Name: Harsh Raj Choudhary

UID: 22BCS11231

Sec/Group = KPIT 901-B

Problem 1:

```cpp
1  #include <iostream>
2  using namespace std;
3
4  // Function to calculate Fibonacci number using recursion
5  int fibonacci(int n) {
6      if (n == 0) {
7          return 0; // Base case: F(0) = 0
8      } else if (n == 1) {
9          return 1; // Base case: F(1) = 1
10     } else {
11         return fibonacci(n - 1) + fibonacci(n - 2); // Recursive case
12     }
13 }
14
15 int main() {
16     int n;
17     cout << "Enter the value of n: ";
18     cin >> n;
19
20     // Validate input
21     if (n < 0 || n > 30) {
22         cout << "Invalid input. Please enter a value between 0 and 30.\n";
```

input
```
Enter the value of n: 12
F(12) = 144

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

Problem 2:

```cpp
1   #include <iostream>
2   using namespace std;
3
4   // Definition for singly-linked List.
5   struct ListNode {
6       int val;
7       ListNode* next;
8
9       ListNode(int x) : val(x), next(nullptr) {}
10  };
11
12  // Function to reverse the Linked List iteratively
13  ListNode* reverseListIterative(ListNode* head) {
14      ListNode* prev = nullptr;
15      ListNode* curr = head;
16      while (curr != nullptr) {
17          ListNode* nextTemp = curr->next; // Save the next node
18          curr->next = prev;                // Reverse the link
19          prev = curr;                      // Move prev forward
20          curr = nextTemp;                  // Move curr forward
21      }
22      return prev;
23  }
24
25  // Function to reverse the Linked List recursively
26  ListNode* reverseListRecursive(ListNode* head) {
27      if (head == nullptr || head->next == nullptr) {
28          return head; // Base case: empty list or single node
```

input

```
Original List: 1 2 3 4 5
Reversed List (Iterative): 5 4 3 2 1
Original List: 1 2
Reversed List (Recursive): 2 1
```

Problem 3:

```cpp
1   #include <iostream>
2   using namespace std;
3
4   // Definition for singly-linked List.
5   struct ListNode {
6       int val;
7       ListNode* next;
8       ListNode(int x) : val(x), next(nullptr) {}
9   };
10
11  // Function to add two numbers represented by Linked lists
12  ListNode* addTwoNumbers(ListNode* l1, ListNode* l2) {
13      ListNode* dummyHead = new ListNode(0); // Dummy node to start the result list
14      ListNode* current = dummyHead;         // Pointer to traverse the result list
15      int carry = 0;                         // Initialize carry to 0
16
17      // Traverse both lists until both are null
18      while (l1 != nullptr || l2 != nullptr || carry != 0) {
19          int sum = carry; // Start with the carry
20          if (l1 != nullptr) {
21              sum += l1->val;
22              l1 = l1->next;
23          }
24          if (l2 != nullptr) {
25              sum += l2->val;
26              l2 = l2->next;
27          }
28          carry = sum / 10;                  // Calculate carry
```

input

```
Input l2: 0
Output: 0
Input l1: 9 -> 9 -> 9 -> 9 -> 9 -> 9 -> 9
Input l2: 9 -> 9 -> 9 -> 9
Output: 8 -> 9 -> 9 -> 9 -> 0 -> 0 -> 0 -> 1
```

Problem 4:

```cpp
 1  #include <iostream>
 2  #include <vector>
 3  #include <string>
 4  using namespace std;
 5
 6  // Function to implement wildcard matching
 7  bool isMatch(string s, string p) {
 8      int m = s.size(), n = p.size();
 9      vector<vector<bool>> dp(m + 1, vector<bool>(n + 1, false));
10
11      // Base case: Empty string matches with empty pattern
12      dp[0][0] = true;
13
14      // Handle patterns with '*'
15      for (int j = 1; j <= n; ++j) {
16          if (p[j - 1] == '*') {
17              dp[0][j] = dp[0][j - 1];
18          }
19      }
20
21      // Fill the DP table
22      for (int i = 1; i <= m; ++i) {
23          for (int j = 1; j <= n; ++j) {
24              if (p[j - 1] == s[i - 1] || p[j - 1] == '?') {
25                  dp[i][j] = dp[i - 1][j - 1]; // Characters match or '?' matches any sir
26              } else if (p[j - 1] == '*') {
27                  dp[i][j] = dp[i - 1][j] || dp[i][j - 1]; // '*' matches zero or more ch
28
```

input

```
Input: s = "aa", p = "a"
Output: false
Input: s = "aa", p = "*"
Output: true
Input: s = "cb", p = "?a"
Output: false
```

Problem 5:

```cpp
 3  #include <algorithm>
 4  using namespace std;
 5
 6  // Function to calculate GCD using the Euclidean algorithm
 7  int findGCD(int a, int b) {
 8      while (b != 0) {
 9          int temp = b;
10          b = a % b;
11          a = temp;
12      }
13      return a;
14  }
15
16  // Main function to find the GCD of the smallest and largest numbers in the array
17  int gcdOfArray(const vector<int>& nums) {
18      int smallest = *min_element(nums.begin(), nums.end());
19      int largest = *max_element(nums.begin(), nums.end());
20      return findGCD(smallest, largest);
21  }
22
23  int main() {
24      // Example 1
25      vector<int> nums1 = {2, 5, 6, 9, 10};
26      cout << "Input: nums = [2, 5, 6, 9, 10]" << endl;
27      cout << "Output: " << gcdOfArray(nums1) << endl;
28
29      // Example 2
30      vector<int> nums2 = {7, 5, 6, 8, 3};
```

input

```
Input: nums = [2, 5, 6, 9, 10]
Output: 2
Input: nums = [7, 5, 6, 8, 3]
Output: 1
Input: nums = [3, 3]
Output: 3
```