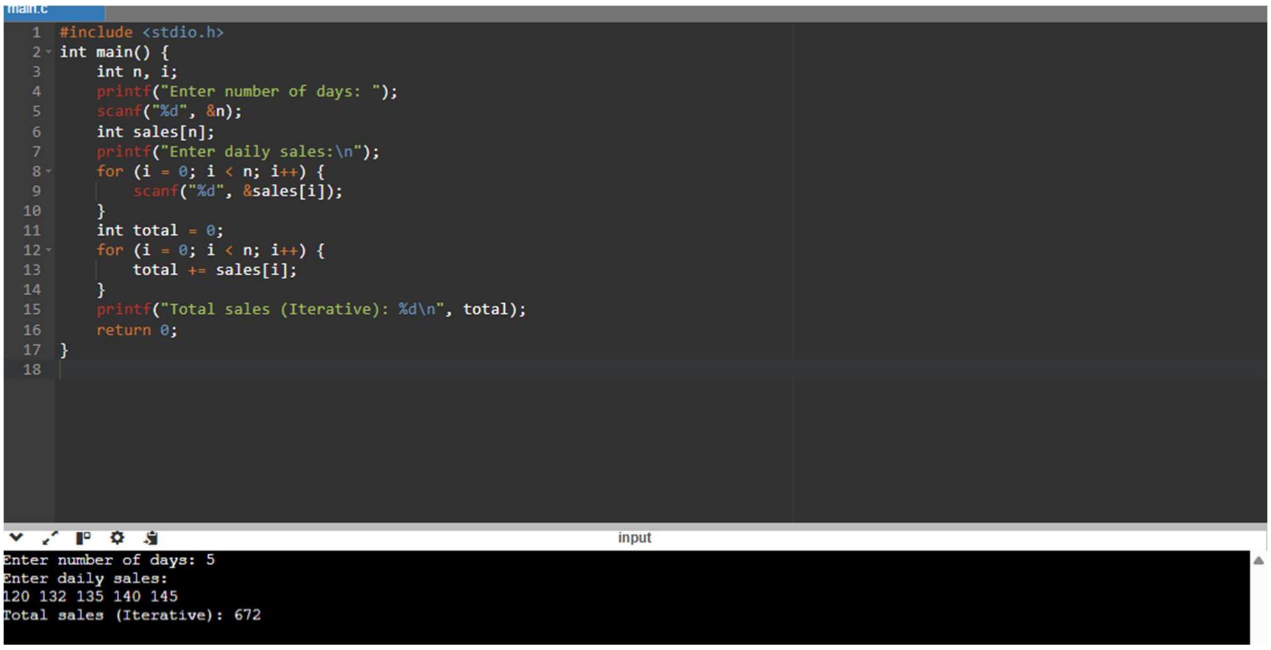
1. Sum of Elements in an Array

• Scenario: You have a list of daily sales of a store.

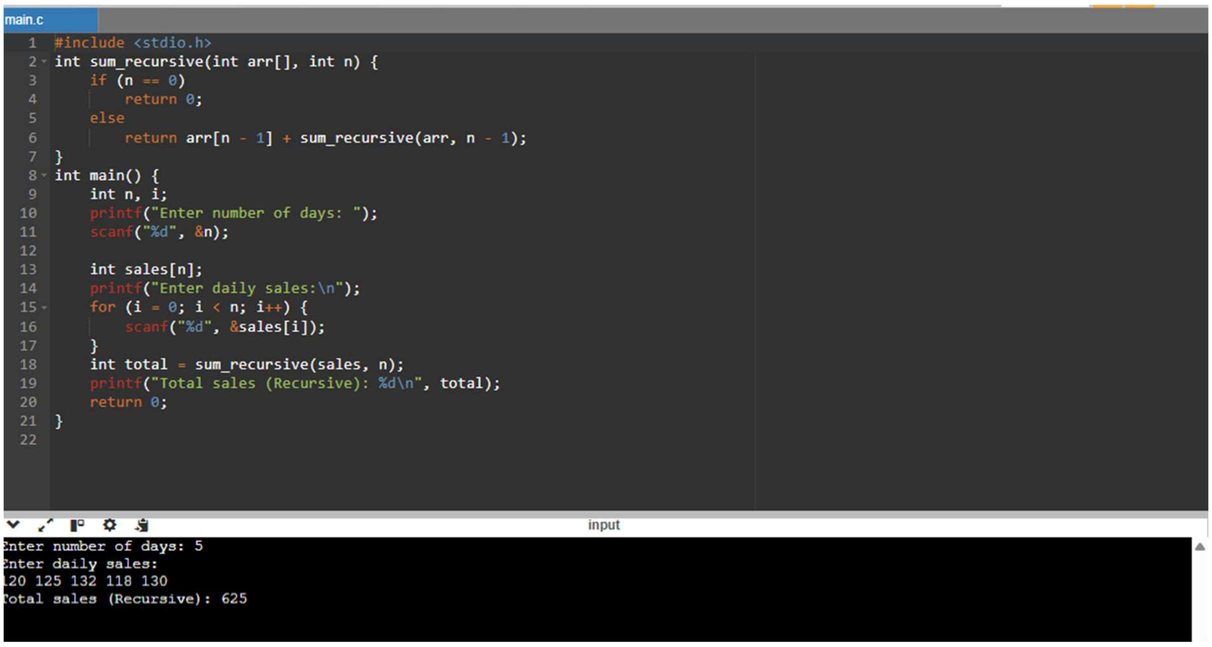
a) Write a simple iterative program to compute the sum of elements.

b) Write a recursive version of the same program.



```c
#include <stdio.h>
int main() {
    int n, i;
    printf("Enter number of days: ");
    scanf("%d", &n);
    int sales[n];
    printf("Enter daily sales:\n");
    for (i = 0; i < n; i++) {
        scanf("%d", &sales[i]);
    }
    int total = 0;
    for (i = 0; i < n; i++) {
        total += sales[i];
    }
    printf("Total sales (Iterative): %d\n", total);
    return 0;
}
```

```
Enter number of days: 5
Enter daily sales:
120 132 135 140 145
Total sales (Iterative): 672
```



```c
#include <stdio.h>
int sum_recursive(int arr[], int n) {
    if (n == 0)
        return 0;
    else
        return arr[n - 1] + sum_recursive(arr, n - 1);
}
int main() {
    int n, i;
    printf("Enter number of days: ");
    scanf("%d", &n);

    int sales[n];
    printf("Enter daily sales:\n");
    for (i = 0; i < n; i++) {
        scanf("%d", &sales[i]);
    }
    int total = sum_recursive(sales, n);
    printf("Total sales (Recursive): %d\n", total);
    return 0;
}
```

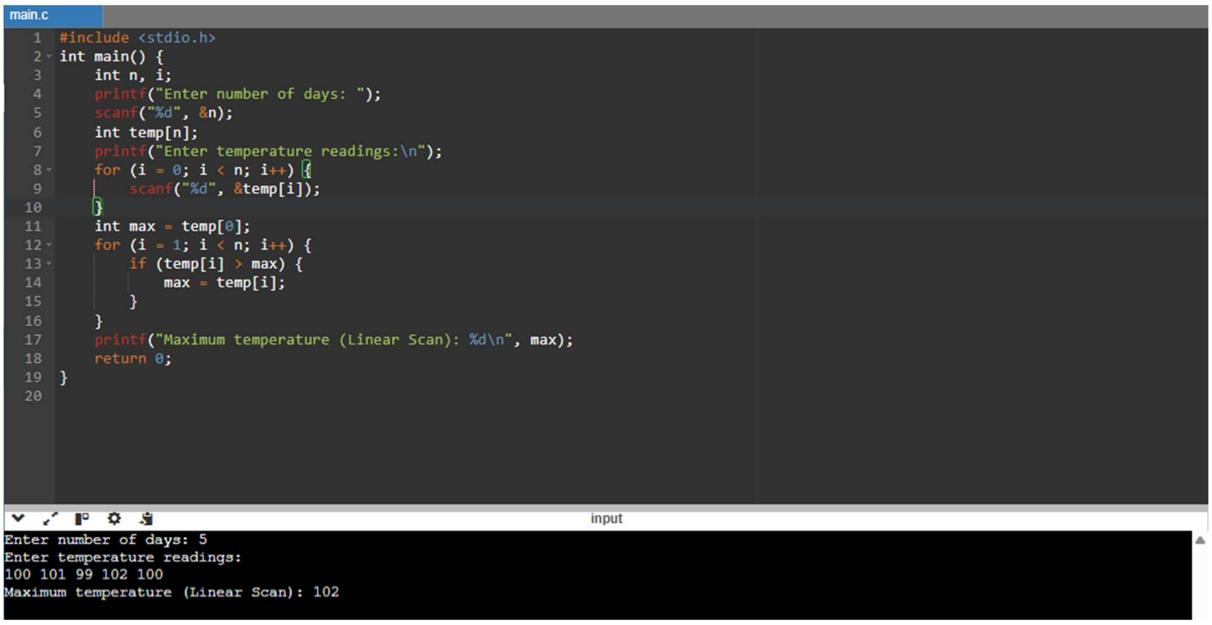
```
Enter number of days: 5
Enter daily sales:
120 125 132 118 130
Total sales (Recursive): 625
```

2. Find Maximum Element in an Array

• Scenario: Given temperature readings for a week.

a) Find maximum element using a simple linear scan (O(n)).

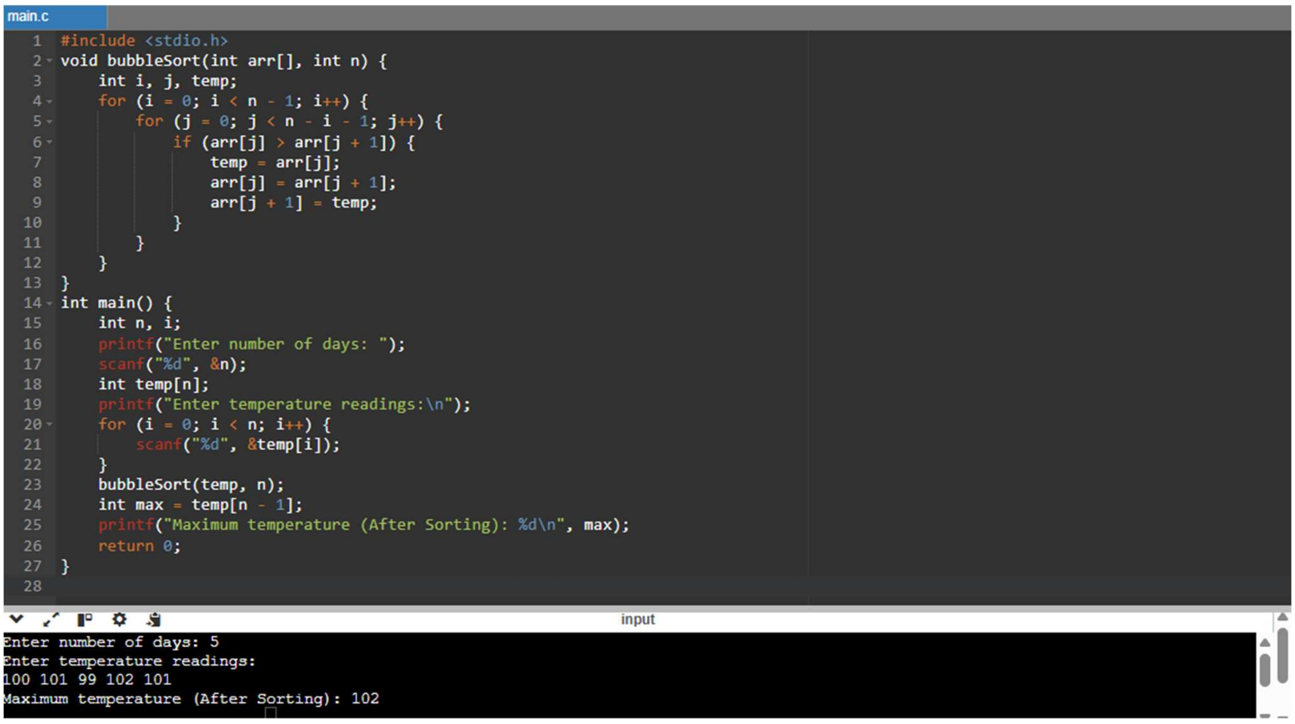
b) Can you find it by first sorting the array and taking the last element?



main.c

```c
#include <stdio.h>
int main() {
    int n, i;
    printf("Enter number of days: ");
    scanf("%d", &n);
    int temp[n];
    printf("Enter temperature readings:\n");
    for (i = 0; i < n; i++) {
        scanf("%d", &temp[i]);
    }
    int max = temp[0];
    for (i = 1; i < n; i++) {
        if (temp[i] > max) {
            max = temp[i];
        }
    }
    printf("Maximum temperature (Linear Scan): %d\n", max);
    return 0;
}
```

```
Enter number of days: 5
Enter temperature readings:
100 101 99 102 100
Maximum temperature (Linear Scan): 102
```



main.c

```c
#include <stdio.h>
void bubbleSort(int arr[], int n) {
    int i, j, temp;
    for (i = 0; i < n - 1; i++) {
        for (j = 0; j < n - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}
int main() {
    int n, i;
    printf("Enter number of days: ");
    scanf("%d", &n);
    int temp[n];
    printf("Enter temperature readings:\n");
    for (i = 0; i < n; i++) {
        scanf("%d", &temp[i]);
    }
    bubbleSort(temp, n);
    int max = temp[n - 1];
    printf("Maximum temperature (After Sorting): %d\n", max);
    return 0;
}
```

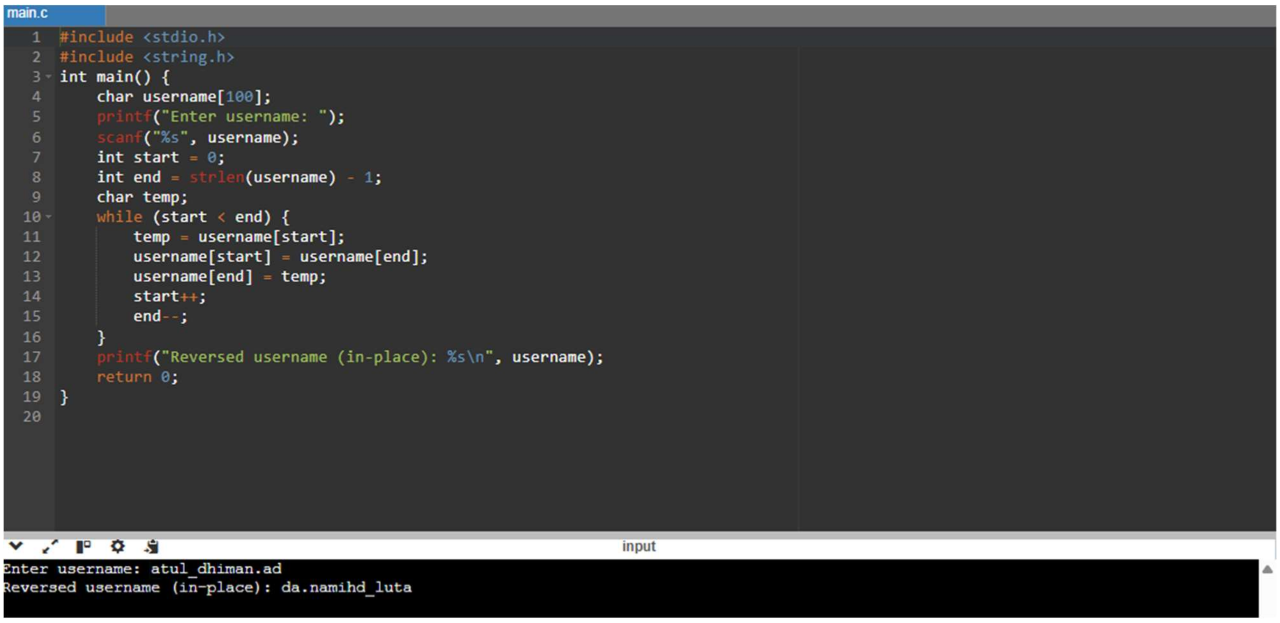
```
Enter number of days: 5
Enter temperature readings:
100 101 99 102 101
Maximum temperature (After Sorting): 102
```

3. Reverse a String

• Scenario: Reversing a username for some encryption purpose.

a) Reverse in-place using two-pointer technique (O(n), O(1)).

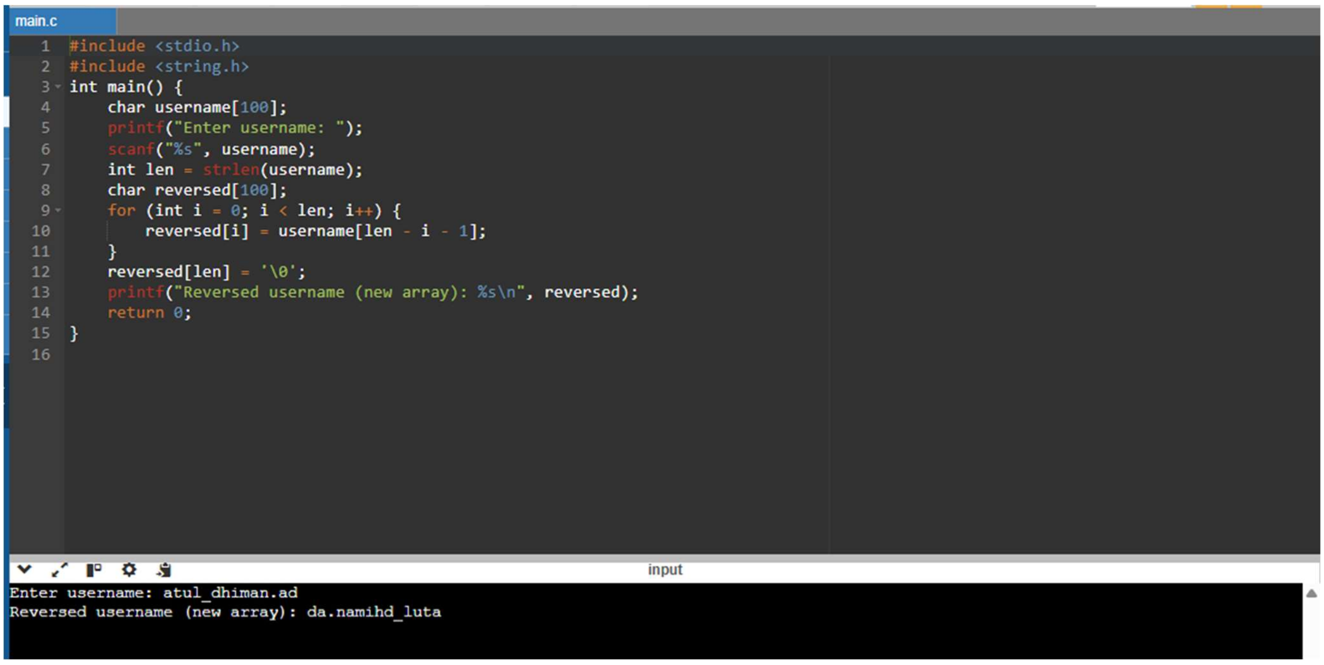
b) Reverse by creating a new array (O(n), O(n)).



main.c

```c
#include <stdio.h>
#include <string.h>
int main() {
    char username[100];
    printf("Enter username: ");
    scanf("%s", username);
    int start = 0;
    int end = strlen(username) - 1;
    char temp;
    while (start < end) {
        temp = username[start];
        username[start] = username[end];
        username[end] = temp;
        start++;
        end--;
    }
    printf("Reversed username (in-place): %s\n", username);
    return 0;
}
```

```
Enter username: atul_dhiman.ad
Reversed username (in-place): da.namihd_luta
```



main.c

```c
#include <stdio.h>
#include <string.h>
int main() {
    char username[100];
    printf("Enter username: ");
    scanf("%s", username);
    int len = strlen(username);
    char reversed[100];
    for (int i = 0; i < len; i++) {
        reversed[i] = username[len - i - 1];
    }
    reversed[len] = '\0';
    printf("Reversed username (new array): %s\n", reversed);
    return 0;
}
```

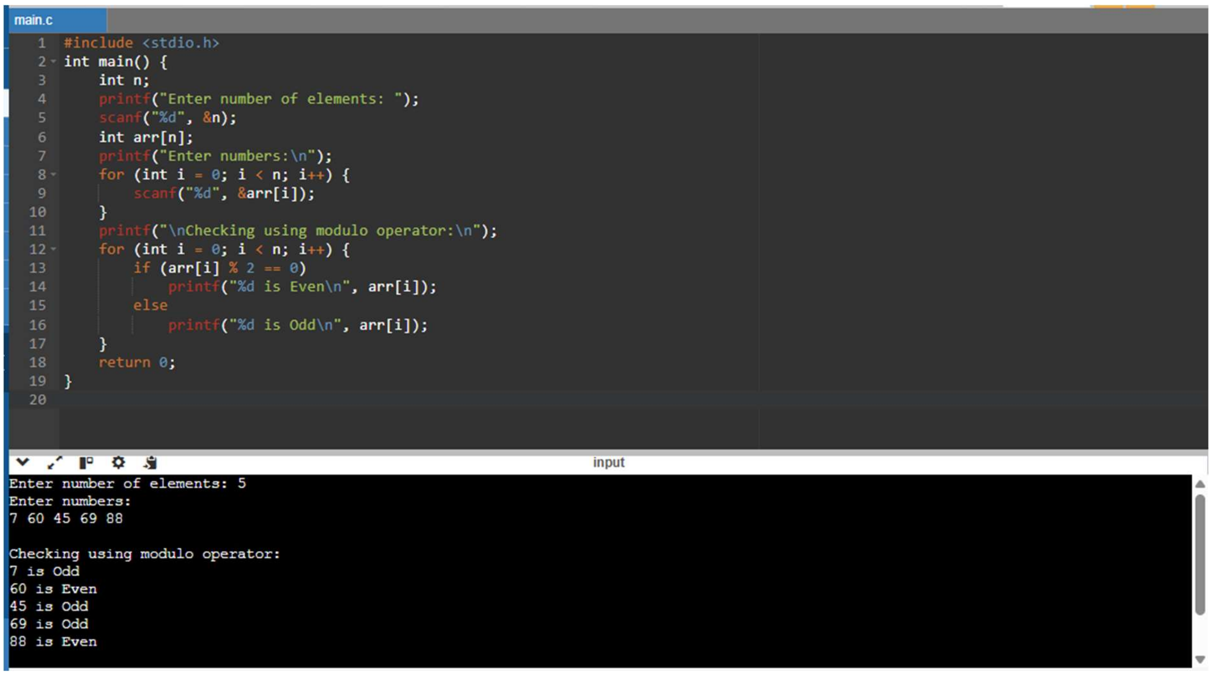
```
Enter username: atul_dhiman.ad
Reversed username (new array): da.namihd_luta
```

4. Check Even or Odd

• Scenario: Given a list of numbers, print whether each number is even or odd.

a) Using modulo operator (O(1)).

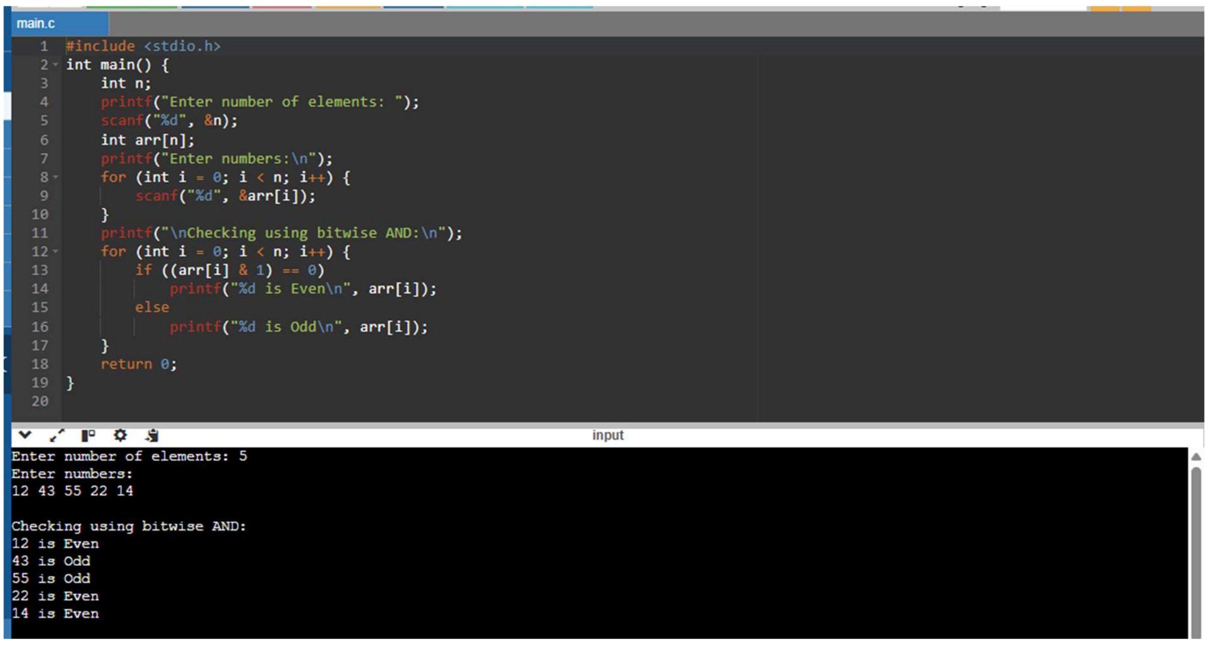
b) Using bitwise AND (n & 1) (O (1)).



main.c

```c
#include <stdio.h>
int main() {
    int n;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter numbers:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    printf("\nChecking using modulo operator:\n");
    for (int i = 0; i < n; i++) {
        if (arr[i] % 2 == 0)
            printf("%d is Even\n", arr[i]);
        else
            printf("%d is Odd\n", arr[i]);
    }
    return 0;
}
```

```
Enter number of elements: 5
Enter numbers:
7 60 45 69 88

Checking using modulo operator:
7 is Odd
60 is Even
45 is Odd
69 is Odd
88 is Even
```



main.c

```c
#include <stdio.h>
int main() {
    int n;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter numbers:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    printf("\nChecking using bitwise AND:\n");
    for (int i = 0; i < n; i++) {
        if ((arr[i] & 1) == 0)
            printf("%d is Even\n", arr[i]);
        else
            printf("%d is Odd\n", arr[i]);
    }
    return 0;
}
```

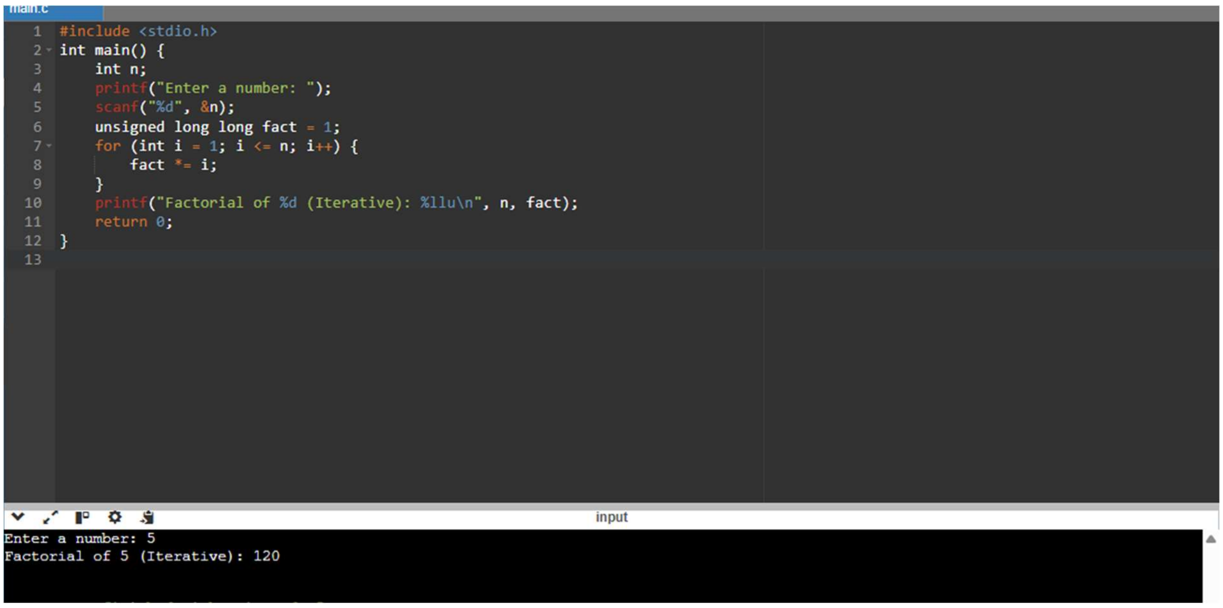
```
Enter number of elements: 5
Enter numbers:
12 43 55 22 14

Checking using bitwise AND:
12 is Even
43 is Odd
55 is Odd
22 is Even
14 is Even
```

5. Factorial of a Number

• Scenario: Calculate the number of possible arrangements.

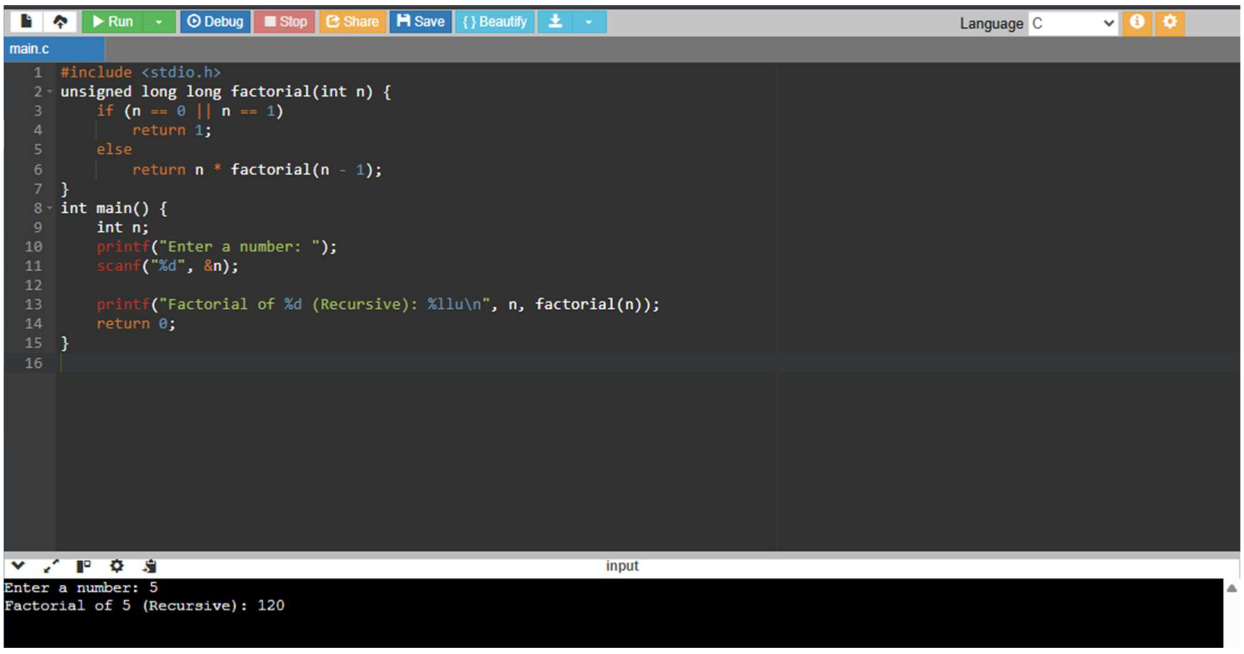
a) Iterative method (O(n), O(1)).

b) Recursive method (O(n), O(n) for call stack).



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

```
Enter a number: 5
Factorial of 5 (Iterative): 120
```



```c
#include <stdio.h>
unsigned long long factorial(int n) {
    if (n == 0 || n == 1)
        return 1;
    else
        return n * factorial(n - 1);
}
int main() {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);

    printf("Factorial of %d (Recursive): %llu\n", n, factorial(n));
    return 0;
}
```

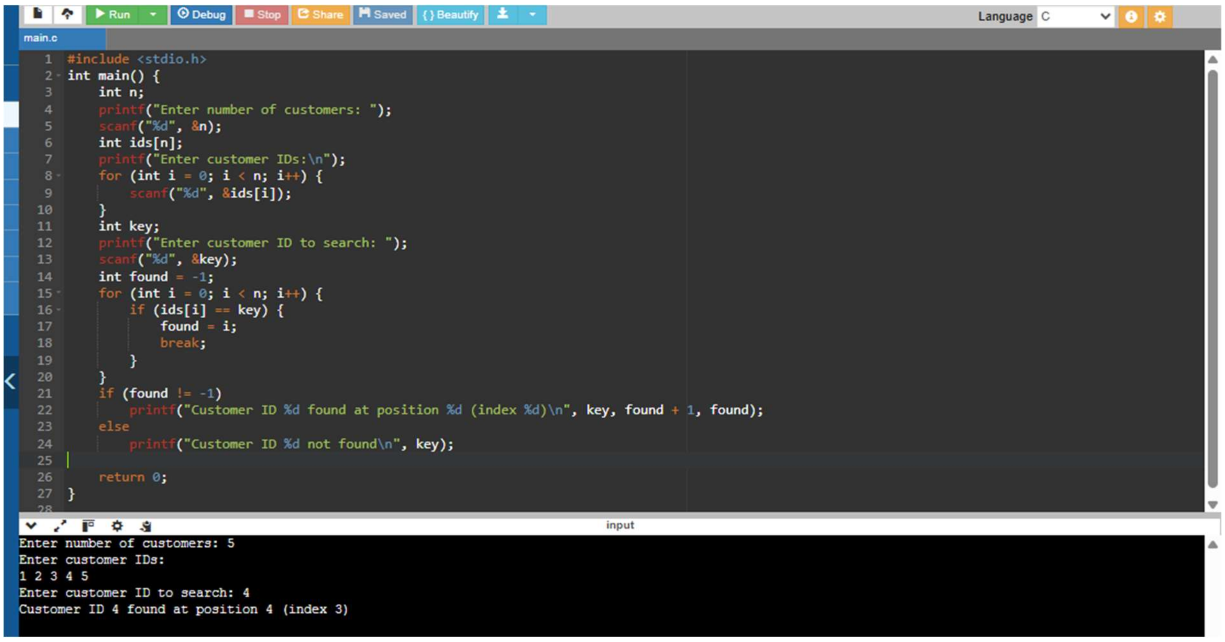
```
Enter a number: 5
Factorial of 5 (Recursive): 120
```

6. Linear Search

• Scenario: Search for a customer ID in a small dataset.

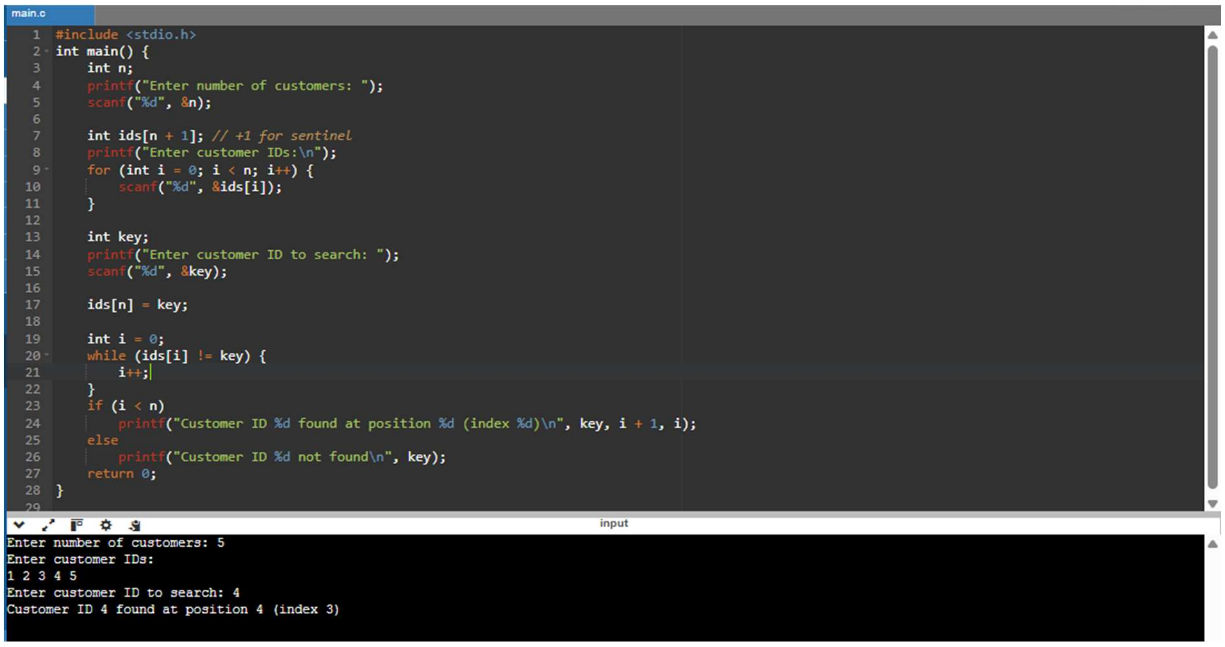
a) Implement basic linear search (O(n), O(1)).

b) Optimize by using sentinel method to reduce comparisons (still O(n), but fewer operations).



```c
#include <stdio.h>
int main() {
    int n;
    printf("Enter number of customers: ");
    scanf("%d", &n);
    int ids[n];
    printf("Enter customer IDs:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &ids[i]);
    }
    int key;
    printf("Enter customer ID to search: ");
    scanf("%d", &key);
    int found = -1;
    for (int i = 0; i < n; i++) {
        if (ids[i] == key) {
            found = i;
            break;
        }
    }
    if (found != -1)
        printf("Customer ID %d found at position %d (index %d)\n", key, found + 1, found);
    else
        printf("Customer ID %d not found\n", key);

    return 0;
}
```

```
Enter number of customers: 5
Enter customer IDs:
1 2 3 4 5
Enter customer ID to search: 4
Customer ID 4 found at position 4 (index 3)
```



```c
#include <stdio.h>
int main() {
    int n;
    printf("Enter number of customers: ");
    scanf("%d", &n);

    int ids[n + 1]; // +1 for sentinel
    printf("Enter customer IDs:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &ids[i]);
    }

    int key;
    printf("Enter customer ID to search: ");
    scanf("%d", &key);

    ids[n] = key;

    int i = 0;
    while (ids[i] != key) {
        i++;
    }
    if (i < n)
        printf("Customer ID %d found at position %d (index %d)\n", key, i + 1, i);
    else
        printf("Customer ID %d not found\n", key);
    return 0;
}
```

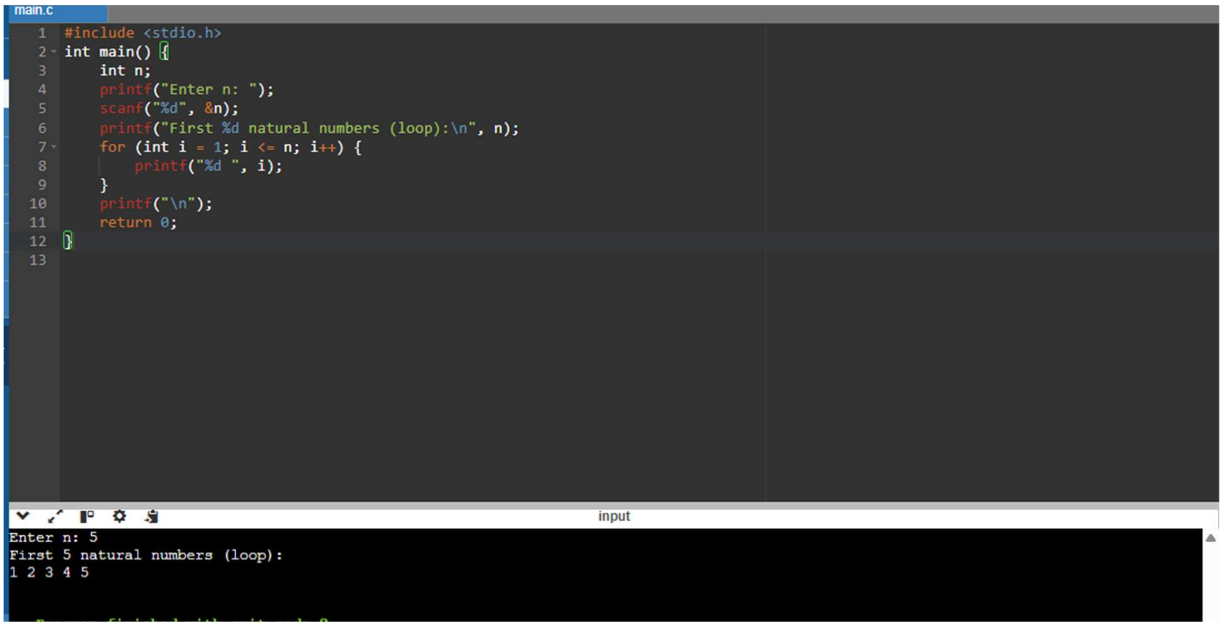
```
Enter number of customers: 5
Enter customer IDs:
1 2 3 4 5
Enter customer ID to search: 4
Customer ID 4 found at position 4 (index 3)
```

7. Print First n Natural Numbers

• Scenario: Generate first n natural numbers for a report.

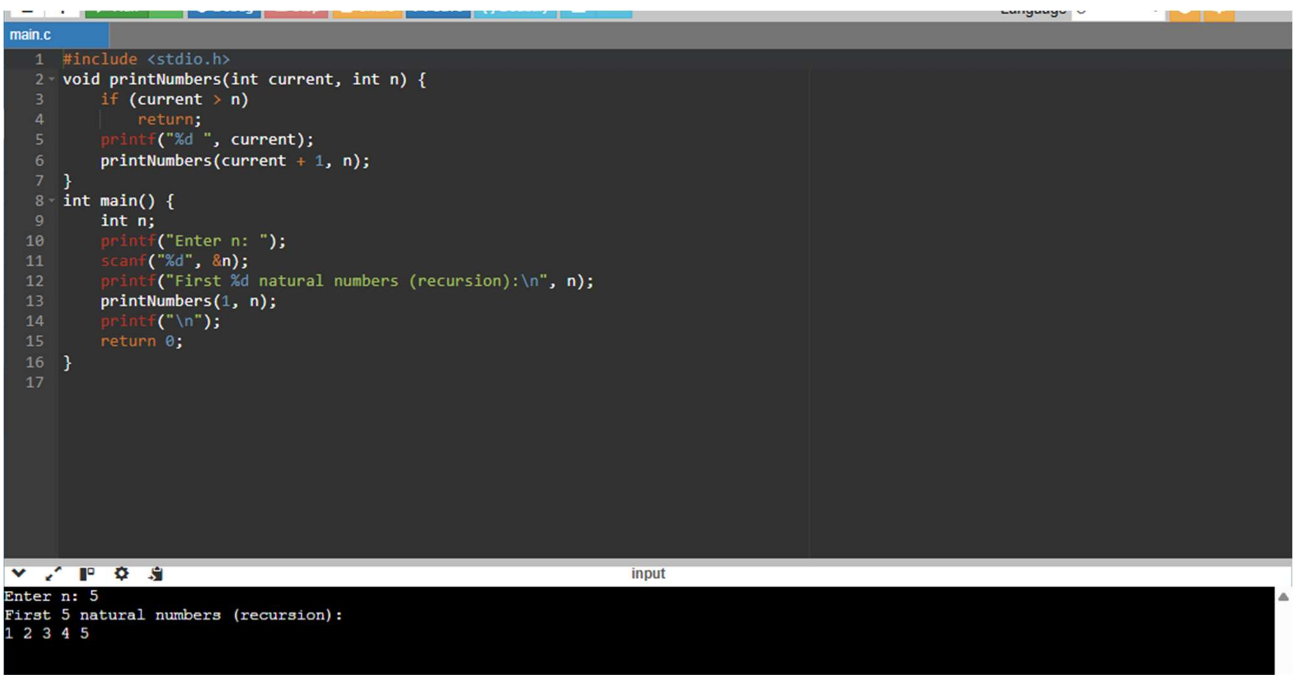
a) Using a simple for loop (O(n), O(1)).

b) Using recursion (O(n), O(n)).



```c
#include <stdio.h>
int main() {
    int n;
    printf("Enter n: ");
    scanf("%d", &n);
    printf("First %d natural numbers (loop):\n", n);
    for (int i = 1; i <= n; i++) {
        printf("%d ", i);
    }
    printf("\n");
    return 0;
}
```

```
Enter n: 5
First 5 natural numbers (loop):
1 2 3 4 5
```



```c
#include <stdio.h>
void printNumbers(int current, int n) {
    if (current > n)
        return;
    printf("%d ", current);
    printNumbers(current + 1, n);
}
int main() {
    int n;
    printf("Enter n: ");
    scanf("%d", &n);
    printf("First %d natural numbers (recursion):\n", n);
    printNumbers(1, n);
    printf("\n");
    return 0;
}
```

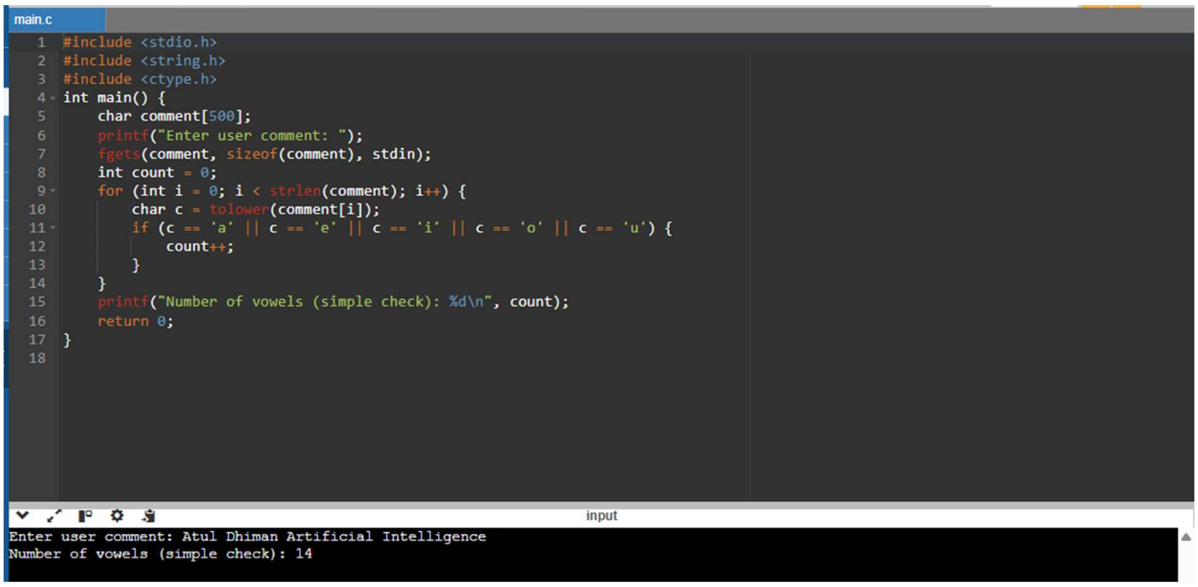
```
Enter n: 5
First 5 natural numbers (recursion):
1 2 3 4 5
```

8. Count Vowels in a String

• Scenario: Analyze user comments to count number of vowels.

a) Traverse string and check each character (O(n), O(1)).

b) Use a lookup table (array of 256 size) to speed up vowel checking (O(n), O(1) but extra space).

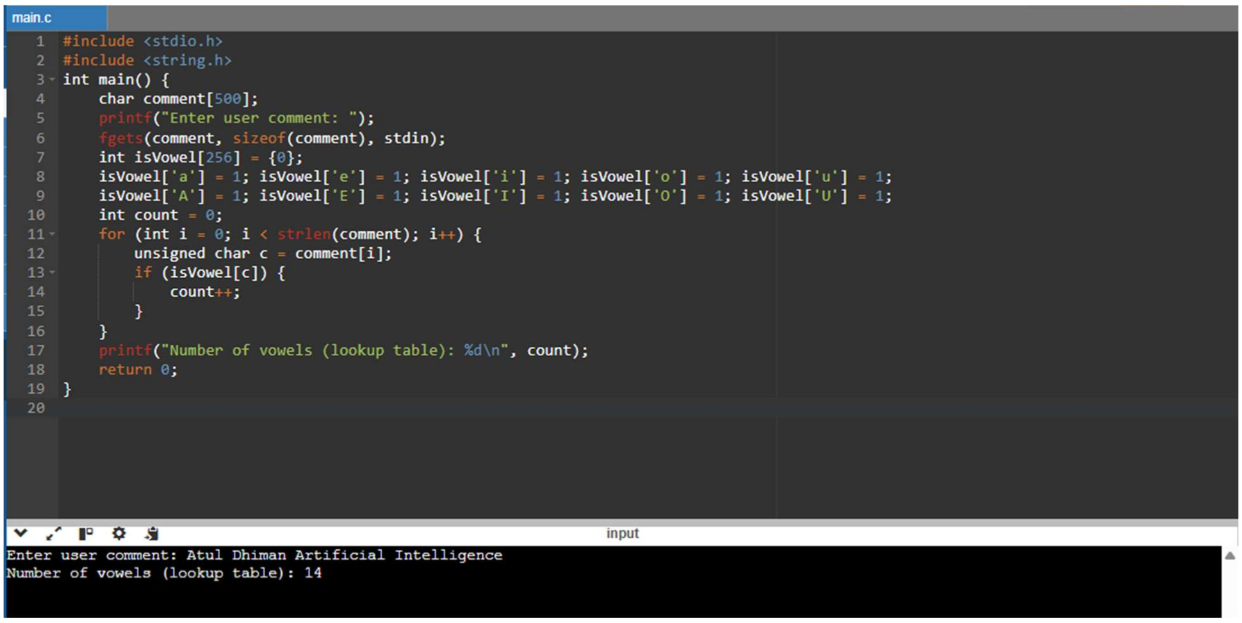


main.c

```c
#include <stdio.h>
#include <string.h>
#include <ctype.h>
int main() {
    char comment[500];
    printf("Enter user comment: ");
    fgets(comment, sizeof(comment), stdin);
    int count = 0;
    for (int i = 0; i < strlen(comment); i++) {
        char c = tolower(comment[i]);
        if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u') {
            count++;
        }
    }
    printf("Number of vowels (simple check): %d\n", count);
    return 0;
}

```

input

```
Enter user comment: Atul Dhiman Artificial Intelligence
Number of vowels (simple check): 14
```



main.c

```c
#include <stdio.h>
#include <string.h>
int main() {
    char comment[500];
    printf("Enter user comment: ");
    fgets(comment, sizeof(comment), stdin);
    int isVowel[256] = {0};
    isVowel['a'] = 1; isVowel['e'] = 1; isVowel['i'] = 1; isVowel['o'] = 1; isVowel['u'] = 1;
    isVowel['A'] = 1; isVowel['E'] = 1; isVowel['I'] = 1; isVowel['O'] = 1; isVowel['U'] = 1;
    int count = 0;
    for (int i = 0; i < strlen(comment); i++) {
        unsigned char c = comment[i];
        if (isVowel[c]) {
            count++;
        }
    }
    printf("Number of vowels (lookup table): %d\n", count);
    return 0;
}

```

input

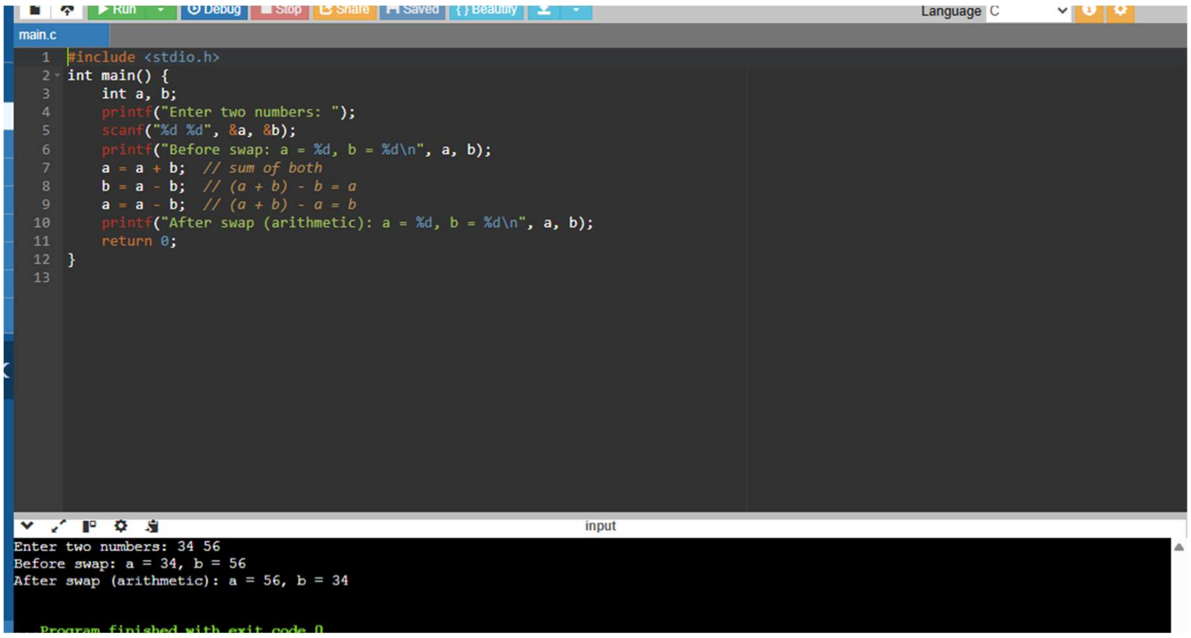
```
Enter user comment: Atul Dhiman Artificial Intelligence
Number of vowels (lookup table): 14
```

9. Swap Two Numbers Without Temporary Variable

• Scenario: In embedded systems where memory is constrained.

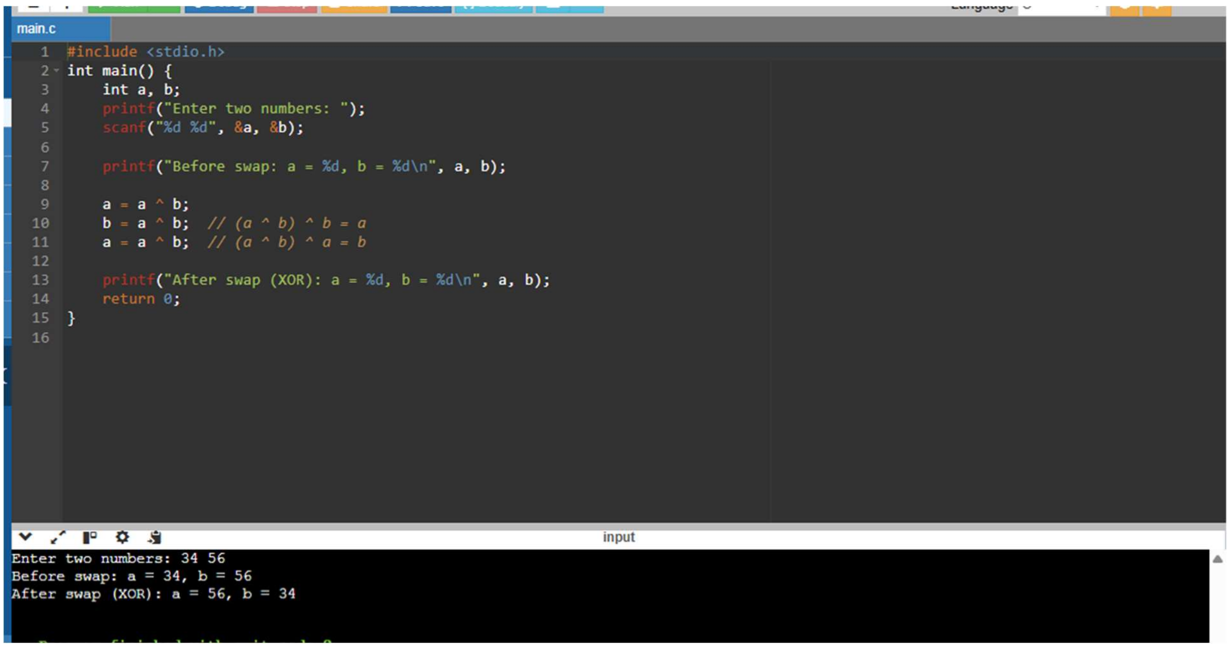
a) Using arithmetic (a = a + b; b = a - b; a = a - b).

b) Using bitwise XOR (a = a ^ b; b = a ^ b; a = a ^ b).



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

```
Enter two numbers: 34 56
Before swap: a = 34, b = 56
After swap (arithmetic): a = 56, b = 34

...Program finished with exit code 0
```



```c
#include <stdio.h>
int main() {
    int a, b;
    printf("Enter two numbers: ");
    scanf("%d %d", &a, &b);

    printf("Before swap: a = %d, b = %d\n", a, b);

    a = a ^ b;
    b = a ^ b;  // (a ^ b) ^ b = a
    a = a ^ b;  // (a ^ b) ^ a = b

    printf("After swap (XOR): a = %d, b = %d\n", a, b);
    return 0;
}
```

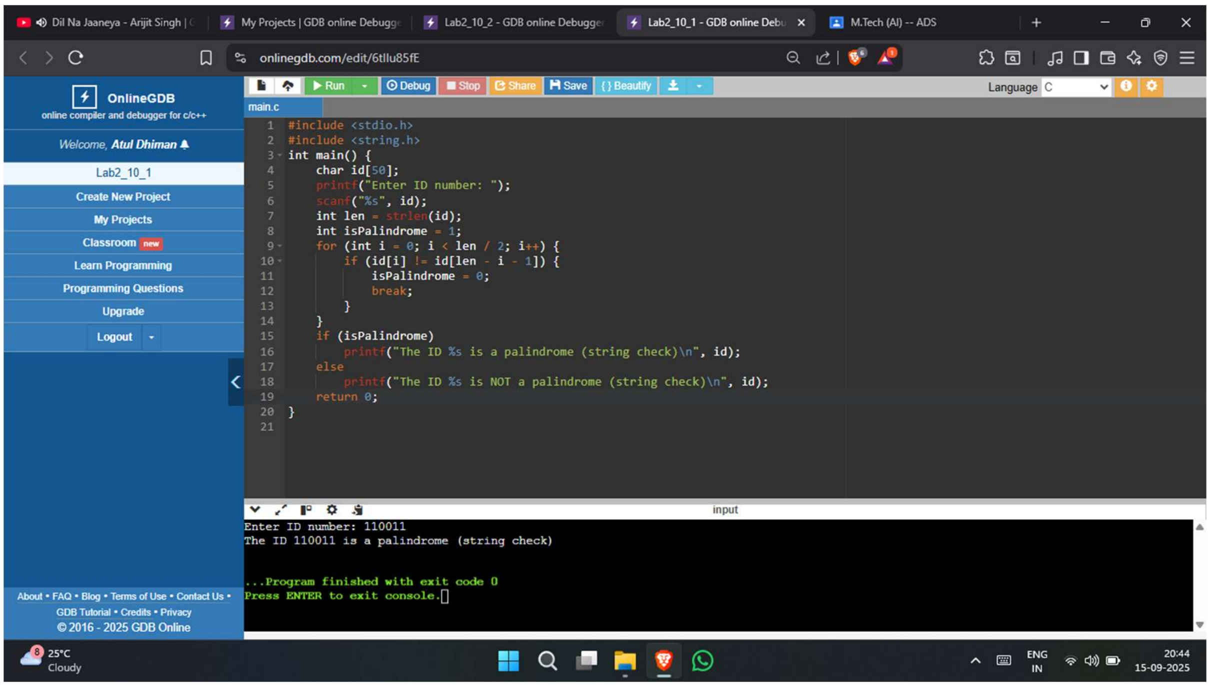
```
Enter two numbers: 34 56
Before swap: a = 34, b = 56
After swap (XOR): a = 56, b = 34
```

10. Check Palindrome Number

• Scenario: Validate identification number (ID) is symmetric.

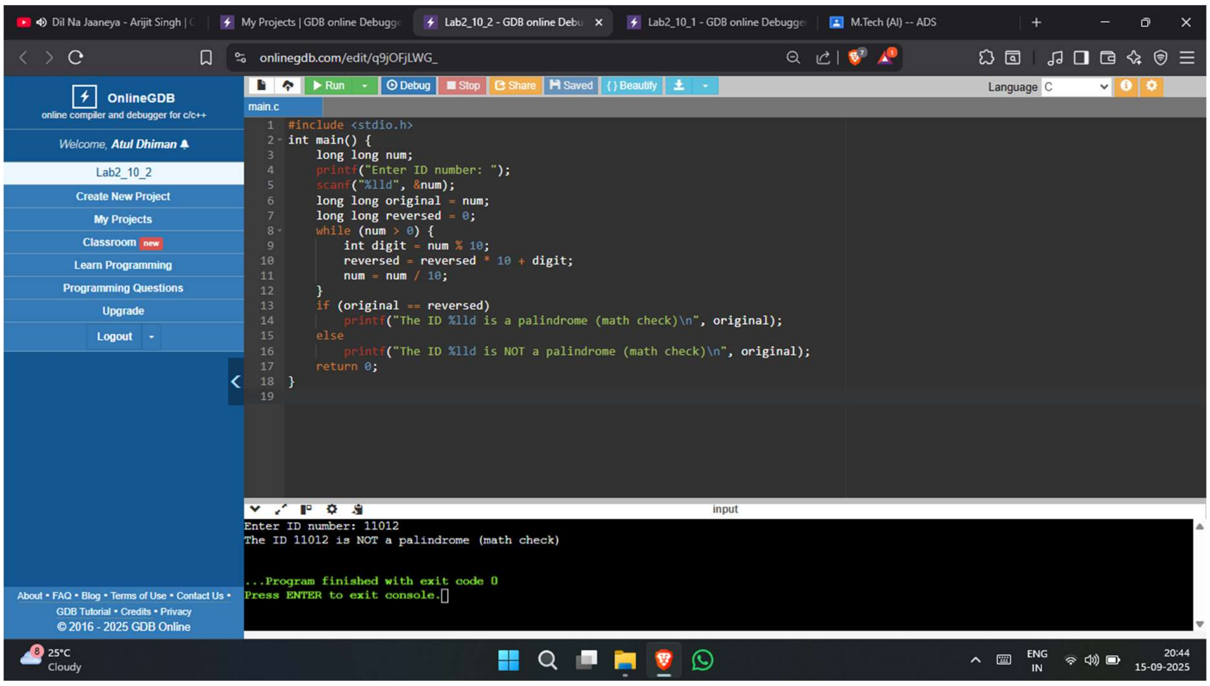
a) Convert number to string and check (O(n), O(n)).

b) Mathematical method: Reverse digits without converting (O (log n), O (1)).



```c
#include <stdio.h>
#include <string.h>
int main() {
    char id[50];
    printf("Enter ID number: ");
    scanf("%s", id);
    int len = strlen(id);
    int isPalindrome = 1;
    for (int i = 0; i < len / 2; i++) {
        if (id[i] != id[len - i - 1]) {
            isPalindrome = 0;
            break;
        }
    }
    if (isPalindrome)
        printf("The ID %s is a palindrome (string check)\n", id);
    else
        printf("The ID %s is NOT a palindrome (string check)\n", id);
    return 0;
}
```

```
Enter ID number: 110011
The ID 110011 is a palindrome (string check)


...Program finished with exit code 0
Press ENTER to exit console.
```



```c
#include <stdio.h>
int main() {
    long long num;
    printf("Enter ID number: ");
    scanf("%lld", &num);
    long long original = num;
    long long reversed = 0;
    while (num > 0) {
        int digit = num % 10;
        reversed = reversed * 10 + digit;
        num = num / 10;
    }
    if (original == reversed)
        printf("The ID %lld is a palindrome (math check)\n", original);
    else
        printf("The ID %lld is NOT a palindrome (math check)\n", original);
    return 0;
}
```

```
Enter ID number: 11012
The ID 11012 is NOT a palindrome (math check)


...Program finished with exit code 0
Press ENTER to exit console.
```