

Kaushik Dhruv Alok

22BCS10210

KPIT-901/A

Day 3 Questions

Q1. Fibonacci Series Using Recursion

The Fibonacci numbers, commonly denoted $F(n)$ form a sequence, called the Fibonacci sequence, such that each number is the sum of the two preceding ones, starting from 0 and 1.

Q2. Reverse Linked List

Given the head of a singly linked list, reverse the list, and return the reversed list.

Q3. Add Two Numbers

You are given two non-empty linked lists representing two non-negative integers. The digits are stored in reverse order, and each of their nodes contains a single digit. Add the two numbers and return the sum as a linked list.

Q4. Wildcard Matching

Given an input string (s) and a pattern (p), implement wildcard pattern matching with support for '?' and '*' where:

- '?' Matches any single character.
- '*' Matches any sequence of characters (including the empty sequence).

Q5. Special Binary String

Special binary strings are binary strings with the following two properties:

- The number of 0's is equal to the number of 1's.
- Every prefix of the binary string has at least as many 1's as 0's.

Code:

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <string>

using namespace std;

int fib(int n){
    return (n<2) ? n : fib(n-1) + fib(n-2);
}

struct ListNode {
    int val;
```

```

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

```

```

ListNode* reverseList(ListNode* head){
    if(!head || !head->next) return head;
    ListNode* newHead = reverseList(head->next);
    head->next->next = head;
    head->next = nullptr;
    return newHead;
}

```

```

ListNode* addTwoNumbersRecursive(ListNode* l1, ListNode* l2, int carry){
    if(!l1 && !l2 && carry==0) return nullptr;
    int sum = (l1? l1->val : 0) + (l2? l2->val : 0) + carry;
    ListNode* node = new ListNode(sum % 10);
    node->next = addTwoNumbersRecursive(l1? l1->next : nullptr,
                                       l2? l2->next : nullptr,
                                       sum / 10);
    return node;
}

```

```

ListNode* addTwoNumbers(ListNode* l1, ListNode* l2){
    return addTwoNumbersRecursive(l1, l2, 0);
}

```

```

bool helper(const string &s, const string &p, int i, int j, vector<vector<int>> &memo){
    if(memo[i][j] != -1) return memo[i][j];
    if(j == (int)p.size()) return memo[i][j] = (i == (int)s.size());
    if(i == (int)s.size()){
        while(j < (int)p.size() && p[j] == '*') j++;
        return memo[i][j] = (j == (int)p.size());
    }
    if(p[j] == s[i] || p[j] == '?'){
        memo[i][j] = helper(s, p, i+1, j+1, memo);
        return memo[i][j];
    }
    if(p[j] == '*'){
        memo[i][j] = helper(s, p, i, j+1, memo) || helper(s, p, i+1, j, memo);
        return memo[i][j];
    }
    return memo[i][j] = false;
}

```

```

bool isMatch(string s, string p){
    vector<vector<int>> memo(s.size()+1, vector<int>(p.size()+1, -1));
    return helper(s, p, 0, 0, memo);
}

```

```

string makeLargestSpecial(string s){
    if(s.size() < 2) return s;
    int count = 0, start = 0;
    vector<string> subs;

```

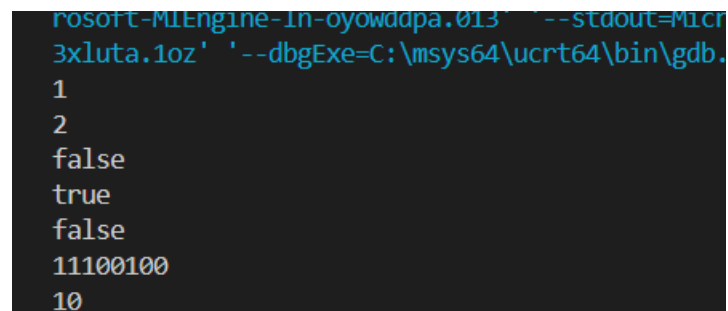
```

for(int i=0; i<(int)s.size(); i++){
    (s[i] == '1') ? count++ : count--;
    if(count == 0){
        subs.push_back("1" + makeLargestSpecial(s.substr(start+1, i - start - 1)) + "0");
        start = i + 1;
    }
}
sort(subs.begin(), subs.end(), greater<string>());
string result;
for(auto &sub : subs) result += sub;
return result;
}

int main(){
    cout << fib(2) << endl;
    cout << fib(3) << endl;
    cout << boolalpha << isMatch("aa", "a") << endl;
    cout << isMatch("aa", "*") << endl;
    cout << isMatch("cb", "?a") << endl;
    cout << makeLargestSpecial("11011000") << endl;
    cout << makeLargestSpecial("10") << endl;
    return 0;
}

```

Output:



```

Microsoft-ML-Engine-In-oyowddpa.013 --stdout=Micro
3x1uta.1oz ' ' --dbgExe=C:\msys64\ucrt64\bin\gdb.
1
2
false
true
false
11100100
10

```

Q6. Write a Function to print first name and last name using function

You are given the firstname and lastname of a person on two different lines. Your task is to read them and print them by using a function

Q7. Find GCD of Number Using Function

Given an integer array nums, return the greatest common divisor of the smallest number and largest number in nums.

The greatest common divisor of two numbers is the largest positive integer that evenly divides both numbers.

Q8. Longest Substring Without Repeating Characters

Write a function that takes a string as input and returns the length of the longest substring without repeating characters. A substring is a contiguous sequence of characters within the string.

Q9. Maximum Subarray Product

Write a function that takes an integer array as input and returns the maximum product of a contiguous subarray. The array can contain both positive and negative integers, and your function must account for these to find the optimal subarray.

Q10. There are n children standing in a line. Each child is assigned a rating value given in the integer array ratings.

- You are giving candies to these children subjected to the following requirements:
- Each child must have at least one candy.
- Children with a higher rating get more candies than their neighbors.

Return the minimum number of candies you need to have to distribute the candies to the children.

Code:

```
#include <iostream>
#include <stdlib.h>
#include <algorithm>
#include <vector>

using namespace std;

void print_full_name(string first, string last){
    cout<<"Hello "<<first<<" "<<last<<"! You just delved into function"<<endl;
}

int gcdF(int a,int b){
    return b==0?a:gcdF(b,a%b);
}

int findGCDofMinMax(vector<int>& nums){
    int mn=*min_element(nums.begin(),nums.end());
    int mx=*max_element(nums.begin(),nums.end());
    return gcdF(mn,mx);
}

int lengthOfLongestSubstring(string s){
    vector<int> idx(256, -1);
    int res=0, start=0;
    for(int i=0;i<s.size();i++){
        if(idx[s[i]]>=start) start=idx[s[i]]+1;
        idx[s[i]]=i;
        res=max(res, i-start+1);
    }
    return res;
}

int maxProduct(vector<int>& nums){
    int ans=nums[0], mx=nums[0], mn=nums[0];
    for(int i=1;i<nums.size();i++){
```

```

        if(nums[i]<0) swap(mx,mn);
        mx=max(nums[i], mx*nums[i]);
        mn=min(nums[i], mn*nums[i]);
        ans=max(ans, mx);
    }
    return ans;
}

int candy(vector<int>& ratings){
    int n=ratings.size();
    vector<int> left(n,1), right(n,1);
    for(int i=1;i<n;i++){
        if(ratings[i]>ratings[i-1]) left[i]=left[i-1]+1;
    }
    for(int i=n-2;i>=0;i--){
        if(ratings[i]>ratings[i+1]) right[i]=right[i+1]+1;
    }
    int ans=0;
    for(int i=0;i<n;i++){
        ans+=max(left[i], right[i]);
    }
    return ans;
}

int main(){
    {
        string f="function", l="example";
        print_full_name(f,l);
    }
    {
        vector<int> nums1={2,5,6,9,10};
        cout<<findGCDofMinMax(nums1)<<endl;
        vector<int> nums2={7,5,6,8,3};
        cout<<findGCDofMinMax(nums2)<<endl;
        vector<int> nums3={3,3};
        cout<<findGCDofMinMax(nums3)<<endl;
    }
    {
        string s="abcabcbb";
        cout<<lengthOfLongestSubstring(s)<<endl;
    }
    {
        vector<int> nums={2,3,-2,4};
        cout<<maxProduct(nums)<<endl;
    }
    {
        vector<int> ratings1={1,0,2};
        cout<<candy(ratings1)<<endl;
        vector<int> ratings2={1,2,2};
        cout<<candy(ratings2)<<endl;
    }
    return 0;
}

```

Output:

```
Hello function example! You just delved into function
2
1
3
3
6
5
4
```