

Bilye

SORU

Eski çağlarda Zekado ülkesinde şöyle bir oyun oynanırmış:

- Ortaya belli sayıda (n) bilye konur.
- İki kişi sıra ile hamleler yapar.
- Birinci oyuncu ilk hamlesinde ortadan 2 ya da 3 bilye alır.
- Sırası gelen oyuncu, bir önceki oyuncunun aldığı bilye sayısı k ise, $k+1 \leq m \leq 2k$ olacak şekilde m tane bilye alır. Son hamlede geriye kalan bilyelerin sayısı $\leq k$ ise bu bilyelerin hepsini alabilir.
- Son bilye grubunu alan oyunu kazanır.

Örnek bir oyun:

Ortada 25 tane bilye var:

	Aldığı Bilye Sayısı	Kalan Bilye Sayısı
1. Oyuncu	2	23
2. Oyuncu	3	20
1. Oyuncu	5	15
2. Oyuncu	8	7
1. Oyuncu	7	0

Bu durumda son hamleyi yapmış olan 1. oyuncu kazanıyor.

Sizden istenen ise bu oyunu hem 1. hem de 2. oyuncu için oynayan bir program yazmanız.

VARSAYIMLAR

- $2 \leq n \leq 100$
- Programınız standart girdiden (stdin) hamleleri alıp standart çıktıya (stdout) hamlelerini (aldığı bilye sayılarını) yazmalıdır.

GİRDİ - ÇIKTI

- Programınız oyun başlarken standart girdiden bilye sayısını (n) ve oyuncu numarasını belirten iki adet tamsayı okuyacaktır. Oyuncu numarasının değeri, birinci oyuncu iseniz 1, ikinci oyuncu iseniz 2 olacaktır.
- İlerleyen aşamalarda, hamle sırası karşıdaki oyuncuda ise onun yaptığı hamleyi okuyacak (1

adet tamsayı), sıra kendinde ise hamlesini yazacaktır (1 adet tamsayı).

DEĞERLENDİRME

- Verilen 'n' değerleri için, herhangi bir kod diğer bütün kodlarla hem 1. hem de 2. oyuncu için oyunu oynayacaktır.
- Herhangi bir anda yanlış bir hamle yapan oyuncu o oyunu kaybetmiş sayılacaktır.

ÇÖZÜM

Sorunun çözümü için gerekli yöntem Prof. Dr. Vasif Vagifoğlu'na ait **yapay zeka** adlı kitabın 252. sayfasında Sprague-Grundy Sayısı başlığı altında anlatılmıştır. Ayrıca *Bilye* oyununa çok benzeyen *Bergson Kibritleri* oyununun çözümü de kitabın 256. sayfasında yer almaktadır.

Çözüm için Sprague-Grundy sayılarını kullanacağız ve her bir durum için bir $SG(p,q) = t$ sayısı tanımlayacağız. Kazanan durumda $t=0$, aksi halde $t=1$ olduğunu kabul edelim. $SG(p,q)$ durumu; ortadan q tane bilye aldım, p tane bilye kaldı anlamına gelsin. Bu durumda ben kazanıyorsam $SG(p,q) = 0$, aksi halde $SG(p,q) = 1$ olsun.

Şimdi oyunun durumuna göre $SG(p,q)$ değerlerini oluşturmaya başlayalım. $SG(0,q)$ durumu her zaman 0'a(kazanan durum) eşittir; çünkü ben ortadan q tane bilye aldıktan sonra geriye 0 tane bilye kalıyorsa, oyunu ben kazanmışım demektir. Bunun tersi şekilde q tane bilye alıp geriye 1,2,3 ya da 4 bilye bırakırsam rakip kesin olarak kalan bilyeleri alıp oyunu kazanacaktır. Dikkat ederseniz oyunun ilk hamlesinde 2 ya da 3 bilye alınmaktadır. Başka bir deyişle sıra rakibe geldiğinde, rakip en kötü ihtimalle 4 bilye alma hakkına sahip olacaktır. Yani $SG(1,q)=1$, $SG(2,q)=1$, $SG(3,q)=1$, $SG(4,q)=1$.

Buraya kadar durumları el ile hesaplamak kolaydı; ancak bu noktadan sonraki durumları bilgisayara hesaplatmak zorundayız. Burada hesaplamayı kolaylaştıracak şu kuralı verebiliriz.

- $p \leq 2*q \Rightarrow SG(p,q) = 1$

Bu kuralı şu şekilde açıklayabiliriz. Biz ortadan q tane bilye aldığımızda rakip $2*q$ bilye alma hakkına sahip olacaktır. Eğer ortada kalan bilye sayısı $2*q$ 'ya eşit ya da daha azsa rakip bu bilyeleri alıp oyunu kazanacaktır. Ayrıca $q=1$ olan durumları ele almamıza gerek yoktur; çünkü oyunun kuralları gereği hiçbir oyuncu ortadan 1 bilye alma hakkına sahip değildir.

Bahsettiğimiz durumların haricindeki durumlara ait değerler ise şu şekilde hesaplanacaktır.


```

#include <stdio.h>
unsigned int bilyeSayisi, sonBilye, SG[101][101],
sira;
//-----
void setSG()
{
    unsigned int p,q,bilye;
    unsigned int kazanan;

    for(p=0; p < bilyeSayisi; ++p)
        for(q=0; q < bilyeSayisi; ++q)
            SG[p][q] = 1;

    for(q=0; q < bilyeSayisi; ++q)
        SG[0][q] = 0;

    for(p=5; p < bilyeSayisi; ++p)
        for(q=2; 2*q < p; ++q)
        {
            kazanan = 1;
            for(bilye=q+1; bilye<=2*q; ++bilye)
                if(SG[p-bilye][bilye] == 0)
                {
                    kazanan = 0;
                    break;
                }

            if(kazanan == 1)
                SG[p][q] = 0;
        }
}
//-----
unsigned int kacBilye(unsigned int min, unsigned
int max)
{
    unsigned int bilye;

    if(bilyeSayisi <= max)
        return bilyeSayisi;

    for(bilye=min; bilye<=max; bilye++)
        if(SG[bilyeSayisi-bilye][bilye] == 0)
            return bilye;

    return min;
}
//-----

```

```

void oyna()
{
    sonBilye = kacBilye(sonBilye+1,2*sonBilye);
    bilyeSayisi -= sonBilye;
    printf("\nCPU %d bilye aldı, %d bilye
kaldı\n",sonBilye,bilyeSayisi);

    if(!bilyeSayisi)
    {
        printf("\nCPU kazandı\n");
        return;
    }
}
//-----
int main()
{
    printf("\nBilye sayisini giriniz (n=?) => ");
    scanf("%d",&bilyeSayisi);

    setSG();
    printf("\nOyuncu numarasını giriniz => ");
    scanf("%d",&sira);
    if(sira == 1)
    {
        sonBilye = kacBilye(2,3);
        bilyeSayisi -= sonBilye;
        printf("\nCPU %d bilye aldı, %d bilye
kaldı",sonBilye,bilyeSayisi);
    }

    while( bilyeSayisi > 0)
    {
        printf("\nKaç bilye alacaksın => ");
        scanf("%d",&sonBilye);
        bilyeSayisi -= sonBilye;
        printf("\nSen %d bilye aldın, %d bilye
kaldı",sonBilye,bilyeSayisi);
        if(bilyeSayisi == 0)
        {
            printf("\nSen kazandın\n");
            return 0;
        }
        oyna();
    }

    return 0;
}

```