A Мишка и старший брат

На реализацию

Решение C++

```
#include <bits/stdc++.h>
using namespace std;
int main() {
	int a, b;
	cin >> a >> b;
	int answer = 0;
	while(a <= b) {
		a *= 3;
		b *= 2;
		++answer;
	}
	printf("%d\n", answer);
}
```

Решение Python

```
a,b = map(int,input().split())
an = 0
while a<=b:
 a *= 3
 b *= 2
 an += 1

print(an)
```

B ПокупатеЛи

Решение C++

```
#include <iostream>

using namespace std;

int main(){
    int t;
    cin >> t;
    while(t--){
        int n;
        cin >> n;
        if(n % 4 == 0){
            cout << "YES\n";
        }
        else cout << "NO\n";
    }
}
```

Решение Python

```python
t = int(input())
for testcase in range(t):
    n = int(input())
    if(n%4 == 0) :
        print("Yes")
    else :
        print("No")
```

C Замена элементов

Заметим, что так как все a_i — положительные, то для любой пары $a_i + a_j > \max(a_i, a_j)$. Это значит, что мы не можем сделать первый и второй минимумы меньше, чем они уже есть: предположим, первый и второй минимум — это mn_1 и mn_2. Если мы выберем заменить любой другой элемент, то мы не можем сделать его меньше, чем $mn_1 + mn_2$, а если выберем на замену mn_1 или mn_2, то только сделаем их больше.

В результате получается, что мы можем для каждого элемента либо не менять его, либо заменить его на $mn_1 + mn_2$. Таким образом, чтобы сделать все элементы $\leq d$, нам нужно проверить только, что либо $mn_1 + mn_2 \leq d$, либо максимальное $a_i \leq d$.

Мы можем это сделать, например, отсортировав наш массив a по возрастанию и проверив, что либо $a_1 + a_2 \leq d$, либо $a_n \leq d$.

Решение C++

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

using namespace std;

int main() {
  int tc;
  cin >> tc;
  while (tc--) {
    int n, d;
    cin >> n >> d;
    vector<int> a(n);
    for (int& x : a) cin >> x;
    sort(a.begin(), a.end());
    cout << (a.back() <= d || a[0] + a[1] <= d ? "YES" : "NO") << endl;
  }
}
```

Решение Python

```python
for _ in range(int(input())):
  n,d=map(int, input().split())
  l=list(map(int, input().split()))
  l.sort()
  if l[-1]<=d or l[0]+l[1]<=d:
    print("YES")
  else:
    print("NO")
```

D Влад и столовые

В этой задаче есть два решения.

1. Давайте в массив last для каждой столовой запишем момент её последнего вхождения, а затем найдём минимум в этом массиве. Номер, на котором достигается является номером искомой столовой.

2. Будем бежать по массиву с конца и для каждого номера столовой помнить, посещали мы его раньше или нет. Если очередной рассмотренный нами номер не встречался ранее, то он будет являться нашим новым кандидатом на ответ. В итоге последний кандидат и является ответом.

Решение C++

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

const int N = 2e5 + 10;

int n, a[N], vis[N], ans;

int main() {
    scanf("%d", &n);
    for (int i = 1; i <= n; ++i)
        scanf("%d", &a[i]);
    for (int i = n; i; --i) {
        if (vis[a[i]] == 0)
            ans = a[i];
        vis[a[i]] = 1;
    }
    printf("%d\n", ans);
    return 0;
}
```

Решение Python

```python
n = int(input())
a = list(map(int, input().split()))
a.reverse()
b = set()
ans = 0
for i in range(len(a)):
    if a[i] not in b:
        ans = a[i]
    b.add(a[i])
print(ans)
```

E Шерлок и его девушка

Подсказка: обратите внимание, что если i + 1 простое, то только те части, цена которых кратна i + 1, должны быть другого цвета.

Мы должны раскрасить ювелирные изделия таким образом, чтобы для каждого ювелирного изделия i, имеющего цену i + 1, все изделия, цены которых являются простыми делителями i + 1, должны были иметь цвет, отличный от цвета i-го изделия. Этого можно достичь, просто раскрасив все части с их ценами как простые числа в один цвет, а все остальные части во второй цвет. Мы вычисляем Решето Эратосфена до n ($\leq 10^5$) и, таким образом, получаем список простых и непростых чисел.

Решение C++

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

int sieve[100005];

int main()
{
	int i, n, j;
	cin>>n;
	for(i=2; i<=n+1; i++)
	{
		if(!sieve[i])
			for(j=2*i; j<=n+1; j+=i)
				sieve[j]=1;
	}

	if(n>2)
		cout<<"2\n";
	else
		cout<<"1\n";

	for(i=2; i<=n+1; i++)
	{
		if(!sieve[i])
			cout<<"1 ";
		else
			cout<<"2 ";
	}

	return 0;
}
```

Решение Python

```
n=int(input())
l=[1]*(n+2)
t=1
for i in range(2,n+2):
    if l[i]==1:
        for j in range(i+i,n+2,i):
            l[j]=2
            t=max(t,l[j])
print(t)
print(*l[2:])
```

F XOR или OR

Прежде всего убедитесь, что длины двух строк равны. Затем, немного попробовав и угадав, вы можете обнаружить, что нулевая строка (00 ... 0) не может быть преобразована ни во что другое, и ничто другое не может быть преобразовано в ноль. Все другие преобразования возможны.

Решение C++

```
#include <iostream>
#include <string>
using namespace std;
int main(){
        string a,b;
        cin>>a>>b;
        bool ans_a = false;
        bool ans_b = false;
        if(a.size()==b.size()){
                for(int i=0;i<a.size();i++){
                        if(a[i]=='1'){
                                ans_a=true;
                        }
                        if(b[i]=='1'){
                                ans_b=true;
                        }
                }
        }
        if(a.size()==1){
                ans_a=false;
        }
        if(a==b){
                ans_a=true;
                ans_b=true;
        }
        if(ans_a and ans_b)
                cout<<"YES"<<endl;
        else
                cout<<"NO"<<endl;
        return 0;
}
```

Решение Python

```
st=input()
st1=input()
if len(st)!=len(st1):
    print("NO")
elif(st.count('1')>0 and st1.count('1')>0 ):
    print("YES")
elif st.count('0')==st1.count('0'):
    print("YES")
else:
    print("NO")
```

G Минимальное число шагов

Последняя ситуация - это несколько символов «a» после нескольких символов «b».

Последняя ситуация уникальна.

Количество шагов тоже уникально.

Каждый символ «b» образует количество символов «b» в последней ситуации в соответствии с количеством символов «a» перед ним.

Решение C++

```
#include <stdio.h>
char s[1000001];
long long ans,a=1,b,m=1000000007;
int main(){
	scanf("%s",s);
	for(int i=0;s[i];++i){
		if(s[i]=='a'){
			a=(a<<1)%m;
		}else{
			ans=(ans+a)%m;
			++b;
		}
	}
	printf("%lld\n",(ans-b)%m);
	return 0;
}
```

Решение Python

```
s=input()
c=0
m=10**9+7
ka=0
for i in s:
    if(i=='b'):
        ka=(ka+c)%m
    else:
        c=(c*2+1)%m
print(ka)
```

H Слава и танки

Будем называть танки, которые изначально в четных позициях четными, а танки, которые изначально в нечетных позициях нечетными.

Бросим бомбы во все четные позиции. Теперь все танки в нечетных позициях. Бросим бомбы во все нечетные позиции. Теперь все четные танки уничтожены, а все нечетные находятся в четных клетках. Снова бросим бомбы во все четные позиции. Теперь все танки уничтожены.

Нетрудно доказать, что эта стратегия оптимальна.

Решение C++

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

using namespace std;

int main(){
	int m;
	while(cin>>m){
		cout<<m+m/2<<endl;
	for(int i = 2 ; i <= m ; i+=2){
		cout<<i<<" ";
	}
	for(int i = 1 ; i <= m ; i+=2){
		cout<<i<<" ";
	}
	for(int i = 2 ; i <= m ; i+=2){
		cout<<i<<" ";
	}
	cout<<endl;
	}

}
```

Решение Python

```
n=int(input())
print(n//2+n)
a=[int(i)for i in range(2,n+1,2)]
b=a+[int(i)for i in range(1,n+1,2)]+a
for i in b:
    print(i,end=' ')
```