Answer Script

Question No. 01	
Write C++ program to solve The Lakes by using dfs.	10

Answer No. 01

```
#include<bits/stdc++.h>
using namespace std;

const int N = 1e3+5;
int p[N][N], n, m, sum;
int dx[]= {0,0,1,-1};
int dy[]= {1,-1,0,0};

int dfs(int x,int y)
{
        if(!p[x][y])
        return 0;
        sum+=p[x][y];
        p[x][y]=0;
        for(int i=0; i<4; i++)
        {
        int wx=x+dx[i];
        int wy=y+dy[i];
        if(wx>0&&wx<=n&&wy>0&&wy<=m&&p[wx][wy])
        dfs(wx,wy);
        }
        return sum;
}
int main()
{
        ios_base::sync_with_stdio(0);
        cin.tie(0);
        int t;
        cin>>t;
        while(t--)
        {
```

```
	int dd=0;
	cin>>n>>m;
	for(int i=1; i<=n; i++)
	for(int j=1; j<=m; j++)
		cin>>p[i][j];
	for(int i=1; i<=n; i++)
	for(int j=1; j<=m; j++)
	{
		sum=0;
		dd=max(dfs(i,j),dd);
	}
	cout<<dd<<"\n";
	}
	return 0;
}
```

Question No. 02	
Write C++ program to solve Network Delay Time by using dijkstra.	10

Answer No. 02

```
#include<bits/stdc++.h>
using namespace std;

class Solution
{
 public:
  int networkDelayTime(vector<vector<int>>& times, int N, int K) {
        unordered_map<int, unordered_map<int, int>> edges;

        for (auto t : times) {
        edges[t[0]][t[1]] = t[2];
        }

        unordered_set<int> visited;

        auto cmp = [](pair<int, int> left, pair<int, int> right) {
        return left.first > right.first;
        };
        priority_queue<pair<int, int>, vector<pair<int, int>>, decltype(cmp)> next(
        cmp);
        next.push(make_pair(0, K));

        while (!next.empty()) {
        int t = next.top().first;
        int n = next.top().second;
        next.pop();

        if (visited.find(n) != visited.end()) continue;

        visited.insert(n);
        if (visited.size() == N) return t;

        for (auto it : edges[n]) {
        next.push(make_pair(it.second + t, it.first));
```

```
        }
        }
        return -1;
  }
};

int main()
{
  int n, m, k;
  cout << "Nodes: ";
  cin >> n;
  cout << "Source Node: ";
  cin >> k;
  cout << "Number of edges: ";
  cin >> m;
  cout << "Times: \n";

  vector<vector<int>> times(m, vector<int>(3));
  for (int i = 0; i < m; i++) {
        cin >> times[i][0] >> times[i][1] >> times[i][2];
  }

  Solution s;
  int result = s.networkDelayTime(times, n, k);

  cout << result << "\n";

  return 0;
}
```

Question No. 03
Write C++ program to solve Coin Combinations I by using the tabulation method. 10
Answer No. 03

```
#include <bits/stdc++.h>
using namespace std;

int main()
{
  int mod = 1e9+7;
  int n, target;
  cin >> n >> target;
  vector<int> c(n);
  for (int&v : c) cin >> v;

  vector<int> dp(target+1,0);
  dp[0] = 1;
  for (int i = 1; i <= target; i++) {
          for (int j = 0; j < n; j++) {
          if (i-c[j] >= 0) {
          dp[i] = (dp[i] + dp[i-c[j]]) % mod;
          }
          }
  }
  cout << dp[target] << "\n";

  return 0;
}
```

Question No. 04	
Write C++ program to solve Sum of Two Values	10

Answer No. 04

```
#include<bits/stdc++.h>
using namespace std;
const int maxN = 2e5+1;

int N, x, a;
pair<int, int> p[maxN];

int find(int val)
{
        int l = 1, r = N;
        while (l <= r) {
        int m = l + (r - l) / 2;
        if (p[m].first == val) {
        return p[m].second;
        }
        else if (p[m].first < val) {
        l = m + 1;
        }
        else {
        r = m - 1;
        }
        }
        return 0;
}

int main()
{
        cin >> N >> x;
        for (int i = 1; i <= N; i++) {
        cin >> a;
        p[i] = make_pair(a, i);
        }
        sort(p+1, p+N+1);
```

```
        for (int i = 1; i <= N; i++) {
        int other = find(x - p[i].first);
        if (other != 0 && other != p[i].second) {
        cout << p[i].second << " " << other << '\n';
        return 0;
        }
        }
        cout << "IMPOSSIBLE\n";
        return 0;
}
```

Question No. 05

Write C++ program to solve Money Sums by using the tabulation method.

15

Answer No. 05

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

const int maxX = 1e5;

int N, x, cnt, largest;
bool dp[maxX+1];

int main() {
        cin >> N;

        dp[0] = true;
        for (int i = 0; i < N; i++) {
        cin >> x;
        for (int j = maxX - x; j >= 0; j--) {
        if (dp[j]) {
                dp[j + x] = true;
        }
        }
```

```
        }
        for (int i = 1; i <= maxX; i++) {
        if (dp[i]) {
        largest = i;
        cnt++;
        }
        }
        cout << cnt << "\n";
        for (int i = 1; i <= maxX; i++) {
        if (dp[i]) {
        cout << i << ((i == largest) ? '\n' : ' ');
        }
        }
        return 0;
}
```

Question No. 06	
Write C++ program to solve Back to Underworld	15

Answer No. 06

```
  #include<bits/stdc++.h>
using namespace std;

const int SIZE = 20005;
const int MOD = 20071027;

list<int> adj[SIZE];
int color[SIZE];

enum {NOT_VISITED, BLACK, RED};

int main()
{
        int tc, t = 0, i, j, k, m, n, mx = 0, x, y, q, value, node;
        char ch;
        long long sum = 0;
```

```
	cin >> tc;

	for (t = 1; t <= tc; t++)
	{
	cin >> n;

	memset(color, 0, sizeof color);
	for (i = 0; i < SIZE; i++)
	{
	adj[i].clear();
	}

	for (i = 0; i < n; i++)
	{
	cin >> x >> y;
	adj[x].push_back(y);
	adj[y].push_back(x);
	}

	mx = 0;

	for (i = 0; i < SIZE; i++)
	{
	if (!adj[i].empty() && color[i] == NOT_VISITED)
	{
		int black = 0, red = 0;
		queue<int> q;
		q.push(i);
		color[i] = BLACK;
		black++;

		while (!q.empty())
		{
		node = q.front();
		q.pop();

		for (list<int>::iterator it = adj[node].begin(); it != adj[node].end();
it++)
		{
		if (color[*it] == NOT_VISITED)
```

```
                {
                        q.push(*it);
                        if (color[node] == BLACK)
                        {
                        color[*it] = RED;
                        red++;
                        }
                        else
                        {
                        color[*it] = BLACK;
                        black++;
                        }
                }
                }
                }
                mx += max(red, black);
        }
        }
        cout << "Case " << t << ": " << mx << "\n";
        }
        return 0;
}
```

Question No. 07	
Write C++ program to solve Required Length.	15

Answer No. 07

```
#include<bits/stdc++.h>
using namespace std;

long long n, x;
map<long long, long long> m;

set<long long> digits(long long x)
{
	set<long long> ret;

	while(x)
	{
	ret.insert(x%10);
	x /= 10;
	}
	return ret;
}
long long length(long long x)
{
	int ret = 0;
	while(x)
	{
	ret++;
	x /= 10;
	}
	return ret;
}
long long dfs(long long x)
{
	int l = length(x);
	if(l == n)  return 0;
	if(l > n )  return 1e9;
	if(m.count(x))  return m[x];
```

```
        long long mini = 1e9;

        for(long long d: digits(x))
        {
        if(d==1 || d==0)  continue;
        mini = min(1 + dfs(x*d), mini);
        }
        return m[x] = mini;
}

int main()
{
        cin>> n >> x;
        long long ans = dfs(x);
        if(ans == 1e9)
        {
        cout<<"-1"<< "\n";
        }
        else
        {
        cout<<ans<< "\n";
        }
        return 0;
}
```

Question No. 08	
Write C++ program to solve LCS by using the memoization method.	15

Answer No. 08

```
#include<bits/stdc++.h>
using namespace std;

int main()
{
        string s, t;
```

```
	cin >> s >> t;
	int n = s.size(), m = t.size();
	vector<vector<int>> dp(n + 1, vector<int>(m + 1, 0));
	for (int i = 1; i <= n; i++)
	{
	for (int j = 1; j <= m; j++)
	{
	dp[i][j] = max(dp[i - 1][j], dp[i][j - 1]);
	if (s[i - 1] == t[j - 1])
		dp[i][j] = max(dp[i][j], dp[i - 1][j - 1] + 1);
	}
	}
	string ans;
	int i = n, j = m;
	while (i && j)
	{
	if (s[i - 1] == t[j - 1])
	{
	ans += s[i - 1];
	i--, j--;
	}
	else if (dp[i][j - 1] >= dp[i - 1][j])
	{
	j--;
	}
	else
	{
	i--;
	}
	}
	reverse(ans.begin(), ans.end());
	cout << ans << "\n";

	return 0;
}
```