DOMAIN WINTER WINNING CAMP

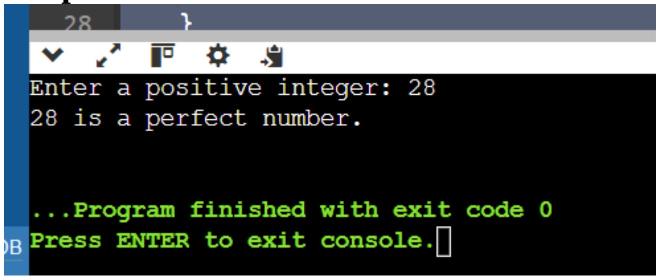
Student Name: Sumit Kumar **UID:** 22BCS15303
Branch: BE-CSE **Section/Group:** 620-A
Semester: 5th **Date of Performance:**23/12/24

Problem 1

1. **Aim: Perfect number**
2. **Code:**

```
#include <iostream>

bool isPerfectNumber(int number) {
    if (number <= 0) {
        return false; // Perfect numbers are positive integers
    }

    int sum = 0;

    for (int i = 1; i <= number / 2; ++i) {
        if (number % i == 0) {
            sum += i;
        }
    }
    return sum == number;
}

int main() {
    int number;

    std::cout << "Enter a positive integer: ";
    std::cin >> number;

    if (isPerfectNumber(number)) {
        std::cout << number << " is a perfect number." << std::endl;
    } else {
        std::cout << number << " is not a perfect number." << std::endl;
    }

    return 0;
}
```

3. **Output:**



```
Enter a positive integer: 28
28 is a perfect number.


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

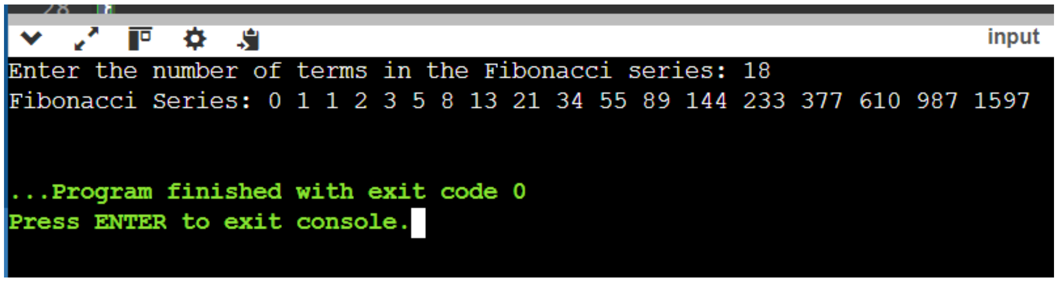
Problem 2

1. **Aim: Fibonacci series using recursion**

2. **Code:**

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

// Function to calculate Fibonacci number using recursion
int fibonacci(int n) {
    if (n <= 0) {
        return 0; // Base case: F(0) = 0
    } else if (n == 1) {
        return 1; // Base case: F(1) = 1
    } else {
        return fibonacci(n - 1) + fibonacci(n - 2); // Recursive case
    }
}

int main() {
    int n;

    cout << "Enter the number of terms in the Fibonacci series: ";
    cin >> n;

    cout << "Fibonacci Series: ";
    for (int i = 0; i < n; i++) {
        cout << fibonacci(i) << " ";
    }
    cout << endl;

    return 0;
  }
```

3. **Output:**



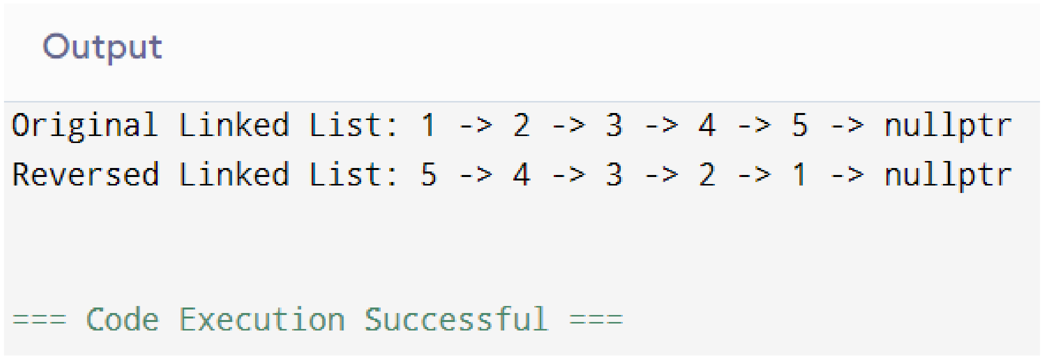
```
Enter the number of terms in the Fibonacci series: 18
Fibonacci Series: 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597


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

Problem 3

1. **Aim: Reversal of linked List**

2. **Code:**

```
#include <iostream>
using namespace std;
struct Node {
    int data;
    Node* next;

    Node(int val) : data(val), next(nullptr) {}
};

// Function to reverse the linked list
Node* reverseLinkedList(Node* head) {
    Node* prev = nullptr;
    Node* current = head;
    Node* next = nullptr;

    while (current != nullptr) {
        next = current->next; // Store next node
        current->next = prev; // Reverse the link
        prev = current;       // Move prev and current one step forward
        current = next;
    }
    return prev; // New head of the reversed list
}
void printList(Node* head) {
    Node* current = head;
    while (current != nullptr) {
        cout << current->data << " -> ";
        current = current->next;
    }
    cout << "nullptr" << endl;
}
int main() {
    // Creating a linked list: 1 -> 2 -> 3 -> 4 -> 5 -> nullptr
    Node* head = new Node(1);
    head->next = new Node(2);
    head->next->next = new Node(3);
    head->next->next->next = new Node(4);
    head->next->next->next->next = new Node(5);

    cout << "Original Linked List: ";
    printList(head);
    head = reverseLinkedList(head);
    cout << "Reversed Linked List: ";
    printList(head);
    Node* current = head;
    while (current != nullptr) {
        Node* next = current->next;
        delete current;
        current = next;
    }
    return 0;
```



Output

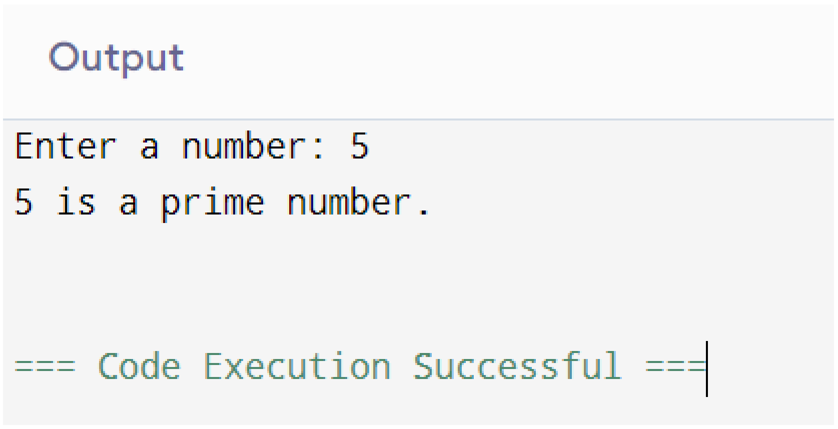
```
Original Linked List: 1 -> 2 -> 3 -> 4 -> 5 -> nullptr
Reversed Linked List: 5 -> 4 -> 3 -> 2 -> 1 -> nullptr


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

Problem 4

1. **Aim: Check whether a number is prime or not**

2. **Code:**

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

// Function to check if a number is prime
bool isPrime(int n) {
    // Handle special cases
    if (n <= 1) return false; // 0 and 1 are not prime numbers
    if (n <= 3) return true;  // 2 and 3 are prime numbers

    // Check for even numbers and multiples of 3
    if (n % 2 == 0 || n % 3 == 0) return false;

    // Check for factors from 5 to sqrt(n)
    for (int i = 5; i * i <= n; i += 6) {
        if (n % i == 0 || n % (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

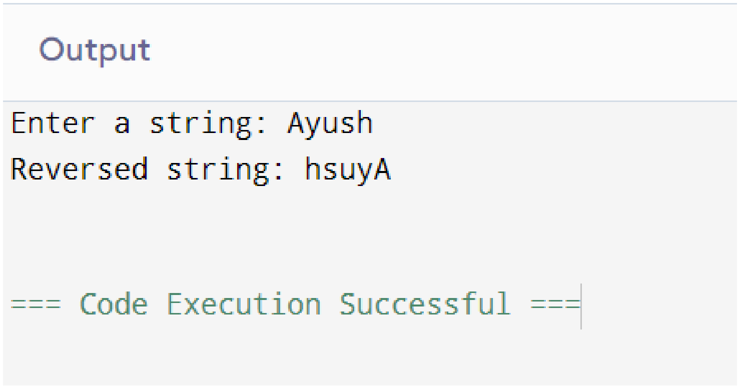
```
Enter a number: 5
5 is a prime number.


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

Problem 5

1. **Aim: Reverse the string**

2. **Code:**

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

// Function to reverse a string
string reverseString(string str) {
    int n = str.length();
    for (int i = 0; i < n / 2; i++) {
        // Swap characters
        swap(str[i], str[n - i - 1]);
    }
    return str;
}

int main() {
    string input;

    cout << "Enter a string: ";
    getline(cin, input); // Use getline to allow spaces in the input

    string reversed = reverseString(input);
    cout << "Reversed string: " << reversed << endl;

    return 0;
}
```



Output

```
Enter a string: Ayush
Reversed string: hsuyA


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

Problem 6

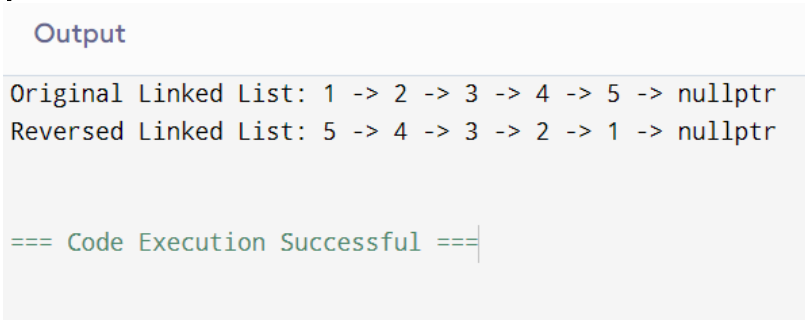
1. **Aim: Add two numbers**
2. **Code:**

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

int main() {
    // Declare variables to hold the numbers
    double num1, num2, sum;

    // Prompt the user for input
    cout << "Enter the first number: ";
    cin >> num1;

    cout << "Enter the second number: ";
    cin >> num2;

    // Calculate the sum
    sum = num1 + num2;

    // Display the result
    cout << "The sum of " << num1 << " and " << num2 << " is: " << sum << endl;

    return 0;
}
```

Problem 7

1. **Aim: Reverse the Linkedlist and return the reversed list**
2. **Code:**

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

// Node structure
struct Node {
    int data;
    Node* next;

    Node(int val) : data(val), next(nullptr) {}
};

// Function to reverse the linked list
Node* reverseLinkedList(Node* head) {
    Node* prev = nullptr;
    Node* current = head;
    Node* next = nullptr;

    while (current != nullptr) {
        next = current->next; // Store next node
        current->next = prev; // Reverse the link
        prev = current;       // Move prev and current one step forward
        current = next;
    }
    return prev; // New head of the reversed list
}

// Function to print the linked list
void printList(Node* head) {
    Node* current = head;
    while (current != nullptr) {
        cout << current->data << " -> ";
        current = current->next;
    }
    cout << "nullptr" << endl;
}

// Main function to demonstrate the reversal
int main() {
    // Creating a linked list: 1 -> 2 -> 3 -> 4 -> 5 -> nullptr
    Node* head = new Node(1);
    head->next = new Node(2);
    head->next->next = new Node(3);
    head->next->next->next = new Node(4);
    head->next->next->next->next = new Node(5);

    cout << "Original Linked List: ";
    printList(head);

    // Reversing the linked list
    head = reverseLinkedList(head);

    cout << "Reversed Linked List: ";
    printList(head);

    // Freeing the allocated memory (optional, but good practice)
    Node* current = head;
    while (current != nullptr) {
        Node* next = current->next;
        delete current;
        current = next;
    }

    return 0;
}
```



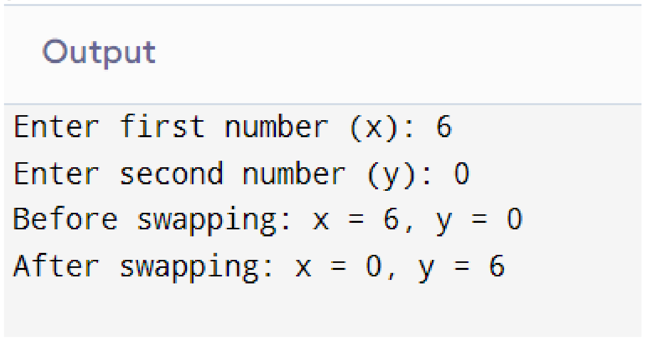
Output

```
Original Linked List: 1 -> 2 -> 3 -> 4 -> 5 -> nullptr
Reversed Linked List: 5 -> 4 -> 3 -> 2 -> 1 -> nullptr


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

Problem 8

1. **Aim: Implement the function that swipe to variable using pass by reference**
2. **Code:**

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

// Function to swap two integers using pass by reference
void swap(int& a, int& b) {
    int temp = a; // Store the value of a in a temporary variable
    a = b;        // Assign the value of b to a
    b = temp;     // Assign the value of temp (original a) to b
}

int main() {
    int x, y;

    // Input two integers from the user
    cout << "Enter first number (x): ";
    cin >> x;
    cout << "Enter second number (y): ";
    cin >> y;
    cout << "Before swapping: x = " << x << ", y = " << y << endl;

    // Call the swap function
    swap(x, y);

    cout << "After swapping: x = " << x << ", y = " << y << endl;

    return 0;
}
```



Output

```
Enter first number (x): 6
Enter second number (y): 0
Before swapping: x = 6, y = 0
After swapping: x = 0, y = 6
```

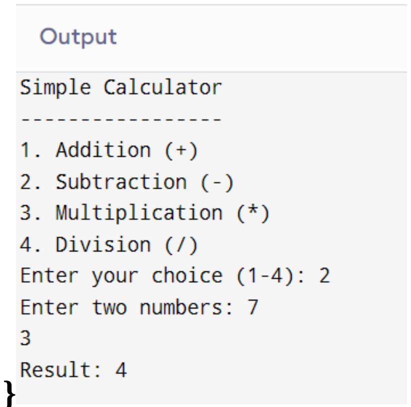
Problem 9

1 **Aim: Create a simple calculator**

2 **Code:**

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


float add(float num1, float num2) {
    return num1 + num2;
}
float subtract(float num1, float num2) {
    return num1 - num2;
}
float multiply(float num1, float num2) {
    return num1 * num2;
}
float divide(float num1, float num2) {
    if(num2 != 0)
        return num1 / num2;
    else
        return 0; // or handle division by zero error
}
int main() {
    float num1, num2, result;
    char op;
```

```
cout << "Simple Calculator\n";
cout << "-----------------\n";
cout << "1. Addition (+)\n";
cout << "2. Subtraction (-)\n";
cout << "3. Multiplication (*)\n";
cout << "4. Division (/)\n";
cout << "Enter your choice (1-4): ";
cin >> op;

cout << "Enter two numbers: ";
cin >> num1 >> num2;

switch(op) {
   case '1':
      result = add(num1, num2);
      break;
   case '2':
      result = subtract(num1, num2);
      break;
   case '3':
      result = multiply(num1, num2);
      break;
   case '4':
      result = divide(num1, num2);
      break;
   default:
      cout << "Invalid operator. Exiting.\n";
      return 1;
}

cout << "Result: " << result << endl;
return 0;
```



```
Simple Calculator
-----------------
1. Addition (+)
2. Subtraction (-)
3. Multiplication (*)
4. Division (/)
Enter your choice (1-4): 2
Enter two numbers: 7
3
Result: 4
```

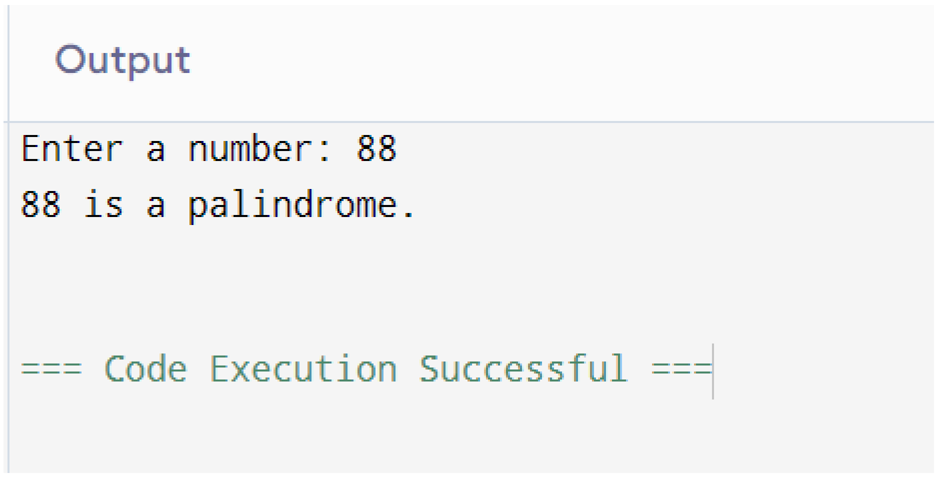
```
 }
```

Problem 10

1 **Aim: Check whether a number is palindrome or not**

2 **Code:**

```
#include <iostream>
#include <cmath>
bool isPalindrome(int num) {
    int reversed = 0;
    int original = num;
    // Reverse the number
    while (num != 0) {
        int digit = num % 10;
        reversed = reversed * 10 + digit;
        num /= 10;
    }

    // Compare the reversed number with the original
    return original == reversed;
}
int main() {
    int num;
    std::cout << "Enter a number: ";
    std::cin >> num;

    if (isPalindrome(num)) {
        std::cout << num << " is a palindrome." << std::endl;
    } else {
        std::cout << num << " is not a palindrome." << std::endl;
    }
    return 0;
  }
```


Output

```
Enter a number: 88
88 is a palindrome.


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