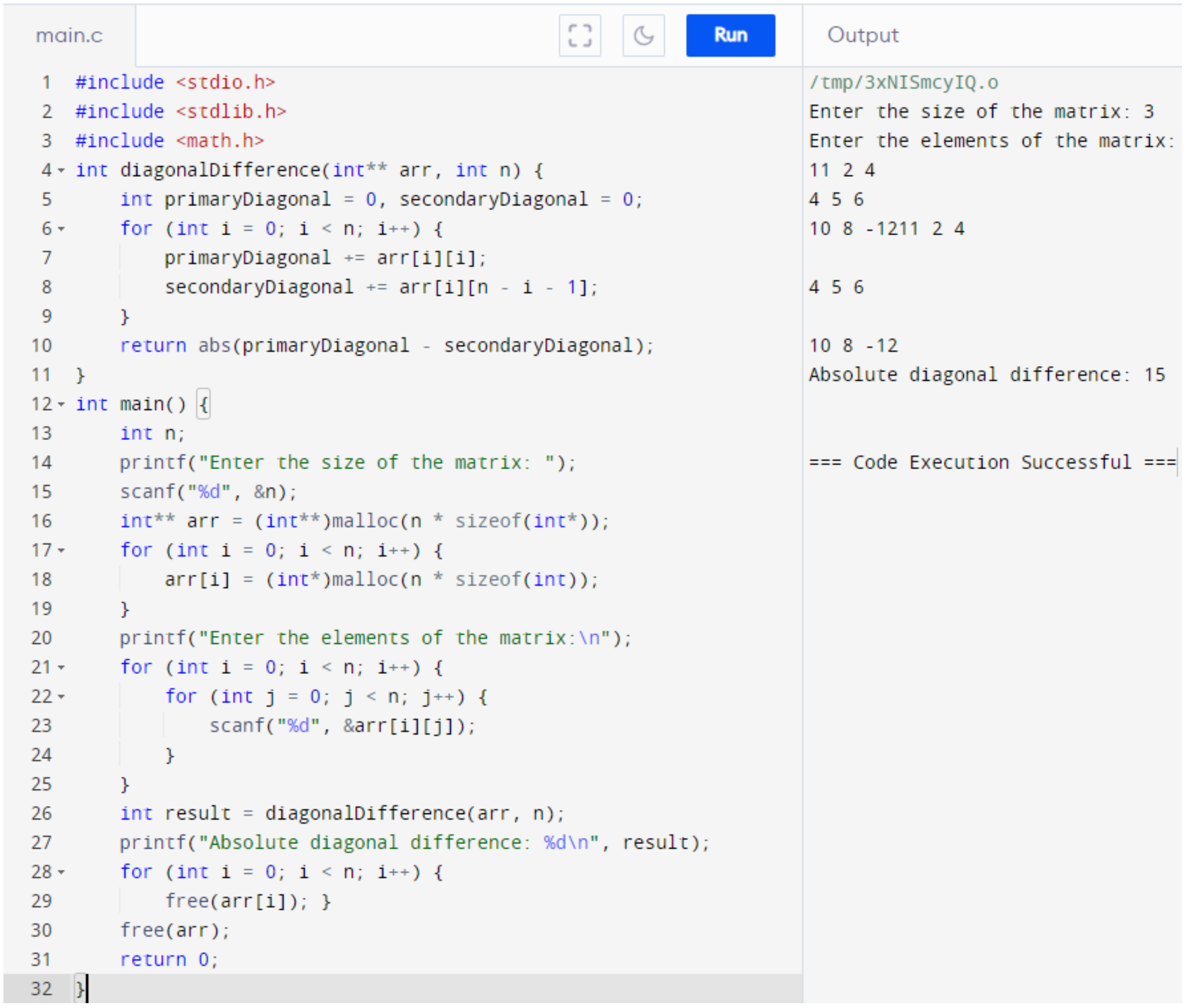
```c
#include <stdio.h>
#include <stdlib.h>
int* twoSum(int* nums, int numsSize, int target, int* returnSize
    ) {
    for (int i = 0; i < numsSize; i++) {
        for (int j = i + 1; j < numsSize; j++) {
            if (nums[i] + nums[j] == target) {
                int* result = malloc(2 * sizeof(int));
                result[0] = i;
                result[1] = j;
                *returnSize = 2;
                return result;
            }
        }
    }
    *returnSize = 0;
    return NULL;
}
int main() {
    int nums[] = {2, 7, 11, 15};
    int target = 9;
    int returnSize;
    int* result = twoSum(nums, 4, target, &returnSize);
    if (returnSize == 2) {
        printf("Indices: %d, %d\n", result[0], result[1]);
        free(result);
    } else {
        printf("No solution found.\n");
    }

    return 0;
}
```

```
/tmp/AbMfcNzTRh.o
Indices: 0, 1


=== Code Execution Successful ===
```

HackerRank: Diagonal Difference: Calculate the absolute difference between the sums of the diagonals in a square matrix.



```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int diagonalDifference(int** arr, int n) {
    int primaryDiagonal = 0, secondaryDiagonal = 0;
    for (int i = 0; i < n; i++) {
        primaryDiagonal += arr[i][i];
        secondaryDiagonal += arr[i][n - i - 1];
    }
    return abs(primaryDiagonal - secondaryDiagonal);
}
int main() {
    int n;
    printf("Enter the size of the matrix: ");
    scanf("%d", &n);
    int** arr = (int**)malloc(n * sizeof(int*));
    for (int i = 0; i < n; i++) {
        arr[i] = (int*)malloc(n * sizeof(int));
    }
    printf("Enter the elements of the matrix:\n");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            scanf("%d", &arr[i][j]);
        }
    }
    int result = diagonalDifference(arr, n);
    printf("Absolute diagonal difference: %d\n", result);
    for (int i = 0; i < n; i++) {
        free(arr[i]); }
    free(arr);
    return 0;
}
```

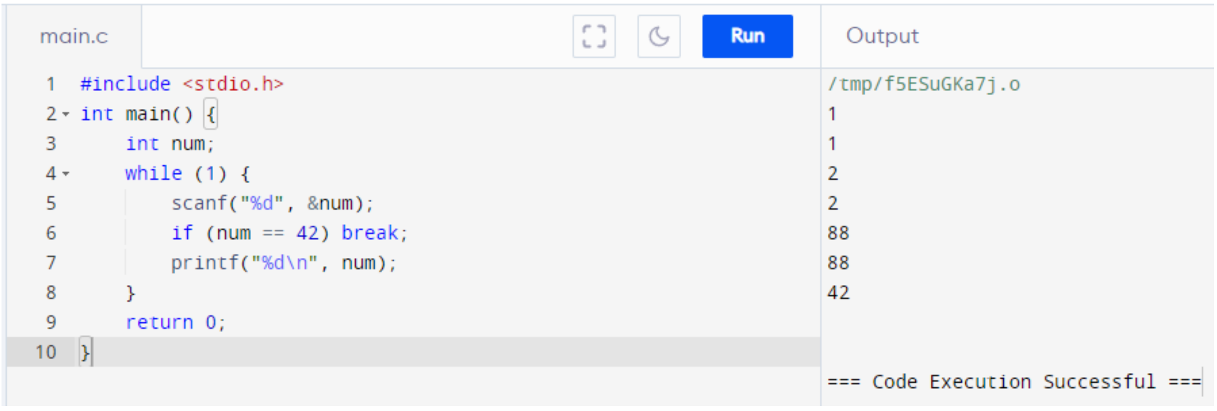
```
/tmp/3xNISmcyIQ.o
Enter the size of the matrix: 3
Enter the elements of the matrix:
11 2 4
4 5 6
10 8 -1211 2 4

4 5 6

10 8 -12
Absolute diagonal difference: 15


=== Code Execution Successful ===
```

CodeChef: Life, the Universe, and Everything: Write a program that reads numbers from input and stops processing input after reading the number 42.

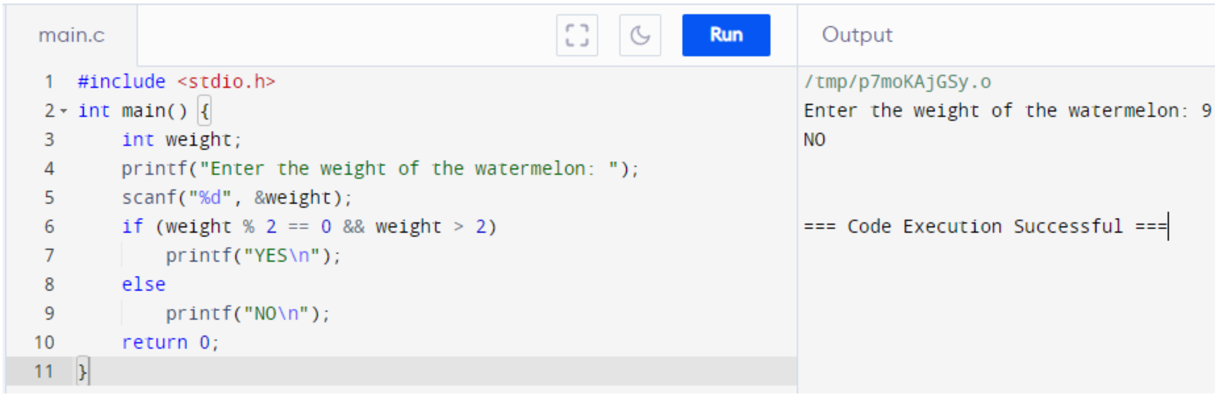


```c
#include <stdio.h>
int main() {
    int num;
    while (1) {
        scanf("%d", &num);
        if (num == 42) break;
        printf("%d\n", num);
    }
    return 0;
}
```

```
/tmp/f5ESuGKa7j.o
1
1
2
2
88
88
42


=== Code Execution Successful ===
```

Codeforces: Watermelon: Determine if a watermelon can be split into two parts, each of which weighs an even number of kilos.

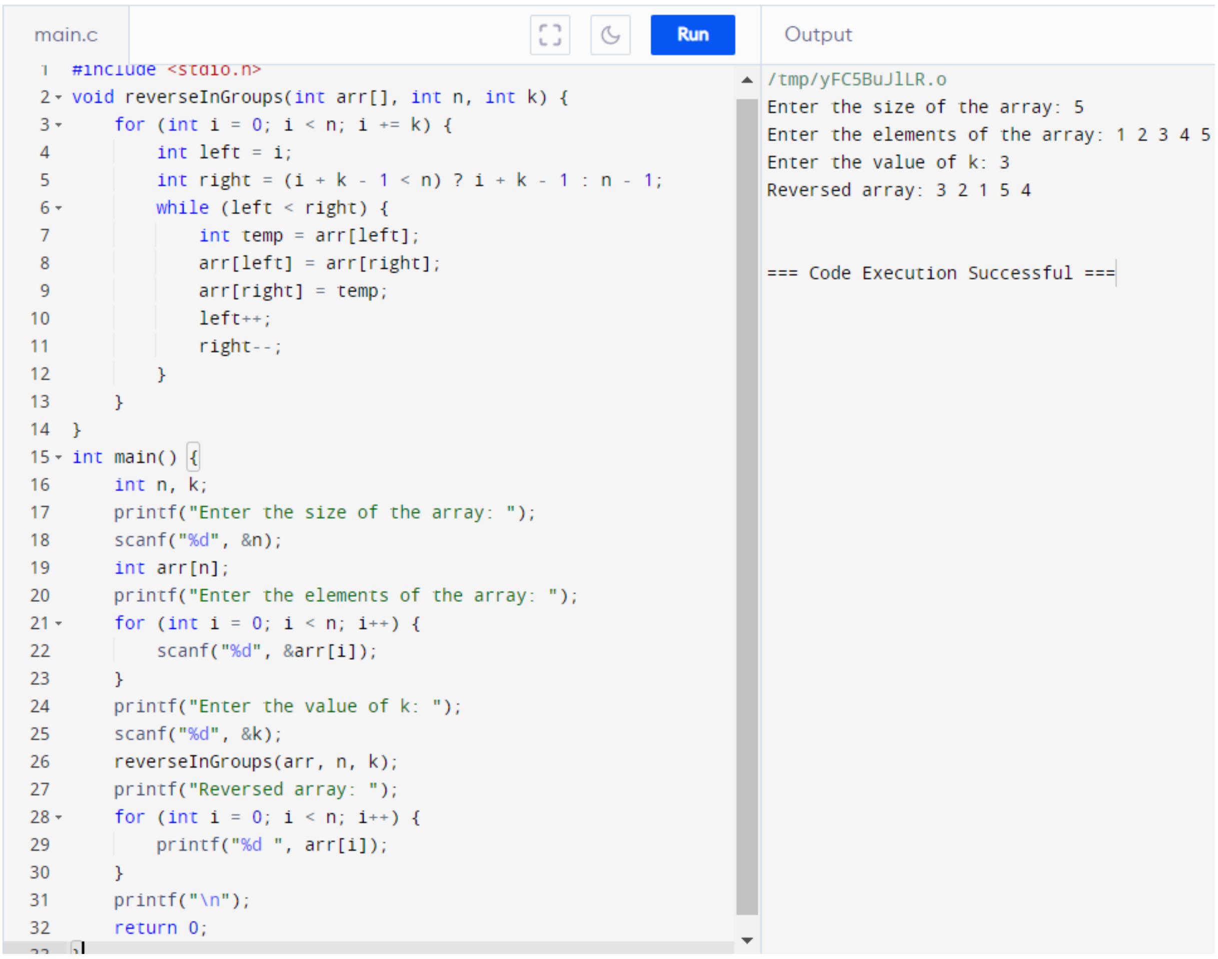


```c
#include <stdio.h>
int main() {
    int weight;
    printf("Enter the weight of the watermelon: ");
    scanf("%d", &weight);
    if (weight % 2 == 0 && weight > 2)
        printf("YES\n");
    else
        printf("NO\n");
    return 0;
}
```

```
/tmp/p7moKAjGSy.o
Enter the weight of the watermelon: 9
NO


=== Code Execution Successful ===
```

GeeksforGeeks: Reverse Array in Groups: Given an array, reverse every sub-array formed by consecutive k elements.

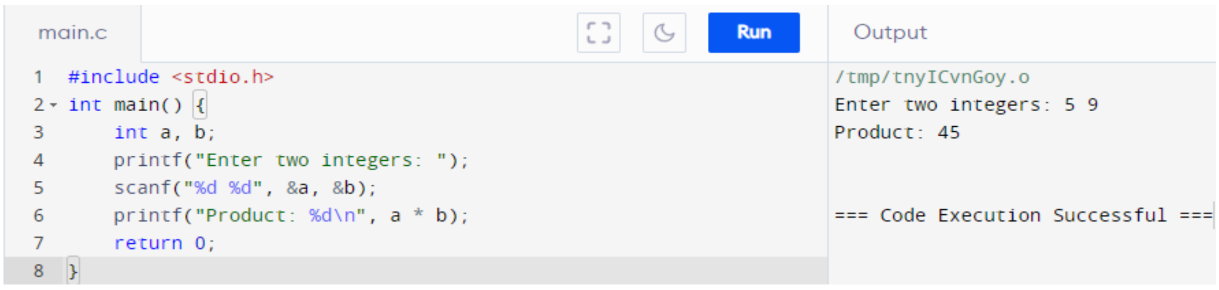


```c
#include <stdio.h>
void reverseInGroups(int arr[], int n, int k) {
    for (int i = 0; i < n; i += k) {
        int left = i;
        int right = (i + k - 1 < n) ? i + k - 1 : n - 1;
        while (left < right) {
            int temp = arr[left];
            arr[left] = arr[right];
            arr[right] = temp;
            left++;
            right--;
        }
    }
}
int main() {
    int n, k;
    printf("Enter the size of the array: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter the elements of the array: ");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    printf("Enter the value of k: ");
    scanf("%d", &k);
    reverseInGroups(arr, n, k);
    printf("Reversed array: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
    return 0;
}
```

```
/tmp/yFC5BuJ1LR.o
Enter the size of the array: 5
Enter the elements of the array: 1 2 3 4 5
Enter the value of k: 3
Reversed array: 3 2 1 5 4


=== Code Execution Successful ===
```

AtCoder: Product: Find the product of two integers.

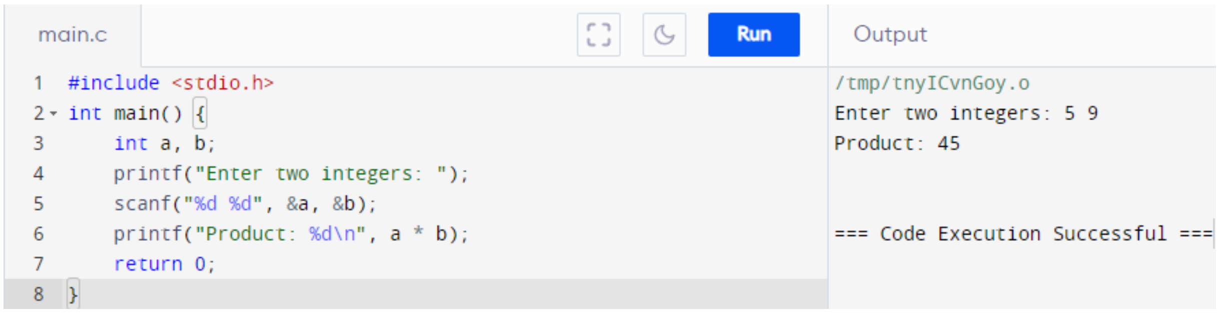


```c
#include <stdio.h>
int main() {
    int a, b;
    printf("Enter two integers: ");
    scanf("%d %d", &a, &b);
    printf("Product: %d\n", a * b);
    return 0;
}
```

```
/tmp/tnyICvnGoy.o
Enter two integers: 5 9
Product: 45


=== Code Execution Successful ===
```

Exercism: Hamming: Calculate the Hamming Distance between two DNA strands.

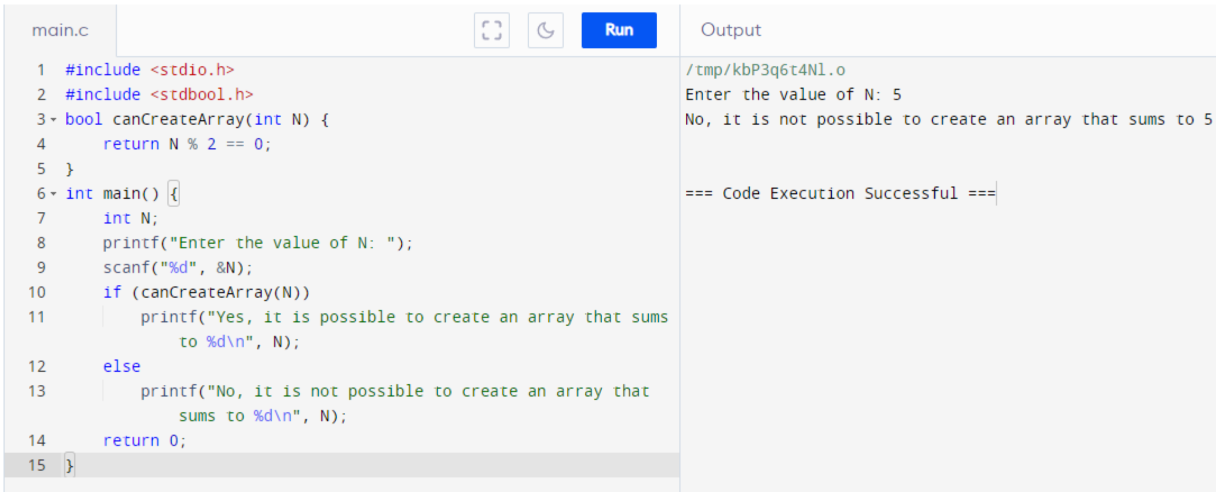


```c
#include <stdio.h>
int main() {
    int a, b;
    printf("Enter two integers: ");
    scanf("%d %d", &a, &b);
    printf("Product: %d\n", a * b);
    return 0;
}
```

```
/tmp/tnyICvnGoy.o
Enter two integers: 5 9
Product: 45


=== Code Execution Successful ===
```

TopCoder: SRM 758 Div 2 - Very Easy Problem: Given an integer N, determine if it is possible to create an array of integers that sums to N.

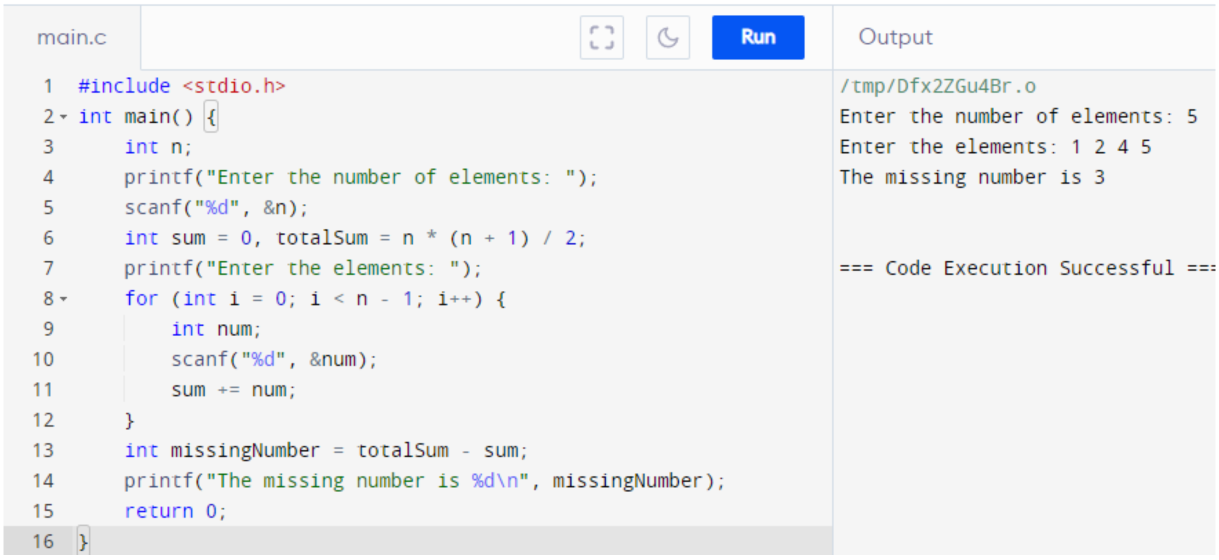


```c
#include <stdio.h>
#include <stdbool.h>
bool canCreateArray(int N) {
    return N % 2 == 0;
}
int main() {
    int N;
    printf("Enter the value of N: ");
    scanf("%d", &N);
    if (canCreateArray(N))
        printf("Yes, it is possible to create an array that sums
            to %d\n", N);
    else
        printf("No, it is not possible to create an array that
            sums to %d\n", N);
    return 0;
}
```

```
/tmp/kbP3q6t4Nl.o
Enter the value of N: 5
No, it is not possible to create an array that sums to 5


=== Code Execution Successful ===
```

CSES Problem Set: Missing Number: Find the missing number in a list of n integers where one number from 1 to n is missing.

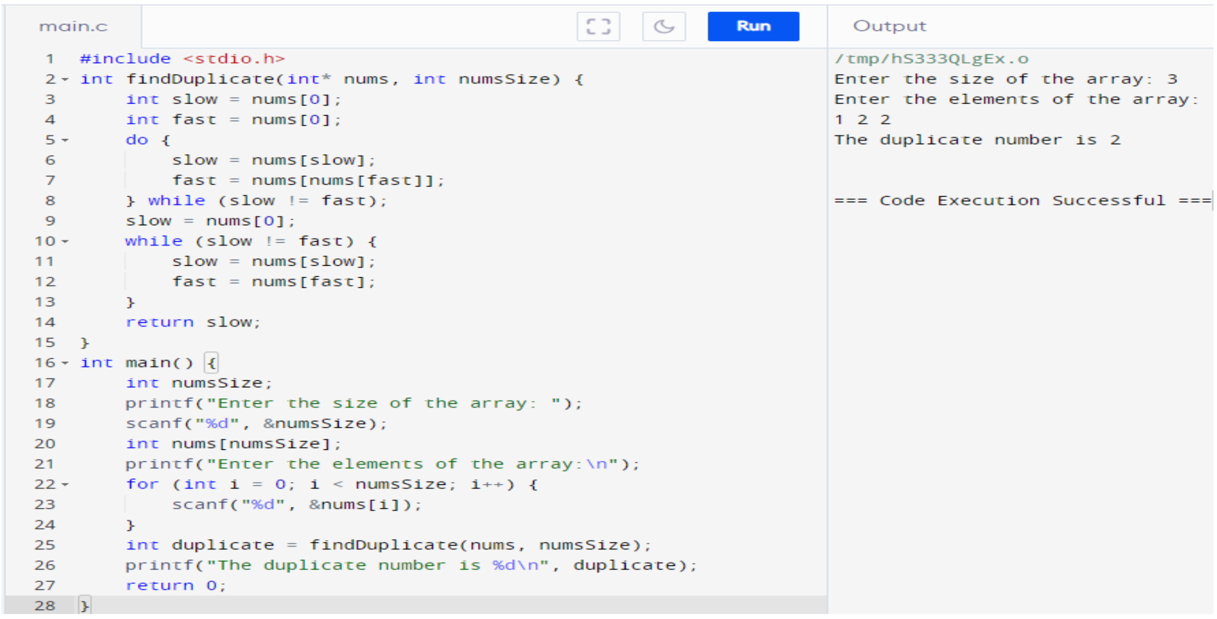


```c
#include <stdio.h>
int main() {
    int n;
    printf("Enter the number of elements: ");
    scanf("%d", &n);
    int sum = 0, totalSum = n * (n + 1) / 2;
    printf("Enter the elements: ");
    for (int i = 0; i < n - 1; i++) {
        int num;
        scanf("%d", &num);
        sum += num;
    }
    int missingNumber = totalSum - sum;
    printf("The missing number is %d\n", missingNumber);
    return 0;
}
```

```
/tmp/Dfx2ZGu4Br.o
Enter the number of elements: 5
Enter the elements: 1 2 4 5
The missing number is 3


=== Code Execution Successful ===
```

InterviewBit: Find Duplicate in Array: Given a read-only array of n+1 integers between 1 and n, find one duplicate number."



```c
#include <stdio.h>
int findDuplicate(int* nums, int numsSize) {
    int slow = nums[0];
    int fast = nums[0];
    do {
        slow = nums[slow];
        fast = nums[nums[fast]];
    } while (slow != fast);
    slow = nums[0];
    while (slow != fast) {
        slow = nums[slow];
        fast = nums[fast];
    }
    return slow;
}
int main() {
    int numsSize;
    printf("Enter the size of the array: ");
    scanf("%d", &numsSize);
    int nums[numsSize];
    printf("Enter the elements of the array:\n");
    for (int i = 0; i < numsSize; i++) {
        scanf("%d", &nums[i]);
    }
    int duplicate = findDuplicate(nums, numsSize);
    printf("The duplicate number is %d\n", duplicate);
    return 0;
}
```

```
/tmp/hS333QLgEx.o
Enter the size of the array: 3
Enter the elements of the array:
1 2 2
The duplicate number is 2


=== Code Execution Successful ===
```