DAY 3 WWC OUESTIONS

Name – Rohit

Uid – 22BCS15476

Section – 620/B

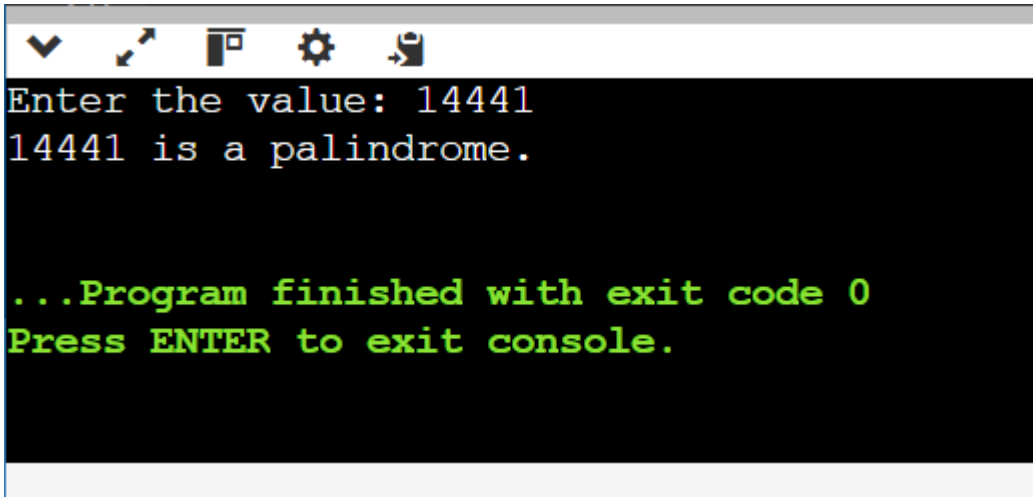
Q1. write a c++ program check if the number is palandrome or not using function

Ans

```
#include <iostream>
using namespace std;
int isPalindrome(int number) {
    int on = number;
    int rn = 0;

    while (number > 0) {
        int digit = number % 10;
        rn = rn * 10 + digit;
        number /= 10;
    }

    return on == rn;
}

int main() {
    int n;
    cout << "Enter the value: ";
    cin >> n;

    if (isPalindrome(n)) {
```

```
        cout << n << " is a palindrome." << endl;
    } else {
        cout << n << " is not a palindrome." << endl;
    }

    return 0;
}
```

Output



Q2c++ program to create a simple calculator for the basic operation like addition subtraction multiply or divide

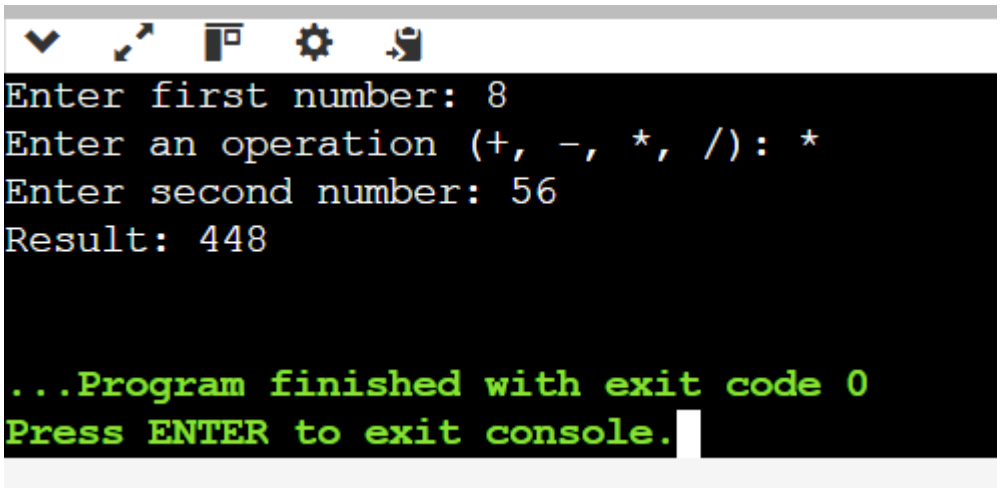
Ans

```
#include <iostream>

using namespace std;


void add(double a, double b) {
    cout << "Result: " << a + b << endl;
}


void subtract(double a, double b) {
    cout << "Result: " << a - b << endl;
```

```
}

void multiply(double a, double b) {
    cout << "Result: " << a * b << endl;
}

void divide(double a, double b) {
    if (b != 0) {
        cout << "Result: " << a / b << endl;
    } else {
        cout << "Error: Division by zero is not allowed." << endl;
    }
}

int main() {
    double num1, num2;
    char operation;

    cout << "Enter first number: ";
    cin >> num1;
    cout << "Enter an operation (+, -, *, /): ";
    cin >> operation;
    cout << "Enter second number: ";
    cin >> num2;

    switch (operation) {
        case '+':
            add(num1, num2);
            break;
```

```
        case '-':
            subtract(num1, num2);
            break;
        case '*':
            multiply(num1, num2);
            break;
        case '/':
            divide(num1, num2);
            break;
        default:
            cout << "Invalid operation." << endl;
            break;
    }

    return 0;
}
```

Output



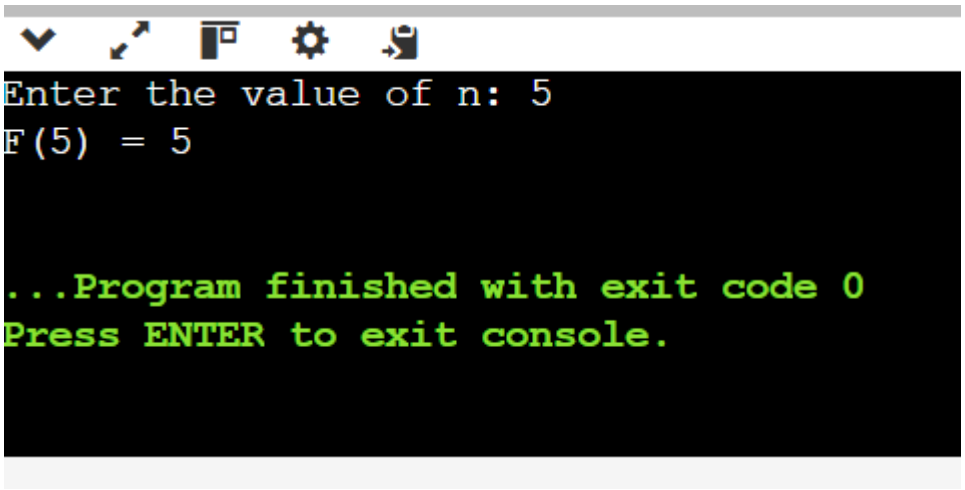
Q3.fibonnica series

Ans

```
#include <iostream>
using namespace std;
```

```
int fibonacci(int n) {
   if (n <= 1) {
      return n;
   }
   return fibonacci(n - 1) + fibonacci(n - 2);
}

int main() {
   int n;
   cout << "Enter the value of n: ";
   cin >> n;
   cout << "F(" << n << ") = " << fibonacci(n) << endl;
   return 0;
}
```

Output



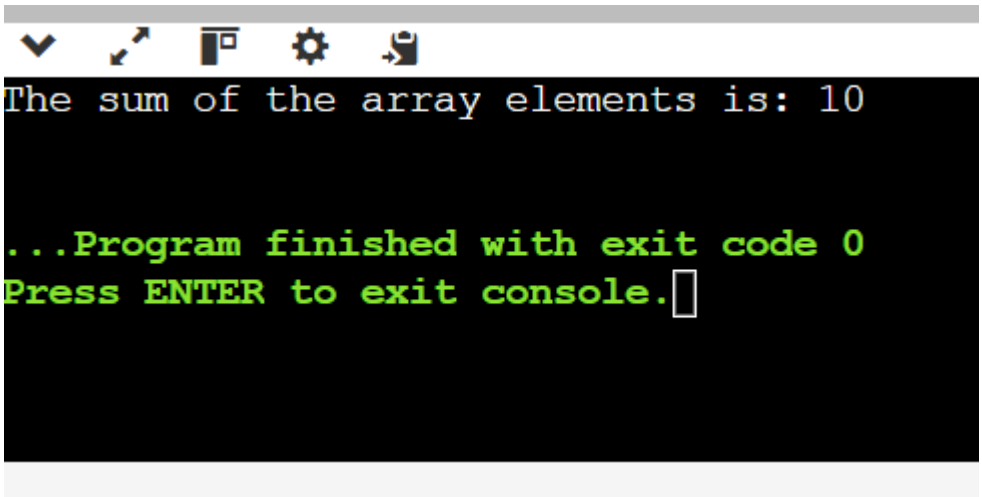
Q4. Given an array of integers, find sum of array elements using recursion.

Ans

```
#include <iostream>
using namespace std;
```

```
int sumArray(int arr[], int n) {
    if (n <= 0) {
        return 0;
    }
    return arr[n-1] + sumArray(arr, n-1);
}

int main() {
    int arr[] = {1, 2, 3, 4};
    int n = sizeof(arr) / sizeof(arr[0]);

    cout << "The sum of the array elements is: " << sumArray(arr, n) << endl;

    return 0;
}
```

Output



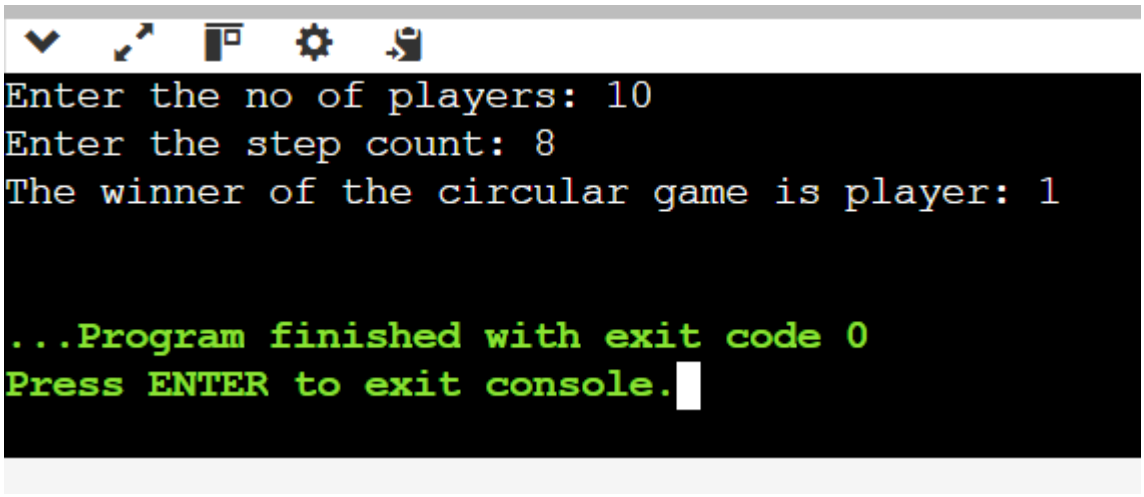
Q5.find the winner of the circular game

Ans

```
#include <iostream>
using namespace std;
```

```
int findWinner(int n, int k) {
    if (n == 1)
        return 0;
    else
        return (findWinner(n - 1, k) + k) % n;
}

int main() {
    int n, k;
    cout << "Enter the no of players: ";
    cin >> n;
    cout << "Enter the step count: ";
    cin >> k;

    int winner = findWinner(n, k) + 1;
    cout << "The winner of the circular game is player: " << winner << endl;

    return 0;
}
```

Output



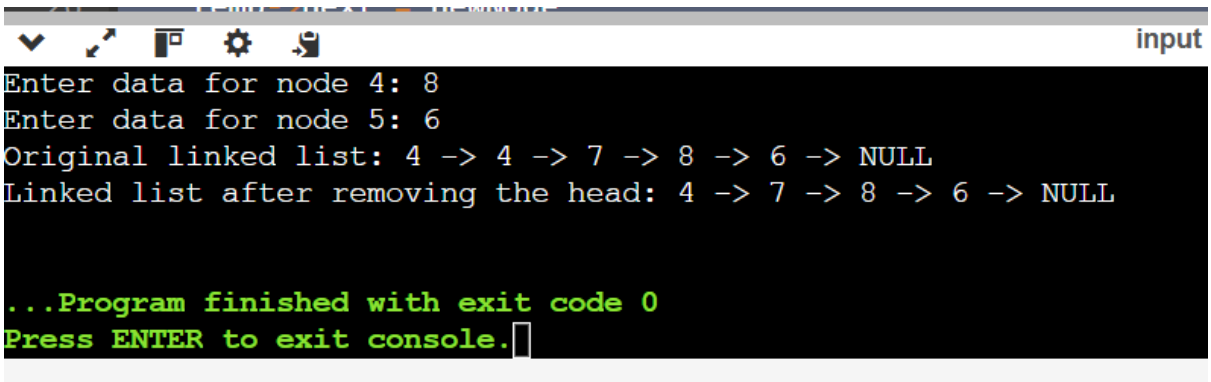
Q6.write a code in cpp to remove linked list in beginning

```
#include <iostream>
using namespace std;

struct Node {
    int data;
    Node* next;
};

Node* createNode(int data) {
    Node* newNode = new Node();
    newNode->data = data;
    newNode->next = nullptr;
    return newNode;
}

void insertNode(Node*& head, int data) {
    Node* newNode = createNode(data);
    if (!head) {
        head = newNode;
        return;
    }
    Node* temp = head;
    while (temp->next) {
        temp = temp->next;
    }
    temp->next = newNode;
}

void removeHead(Node*& head) {
```

```
    if (!head) {
        cout << "The list is already empty." << endl;
        return;
    }
    Node* temp = head;
    head = head->next;
    delete temp;
}

void displayList(Node* head) {
    Node* temp = head;
    while (temp) {
        cout << temp->data << " -> ";
        temp = temp->next;
    }
    cout << "NULL" << endl;
}

int main() {
    Node* head = nullptr;
    int n, data;

    cout << "Enter the number of nodes: ";
    cin >> n;
    for (int i = 0; i < n; ++i) {
        cout << "Enter data for node " << i + 1 << ": ";
        cin >> data;
        insertNode(head, data);
    }
```

```
    cout << "Original linked list: ";
    displayList(head);

    removeHead(head);

    cout << "Linked list after removing the head: ";
    displayList(head);

    Node* current = head;
    Node* next = nullptr;
    while (current) {
        next = current->next;
        delete current;
        current = next;
    }

    return 0;
}
```

Output



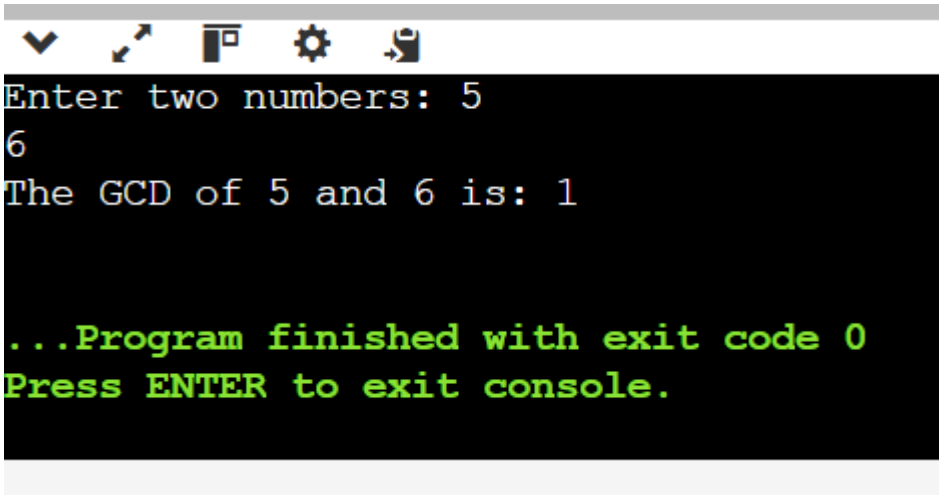
Q7. : Writer recursive function to compute the GCD of 2 numbers

Ans

```
#include <iostream>
```

```
using namespace std;

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

int main() {
    int num1, num2;
    cout << "Enter two numbers: ";
    cin >> num1 >> num2;

    cout << "The GCD of " << num1 << " and " << num2 << " is: " << gcd(num1, num2) << endl;

    return 0;
}
```

Output

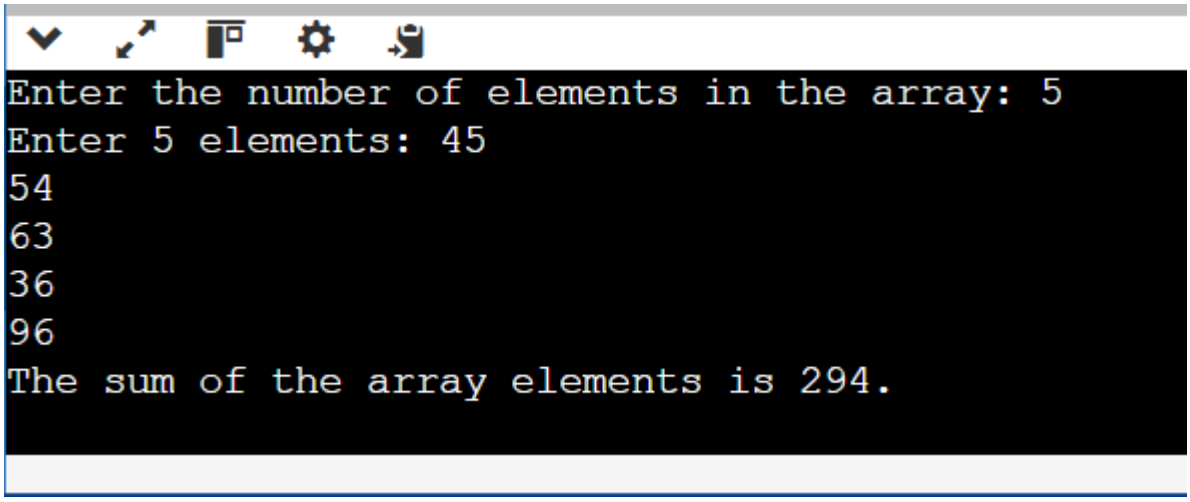


Q8. sum of array element using recursion function

Ans

```
#include <iostream>
using namespace std;
int sumOfArray(int arr[], int size) {
if (size == 0) {
return 0;
}
return arr[size - 1] + sumOfArray(arr, size - 1);
}
int main() {
int n;
cout << "Enter the number of elements in the array: ";
cin >> n;
if (n <= 0) {
cout << "Please enter a positive number of elements." << endl;
return 1;
}
int arr[n];
cout << "Enter " << n << " elements: ";
for (int i = 0; i < n; i++) {
cin >> arr[i];
}
cout << "The sum of the array elements is " << sumOfArray(arr, n) << "." << endl;
return 0;
}
```

Output



Q9. Give the head of the linked list reverse the nodes of list k at a time and return the modified time.

Ans

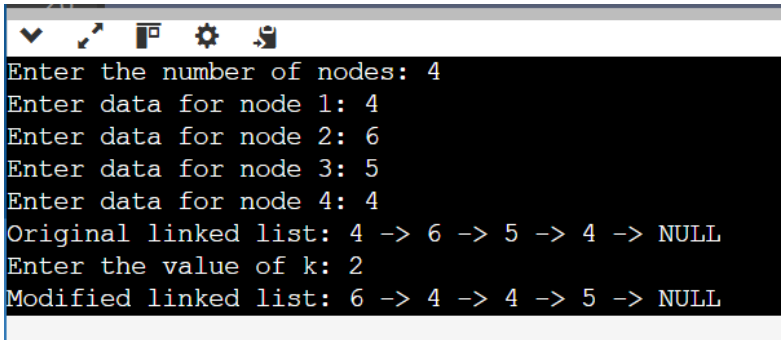
```
#include <iostream>

using namespace std;

struct Node {
    int data;
    Node* next;


    Node(int val) : data(val), next(nullptr) {}
};
void insertNode(Node*& head, int data) {
    Node* newNode = new Node(data);
    if (!head) {
        head = newNode;
        return;
    }
    Node* temp = head;
    while (temp->next) {
        temp = temp->next;
    }
```

```
    temp->next = newNode;
}

Node* reverseKGroup(Node* head, int k) {
    if (!head || k == 1) {
        return head;
    }

    Node* dummy = new Node(0);
    dummy->next = head;
    Node *curr = dummy, *next = dummy, *prev = dummy;
    int count = 0;
    while (curr->next) {
        curr = curr->next;
        count++;
    }

    while (count >= k) {
        curr = prev->next;
        next = curr->next;
        for (int i = 1; i < k; ++i) {
            curr->next = next->next;
            next->next = prev->next;
            prev->next = next;
            next = curr->next;
        }
        prev = curr;
        count -= k;
    }
```

```
    return dummy->next;
}
void displayList(Node* head) {
    while (head) {
        cout << head->data << " -> ";
        head = head->next;
    }
    cout << "NULL" << endl;
}

int main() {
    Node* head = nullptr;
    int n, data, k;
    cout << "Enter the number of nodes: ";
    cin >> n;
    for (int i = 0; i < n; ++i) {
        cout << "Enter data for node " << i + 1 << ": ";
        cin >> data;
        insertNode(head, data);
    }
    cout << "Original linked list: ";
    displayList(head);
    cout << "Enter the value of k: ";
    cin >> k;
    head = reverseKGroup(head, k);

    cout << "Modified linked list: ";
    displayList(head);
```

```
    return 0;
}
```

Output



Q10. sum of natural number using recursion function

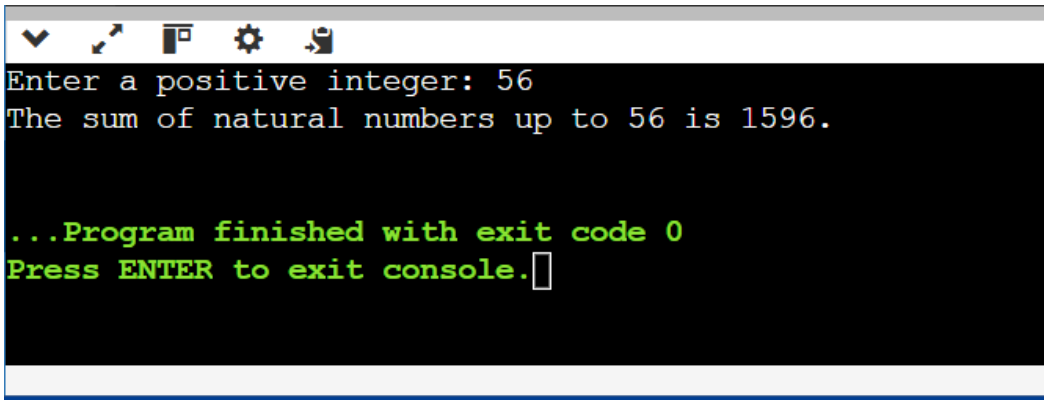
Ans

```
#include <iostream>
using namespace std;
int sumOfNaturalNumbers(int n) {
if (n == 0) {
return 0;
}
return n + sumOfNaturalNumbers(n - 1);
}
int main() {
int number;
cout << "Enter a positive integer: ";
cin >> number;
if (number < 0) {
cout << "Please enter a positive integer." << endl;
} else {
cout << "The sum of natural numbers up to " << number << " is " <<
sumOfNaturalNumbers(number) << "." << endl;
}
```

return 0;

}

Output



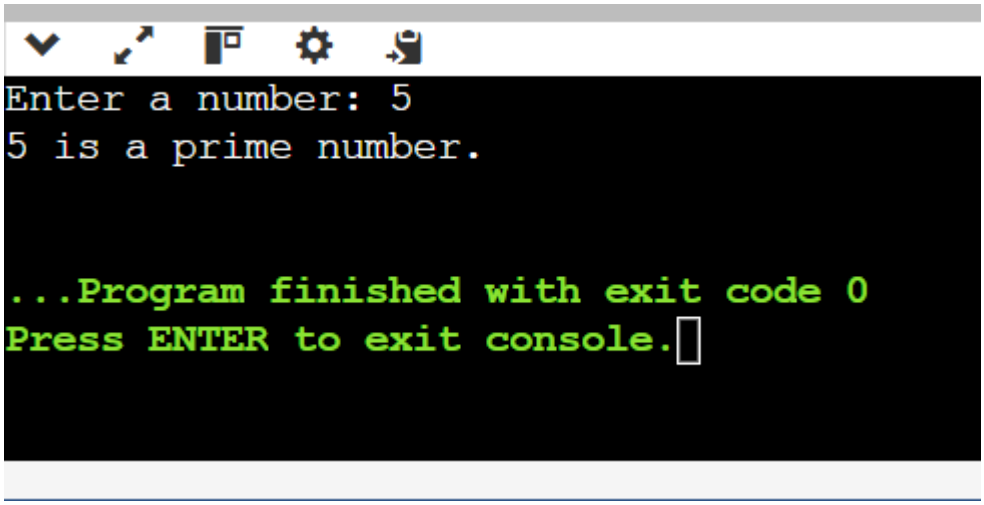
Q11. Function method to find the no is prime

Ans

```
#include <iostream>
using namespace std;
bool isPrime(int num) {
    if (num <= 1) return false;
    if (num <= 3) return true;
    if (num % 2 == 0 || num % 3 == 0) return false;
    for (int i = 5; i * i <= num; i += 6) {
        if (num % i == 0 || num % (i + 2) == 0)
            return false;
    }
    return true;
}
int main() {
    int number;
    cout << "Enter a number: ";
    cin >> number;
        if (isPrime(number)) {
        cout << number << " is a prime number." << endl;
```

```
    } else {
        cout << number << " is not a prime number." << endl;
    }

    return 0;
}
```

Output



Q12. Swap two numbers

Ans

```
#include <iostream>
using namespace std;
void swapNumbers(int &a, int &b) {
int temp = a;
a = b;
b = temp;
}
int main() {
int num1, num2;
cout << "Enter the first number: ";
cin >> num1;
cout << "Enter the second number: ";
```

```
cin >> num2;
cout << "\nBefore swapping: num1 = " << num1 << ", num2 = " << num2 << endl;
swapNumbers(num1, num2);
cout << "After swapping: num1 = " << num1 << ", num2 = " << num2 << endl;
return 0;
}
```

Output