Answer Script

Question No. 01
Fibonacci Number
Answer No. 01

Memoization Method :

```
#include<bits/stdc++.h>
using namespace std;
const int N = 30+1;
int dp[N];

int fib(int n){
   if(n == 0)return 0;
   if(n == 1) return 1;
   if(dp[n] == -1){
      dp[n] = fib(n-1) + fib(n-2);
   }
   return dp[n];
}

int main(){
   int n;cin>>n;
   memset(dp, -1, sizeof(dp));
   cout<<fib(n)<<endl;
   return 0;
}
```

Tabulation Method :

```
#include<bits/stdc++.h>
using namespace std;
const int N = 33;
int dp[N];

int main()
{
   int n;cin >> n;
```

```
    dp[0] = 0;
    dp[1] = 1;
    for(int i = 2; i<=n; i++){
        dp[i] = dp[i-1] + dp[i-2];
    }

    cout <<dp[n]<< endl;

    return 0;
}
```

Question No. 02

FARIDA

Answer No. 02

Memoization Method :

```
#include<bits/stdc++.h>
using namespace std;
const int N = 1e5+5;
int dp[N] ;

int farida(int i) {
    if(i==0)return 0;
    if(i==1)return dp[1];
    return farida(i-2)+i*dp[i];
}

int main(){

    int t, d;cin>>t;d=t;
    while(t--)
    {
        int n;cin>>n;
        int maxi = -1;
        for(int k = 0 ; k < n ; k++){
            int l;cin>>l;
            dp[l]++;
```

```
        maxi=max(maxi,l);
      }

      cout<<"Case "<<d-t<<": "<< farida(maxi)<<endl;
      memset(dp,0,sizeof(dp));
   }

   return 0;
}
```

Tabulation Method :

```
#include<bits/stdc++.h>
using namespace std;
const int N = 1e5+5;
int dp[N],cnt[N] ;

int main(){

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

      if(n==0){
         cout<<"Case "<<i<<": "<<0<<endl;
         continue;
      }

      for(int k=0; k<n; k++) {
         cin >> cnt[k];
      }

      dp[0]=cnt[0];
      dp[1]=max(dp[0],cnt[1]);
      for(int k=2;k<n;k++){
      dp[k]= max(dp[k-1],dp[k-2]+cnt[k]);
      }

      cout<<"Case "<<i<<": "<<dp[n-1]<<endl;
   }
```

```
    return 0;
}
```

Question No. 03

Boredom

Answer No. 03

Memoization Method :

```
#include<bits/stdc++.h>
using namespace std;
const int N = 1e5+5;
int dp[N];

int Alex(const int i) {
    if(i==0)return 0;
    if(i==1)return dp[1];
    return max(Alex(i-1),Alex(i-2)+i*dp[i]);
}

int main()
{
    int maxi = -1;
    int n;cin>>n;
    for(int k=0; k<n; k++) {
        int l;cin>>l;
        dp[l]++;
        maxi=max(maxi,l);
    }

    cout<<Alex(maxi)<<endl;
    return 0;
}
```

Tabulation Method :

```
#include<bits/stdc++.h>
```

```
using namespace std;
const int N = 1e5+5;
int dp[N], dp2[N];

int main(){

   int n; cin>>n;
   for(int k=0; k<n; k++) {
      int l;cin>>l;
      dp[l]++;
   }

   dp2[0]=0;
   dp2[1]=dp[1];

   for(int k=2; k<=n; k++){
      dp2[k]= max(dp2[k-1],dp2[k-2]+k*dp[k]);
   }

   cout<<dp2[n]<<endl;
   return 0;
}
```

Question No. 04

N-th Tribonacci Number

Answer No. 04

Memoization Method :

```
#include<bits/stdc++.h>
using namespace std;
const int N = 40;
int dp[N];

int fibo(int n){
   if(n==0) return 0;
   if(n==1 || n==2) return 1;
   if(dp[n] != -1){
      return dp[n];
```

```
    }

    int ans = fibo(n-1) + fibo(n-2)+fibo(n-3);
    dp[n] = ans;
    return ans;
}

int main()
{
    int n;cin >> n;
    for(int i = 1 ; i <= n ; i++) {
        dp[i] = -1;
    }

    cout<<fibo(n)<<endl;
    return 0;
}
```

Tabulation Method :

```
#include<bits/stdc++.h>
using namespace std;
const int N = 40;
int dp[N];

int main(){
    int n;cin>>n;

    dp[0] = 0;
    dp[1] = 1;
    dp[2] = 1;

    for(int i=3; i<=n; i++){
        dp[i] = dp[i-1] + dp[i-2]+dp[i-3];
    }

    cout<<dp[n]<<endl;
    return 0;
}
```

Question No. 05

You are given an integer n.You can perform any of the following operations on it as many times you want -

- Subtract 1 from it
- If it is divisible by 2 divide by 2
- If it is divisible by 3 divide by 3

Find the minimum number of operations to make n=1

Constraints -

1<=n<=10^5

Output -

Print a single integer,the minimum number of operations to make n=1

Sample Input-	Sample Output-
7	3
11	4

Explanation-

When n = 7,
By using 3 operations we can go from 7 to 1.
>> 1st step -> subtract 1 from 7 then it became 6
>> 2nd step -> 6 is divisible by 3 hence we can divide it by 3 and it became 2
>> 3rd step -> 2 is divisible by 2 hence we can divide it by 2 and it became 1

Answer No. 05

Memoization Method :

```
#include<bits/stdc++.h>
using namespace std;
const int N=1e5+5;
int dp[N];

int do_it(int n, int *dp){
   if(n==1)return 0;
   if(dp[n-1]!=-1) return dp[n-1];
```

```
    int two=INT_MAX,three=INT_MAX;
    int mino=1+do_it(n-1,dp);

    if(n%2==0) two=1+do_it(n/2,dp );

    if(n%3==0) three=1+do_it(n/3 ,dp);

    dp[n-1]=min({mino,two,three});
    return dp[n-1];
}

int helper(int n){
    for(int i=0;i<n;i++){
        dp[i]=-1;
    }
    return do_it(n,dp);
}
int main(){
    int n;cin>>n;
    cout<<helper(n );
    return 0;
}
```

Tabulation Method :

```
#include<bits/stdc++.h>
using namespace std;
const int N=100005;
int dp[N];

int main()
{
    int n;cin>>n;
    dp[1]=0;

    for(int i=2; i<=n; i++){
        if (!(i%2) && (i%3))
            dp[i] = 1+min(dp[i-1], dp[i/2]);
        else if (!(i%3) && (i%2))
            dp[i] = 1+min(dp[i-1], dp[i/3]);
```

```
        else if(!(i%2) && !(i%3))
            dp[i] = 1+min(dp[i-1],min(dp[i/2],dp[i/3]));
        else
            dp[i] =1+dp[i-1];
    }

    cout<<dp[n]<<endl;
    return 0;

}
```