سوال A: 1348A

- میدانیم که وزن سکه ی 2^n از مجموع وزن سکه های دیگر بیشتر است، بنابراین قسمتی که دو به توان ان را دارد، بی شک وزن بیشتری از قسمت دیگر دارد برای کمینه کردن این اختلاف n/2 -1 سکه ی کوچک را در بخشی که 2به توان n وجود دارد قرار میدهیم و بقیه ی سکه ها را در بخش دیگر. درواقع جواب مقدارِ عبارت زیر است:

$$(2^n + \sum_{i=1}^{n/2-1} 2^i) - \sum_{i=n/2}^{n-1} 2^i.$$

- با خلاصه کردن محاسبات بالا داریم: (2^(n/2+1)-2)

C++:

```cpp
#include <bits/stdc++.h>
using namespace std;
void solve() {
    int N;
    cin >> N;
    //note: 1<<X means 2^X
    //we put largest coin in first pile
    int sum1 = (1 << N), sum2 = 0;
    //we put n/2-1 smallest coins in first pile
    for (int i = 1; i < N / 2; i++)
        sum1 += (1 << i);
    //we put remaining n/2 coins in second pile
    for (int i = N / 2; i < N; i++)
        sum2 += (1 << i);
    cout << sum1 - sum2 << endl;
}
int main() {
    int t; cin >> t;
    while (t--)
        solve();
}
```

Python:

```python
for _ in range(int(input())):
        n = int(input())
        print(2**(n//2+1)-2)
```

سوال B: 1343A

❖ x+2x+4x+⋯+2^(k−1)x=n

اگر x را از معادله بالا فاکتور بگیریم معادله به فرم این میشود: x(1+2+4+.....) = n

❖ بنابر جمع جملات دنباله ی هندسی، حاصل پرانتز برابر است با 2^(k)-1 پس x برابر است با:

x=n/(2^k)-1 و به ازای همه ی kهای ممکن بررسی میکنیم که ایا n بر (2^k)-1 بخش پذیر است یا نه.

C++:

```cpp
#include <bits/stdc++.h>
using namespace std;
typedef unsigned long long ull;
int main(){
    int t;
    cin >> t;
    while (t--){
        ull n;
        cin >> n;
        for (int k = 2; k < 30; k++){
            ull p = pow(2, k);
            p -= 1;
            if (n % p == 0){
                cout << n / p << endl;
                break;
            }
        }
    }
    return 0;
}
```

Python:

```python
for _ in range(int(input())):
    n=int(input())
    a=3
    while(n%a!=0):
        a=(a<<1)+1
    print(int(n/a))
```

سوال C: 450A

- برای حل این سوال کافی است راه گفته شده در سوال را پیاده سازی کنید و یا مقدار ماکسمیم $ceil(a_i / m)$ را حساب کنید، اگر چند ماکسمیم وجود داشت، انکه اندیس (i) بزرگتری دارد جواب است.

C++:

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    double n, m, t;
    cin >> n >> m;
    double last = n;
    double max = INT_MIN;
    for (double i = 1; i <= n; i++) {
        cin >> t;
        if (ceil(t / m) >= max){
            max = ceil(t / m);
            last = i;
        }
    }
    cout << last << endl;
return 0;
}
```

Python:

```
import math
n, m = map(int, input().split())
l = list(map(int, input().split()))
for i in range(n):
        l[i] = math.ceil(l[i]/m)

    l.reverse()
    k = l.index(max(l))

    print(n-k)
```

سوال D: 102B

- برای حل این سوال کافیست برای عدد ورودی، گام های گفته شده در سوال را دنبال کنیم؛ یعنی تا زمان رسیدن به یک رقم، ارقام آن را جمع و با ان عدد جایگزین کنیم.

C++:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int k=0,c=0;
    string s;
    cin>>s;
    while(s.length()>1){
        k=0;
        for (int i=0;i<(int)s.length();i++)
            k+=(s[i]-48);
        s=to_string(k);
        c++;
    }
    cout<<c;
}
```

Python:

```
s = input()
c = 0
while len(s) > 1:
        s = str(sum(map(int, s)))
        c += 1
    print(c)
```

سوال E: 1332B

- میدانیم برای هر عدد مرکب مانند k عدد اولی مانند d وجود دارد به طوریکه : d<= sqrt(k)
- بنابراین برای ۱۰۰۰ حداکثر یازده عدد اول داریم.
- بنابراین میتوان نتیجه گرفت که هرعدد مرکبی که دربازه‌ی یک تا هزار داده شود حتما به یکی از یازده عدد اول بخش پذیر است و بنابر این برای رنگ امیزی کافی است به هر توپی، رنگی معادل با کوچکترین عدد اولی که به ان بخش پذیر است اختصاص دهیم.
- در این صورت توپ هایی همرنگ‌اند که در ب م م خود حداقل ان کوچکترین عدد اول را دارند و بنابراین ب م م انها بزرگتر از یک است.

C++:

```cpp
#include <bits/stdc++.h>
using namespace std;
typedef long long int ll;
int main(){
    int t, n;
    cin >> t;
    while (t--){
        cin >> n;
        ll arr[n], c = 1, k = 0;
        for (ll i = 0; i < n; i++)
            cin >> arr[i];
        int b[11] = {2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31};
        ll ans[n] = {0};
        for (ll i = 0; i < 11; i++){
            k = 0;
            for (ll j = 0; j < n; j++){
                if (ans[j] == 0 && arr[j] % b[i] == 0)
                    ans[j] = c, k = 1;
            }
            if (k == 1)
                c++;
        }
        cout << c - 1 << endl;
        for (ll i = 0; i < n; i++)
            cout << ans[i] << " ";
        cout << endl;
    }
    return 0;
}
```

Python:

```
primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31]
t = int(input())
for i in range(t):
     n = int(input())
    ls = [int(a) for a in input().split()]
    an = []
    for j in range(n):
        an.append(0)
    ctr = 1
        for p in range(len(primes)):
            nx = False
            for j in range(n):
                if ls[j] % primes[p] == 0 and an[j] == 0:
                    an[j]=ctr
                    nx=True
            if nx==True:
            ctr+=1
        s=''
        for a in an:
            s+=str(a)+' '
        print(ctr-1)
        print(s)
```