Name:Tarun

Uid: 22BCS15293

Section: 620/A

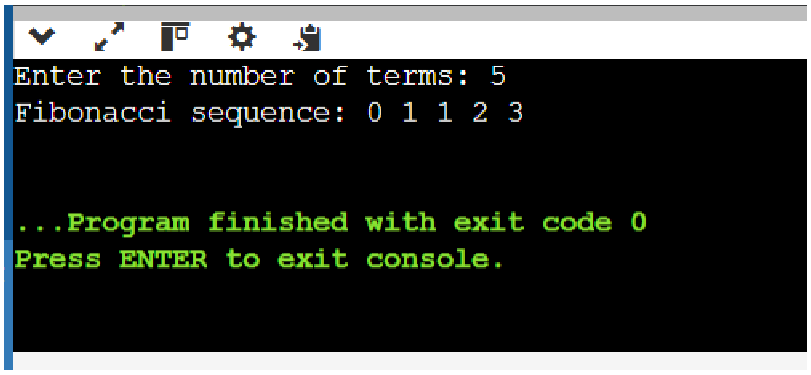
DAY 3

Ques 1: Fibonnacci Series Using Recursion.

Code//

```
#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 number of terms: ";
    cin >> n;
    cout << "Fibonacci sequence: ";
    for (int i = 0; i < n; i++) {
        cout << fibonacci(i) << " ";
    }
    cout << endl;

    return 0;
}
```

Output//



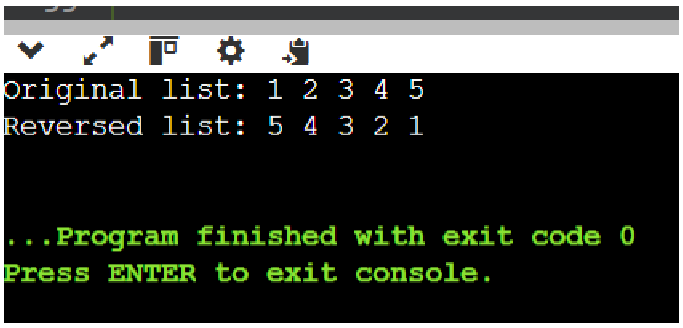
```
Enter the number of terms: 5
Fibonacci sequence: 0 1 1 2 3


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

Ques 2 : Reverse Linked List

Code//

```
#include <iostream>

using namespace std;

struct ListNode {

    int val;

    ListNode* next;

    ListNode(int x) : val(x), next(nullptr) {}

};


// Function to reverse a singly linked list

ListNode* reverseList(ListNode* head) {

    ListNode* prev = nullptr;

    ListNode* curr = head;

    ListNode* next = nullptr;


    while (curr != nullptr) {

        next = curr->next;  // Save the next node

        curr->next = prev;  // Reverse the current node's pointer

        prev = curr;        // Move prev to the current node

        curr = next;        // Move to the next node

    }
    return prev; // New head of the reversed list
}
void printList(ListNode* head) {
    while (head != nullptr) {
        cout << head->val << " ";
        head = head->next;
    }
    cout << endl;
}

int main() {
    ListNode* head = new ListNode(1);
    head->next = new ListNode(2);
    head->next->next = new ListNode(3);
    head->next->next->next = new ListNode(4);
    head->next->next->next->next = new ListNode(5);

    cout << "Original list: ";
    printList(head);
    head = reverseList(head);
    cout << "Reversed list: ";
    printList(head);

    return 0;
}
```

Output//

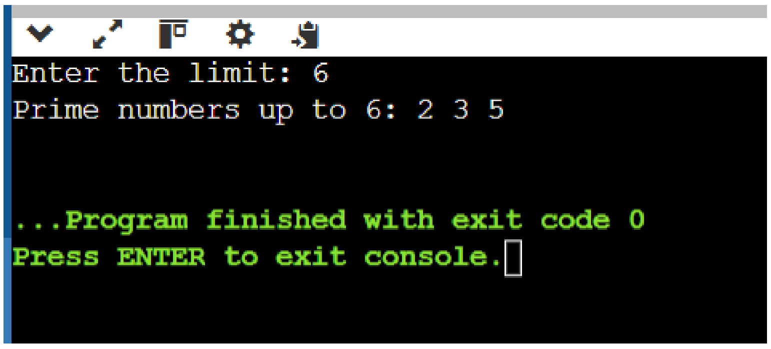

```
Original list: 1 2 3 4 5
Reversed list: 5 4 3 2 1


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

Ques 3 : Prime number

Code//

```
#include <iostream>

using namespace std;

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;

}

void printPrimes(int limit) {

  cout << "Prime numbers up to " << limit << ": ";

  for (int i = 2; i <= limit; i++) {

    if (isPrime(i)) {

      cout << i << " ";

    }

  }

  cout << endl;
}

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

  printPrimes(n);

  return 0;
}
```

Output//



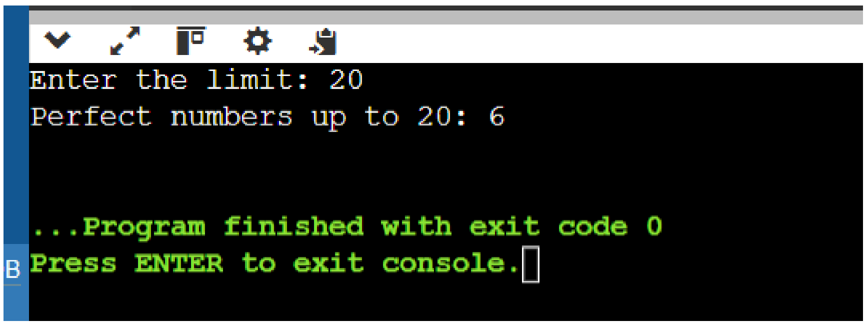
```
Enter the limit: 6
Prime numbers up to 6: 2 3 5


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

Ques 4: Perfect number

Code//

```
#include <iostream>
using namespace std;
bool isPerfect(int num) {
    if (num <= 1) {
        return false;
    }

    int sum = 1; // 1 is always a divisor
    for (int i = 2; i * i <= num; i++) {
        if (num % i == 0) {
            if (i == num / i) {
                sum += i; // Add the divisor only once if it is a square root
            } else {
                sum += i + num / i; // Add both divisors
            }
        }
    }

    return sum == num; // Check if the sum of divisors equals the number
}

// Function to print perfect numbers up to a given limit
void printPerfectNumbers(int limit) {
    cout << "Perfect numbers up to " << limit << ": ";
    for (int i = 2; i <= limit; i++) {
        if (isPerfect(i)) {
            cout << i << " ";
        }
    }
    cout << endl;
}

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


    return 0;
}
```

Output//



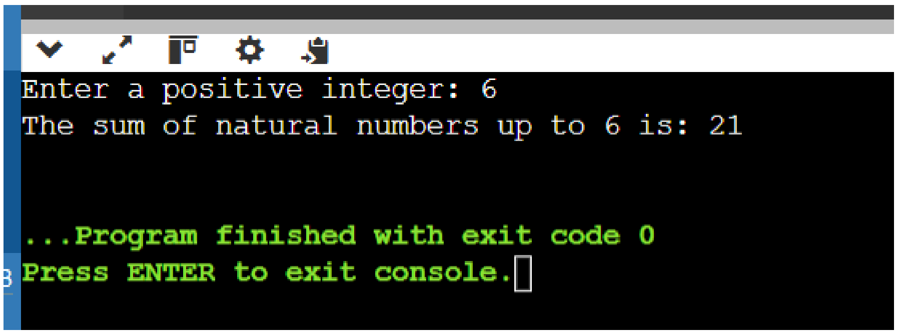
```
Enter the limit: 20
Perfect numbers up to 20: 6


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

Ques 5: Sum of natural number using recusion c++ code

Code//

```
#include <iostream>


using namespace std;


// Recursive function to calculate the sum of natural numbers up to n
int sumOfNaturalNumbers(int n) {
  // Base case: If n is 0, the sum is 0
  if (n == 0) {
    return 0;
  } else {
    // Recursive step: Add n to the sum of numbers up to n-1
    return n + sumOfNaturalNumbers(n - 1);
  }
}


int main() {
  int num;

  cout << "Enter a positive integer: ";
  cin >> num;

  if (num < 0) {
    cout << "Please enter a non-negative integer." << endl;
  } else {
      int sum = sumOfNaturalNumbers(num);
      cout << "The sum of natural numbers up to " << num << " is: " << sum << endl;
  }

  return 0;
}
```

Output//



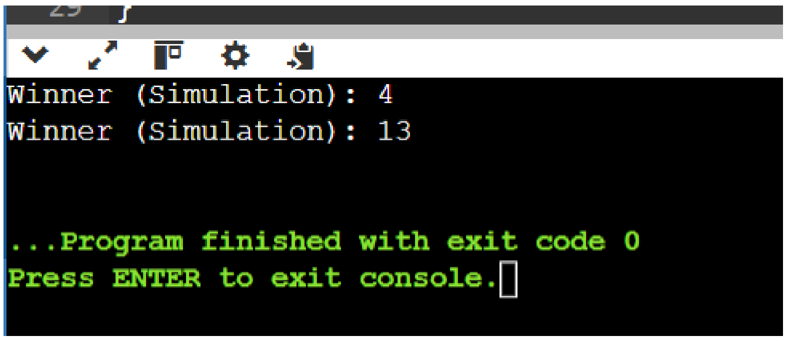
```
Enter a positive integer: 6
The sum of natural numbers up to 6 is: 21


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

Ques 6: find the winner of circular game.

Code//

```
#include <iostream>
#include <vector>

using namespace std;

int josephus_simulation(int n, int k) {
    vector<int> people;
    for (int i = 1; i <= n; ++i) {
        people.push_back(i);
    }
    int current = 0;
    while (people.size() > 1) {
        current = (current + k - 1) % people.size();
        people.erase(people.begin() + current);
    }
    return people[0];
}
int main() {
    int n = 7;
    int k = 3;
    cout << "Winner (Simulation): " << josephus_simulation(n, k) << endl; // Output: 4
    n = 14;
    k = 2;
    cout << "Winner (Simulation): " << josephus_simulation(n, k) << endl; // Output: 13
    return 0;
}
```

Code//



```
Winner (Simulation): 4
Winner (Simulation): 13


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

Ques 7: write a c++ program to check if no. is palindrome or not using function.

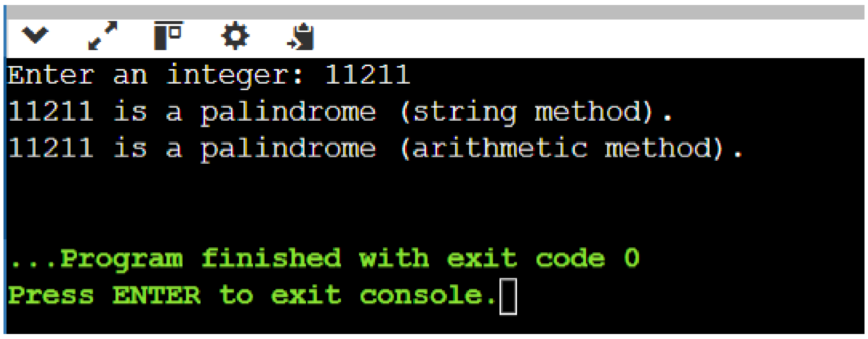
Code//

```
#include <iostream>
#include <string>
```

```
#include <algorithm>

using namespace std;

bool isPalindrome(int num) {
   string numStr = to_string(num);
   string reversedStr = numStr;
   reverse(reversedStr.begin(), reversedStr.end());
   return numStr == reversedStr;
}
bool isPalindromeArithmetic(int num) {
   if (num < 0) {
      return false;
   }
   int originalNum = num;
   int reversedNum = 0;
   while (num > 0) {
      int lastDigit = num % 10;
      reversedNum = reversedNum * 10 + lastDigit;
      num /= 10;
   }
   return originalNum == reversedNum;
}
int main() {
   int num;
   cout << "Enter an integer: ";
   cin >> num;
   if (isPalindrome(num)) {
      cout << num << " is a palindrome (string method)." << endl;
```

```
    } else {
        cout << num << " is not a palindrome (string method)." << endl;
    }
    if (isPalindromeArithmetic(num)) {
        cout << num << " is a palindrome (arithmetic method)." << endl;
    } else {
        cout << num << " is not a palindrome (arithmetic method)." << endl;
    }

    return 0;
}
```

Output//



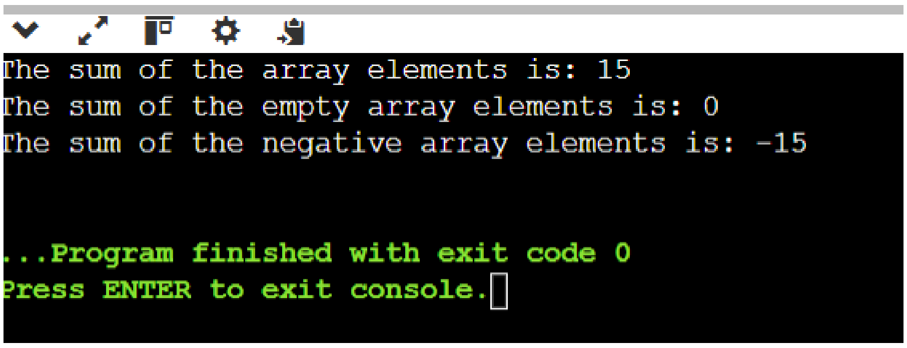
```
Enter an integer: 11211
11211 is a palindrome (string method).
11211 is a palindrome (arithmetic method).


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

Ques 8: Sum of array element using recursion.

CODE//

```
#include <iostream>
#include <vector>


using namespace std;
int sumOfArray(const vector<int>& arr, int index) {
    if (index >= arr.size() || index < 0) {
        return 0;
    } else {


        return arr[index] + sumOfArray(arr, index + 1);
    }
}

int main() {
    vector<int> numbers = {1, 2, 3, 4, 5};
    int sum = sumOfArray(numbers, 0);

    cout << "The sum of the array elements is: " << sum << endl;

    vector<int> emptyArray;
    sum = sumOfArray(emptyArray, 0);
    cout << "The sum of the empty array elements is: " << sum << endl;

    vector<int> numbers2 = {-1, -2, -3, -4, -5};
    sum = sumOfArray(numbers2, 0);
    cout << "The sum of the negative array elements is: " << sum << endl;

    return 0;
}
```

Code//



```
The sum of the array elements is: 15
The sum of the empty array elements is: 0
The sum of the negative array elements is: -15


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

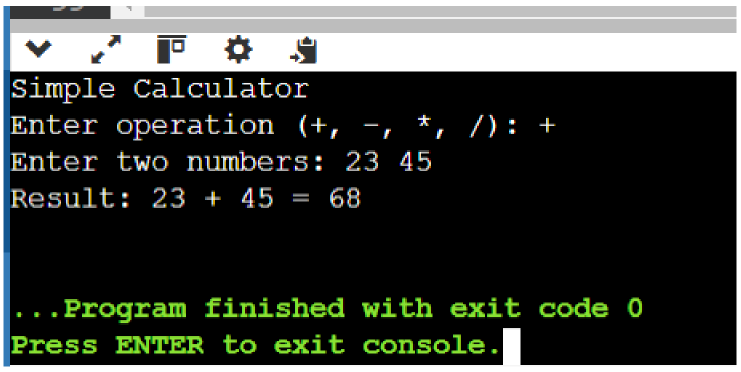
Ques 9: write a c++ program to create a simpler calculator that perform basic airthmatic function that performs add,multiply,sub and division.

Code//

```
#include <iostream>
```

```
#include <limits> // Required for numeric_limits

using namespace std;

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

    cout << "Simple Calculator" << endl;
    cout << "Enter operation (+, -, *, /): ";
    cin >> operation;

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

    // Input validation for division by zero
    if (operation == '/' && num2 == 0) {
        cerr << "Error: Division by zero is not allowed." << endl;
        return 1; // Indicate an error
    }

    // Input validation to handle non-numeric input.
    if (cin.fail()) {
        cerr << "Error: Invalid Input. Please enter numbers only." << endl;
        cin.clear(); // clears the error flags
        cin.ignore(numeric_limits<streamsize>::max(), '\n'); // discards the invalid input from
the input buffer.
        return 1;
    }
    double result;

    switch (operation) {
        case '+':
            result = num1 + num2;
            break;
        case '-':
            result = num1 - num2;
            break;
        case '*':
            result = num1 * num2;
            break;
        case '/':
            result = num1 / num2;
            break;
        default:
            cerr << "Error: Invalid operation." << endl;
            return 1; // Indicate an error
    }

    cout << "Result: " << num1 << " " << operation << " " << num2 << " = " << result << endl;

    return 0; // Indicate successful execution
}
```

Output//



```
Simple Calculator
Enter operation (+, -, *, /): +
Enter two numbers: 23 45
Result: 23 + 45 = 68


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