ساختمان داده ها

دانشگاه صنعتی نوشیروانی بابل

دکتر حسام عمران پور

طراحان اسلاید:
زهرا ریحانیان و دانیال علیزاده

حل تمرین :
علی باقری

لینک کانال تلگرام اطلاع رسانی و حل تمرین:
t.me/ds_nit_4011

فصل دوم
توابع بازگشتی و تحلیل مرتبه زمانی آن

استقرا رياضی

- پايه ي الگوريتم هاي بازگشتی را ، اصلی در رياضيات به نام اصل استقراي رياضی تشكيل می دهد.
- اين اصل، شيوه اي براي اثبات درستی قضاياي رياضی روي اعداد طبيعی است .در اين جا دو نوع استقراي رياضی مطرح می شود:

- 1.استقراي ساده ي رياضی
- 2.استقراي قوي رياضی

استقرای ساده ی ریاضی

پایه استقرا : در این بخش، درستی یک قضیه ی P برای عددی پایه (مثلاً عدد $n_0 = 1$) تحقیق میشود.

فرض استقرا : فرض میشود که قضیه ی P برای عدد صحیح دلخواه (n-1) برقرار است .

حکم : در این بخش، باید با استفاده از فرض استقرا، درستی P برای عدد $n' = n$ اثبات شود.

به عنوان یک مثال ساده، رابطه ی $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ با استفاده از اصل استقرای ریاضی اثبات میشود :

بخش اول – بررسی شرط پایه: باید درستی رابطه برای $n_0 = 1$ تحقیق گردد:

که درست است .

$$\sum_{i=1}^{1} i = \frac{1(1+1)}{2} = 1$$

بخش دوم – فرض استقرا: فرض می شود که رابطه ی $\sum_{i=1}^{n-1} i = \frac{(n-1)(n-1+1)}{2}$ برای $n' > n_0$ صحیح باشد .

بخش سوم- حکم استقرا: باید بتوان با استفاده از فرض بالا، درستی $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ را برای $n' = n$ اثبات کرده. بر اساس اطلاعات بالا :

$$\sum_{i=1}^{n} i = n + \sum_{i=1}^{n-1} i = n + \sum_{i=1}^{n'} i = n + \frac{n'(n'+1)}{2} = n + \frac{(n-1)n}{2} = \frac{n(n+1)}{2}$$

با توجه به این که درستی این رابطه برای شرط پایه و $n' = n$ اثبات شد، درستی این رابطه برای تمام اعداد طبیعی بزرگ تراز 1 (بزرگ تر از شرط پایه) نیز اثبات می شود.

توابع بازگشتی

تـابعی را کـه درقسمتی از تعریف آن، از خود تابع استفاده شده باشد، تابع بازگشتی می نامند .

شرط پایه : خروجی مناسب را برای $n = n_0$ تولید می نماییم .

فرض : فرض می کنیم اگر تابع با آرگومان $(n' = n - 1)$ $n' < n$ فراخوانی نماییم درست کار میکند و خروجی مناسب را تولید می نماید .

حکم : فراخوانی تابع با آرگومان $(n' = n - 1)$ $n' < n$ و چند دستور کاری می کنیم که تابع خروجی مناسب را برای بلاک n تولید نماید .

?? تابع بازگشتی برای تولید 1تا nبنویسید.

```
Print (int n){
    if(n==1){
        cout<<1 ;
    }
    else{
        print(n-1);
        cout<<n ;
    }
}
```

?? تابع بازگشتی برای تولید فاکتوریل بنویسید.

```
fact(n : integer) : integer
{
    if n == 0 then
        return 1;
    else
    {
        x = fact(n - 1);
        return (n * x);
    }
}
```

تابع بازگشتی برای فیبوناتچی بنویسید .

```
fibo(n : integer) : integer
{
    if n == 1 or n == 2 then
        return 1;
    else
    {
        x = fibo(n - 1);
        y = fibo(n - 2);
        return (x + y);
    }
}
```

برج هانوی Hanoi Tower

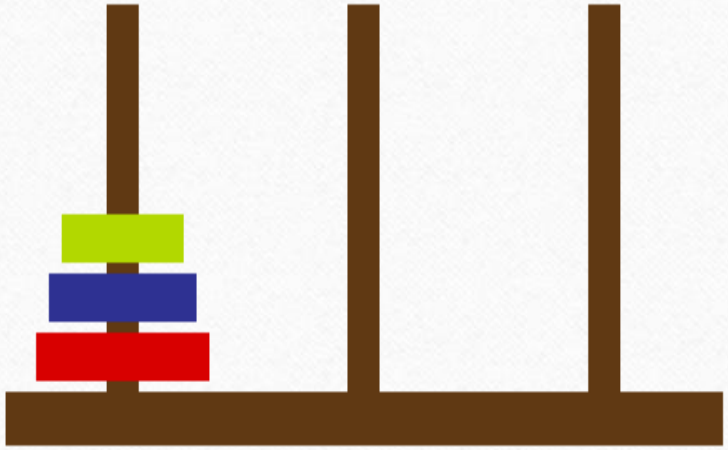
تعدادي حلقه با اندازه هاي متمايز، همانند شكل وجود دارد .اين حلقه ها به ترتيب اندازه از 1تـا) nاز كوچک به بزرگ (شماره گذاري

می شوند .

همچنين سه ميله نيز وجود دارد :يکـی از ميلـه هـا ،ميله مبدأ (A)، يکی ميله ي كمكی (B)و ديگري ميله ي مقصد (C) است .

هدف مسأله انتقال تمـام حلقه ها از میله ی مبدأ به میله ی مقصد، با رعایت شرایط زیر است :

- در ابتدا همه ي حلقه ها به ترتیب، از بزرگترین حلقه در پایین تا کوچـکتـرین حلقـه در بـالا، درمیله ي مبدأ قرار دارند .
- در هر زمان فقط یک حلقه را می توان جابه جا نمود .
- هرگز نباید حلقه اي روي حلقه ي کوچک تر از خود قرار بگیرد .
- هرگز حلقه اي خارج از میله ها قرار نمی گیرد .



شبه کد حل این مسئله به روش بازگشتی

```
Hanoi( A, B, C : pivot ; n : integer )
{
    if n == 1 then
        MOVE ( A -> C );
    else
    {
        Hanoi( A, C, B, n-1 );
        MOVE ( A -> C );
        Hanoi( B, A, C, n-1 );
    }
}
```

محاسبه مرتبه زمانی مساله برج های هانوی

هدف تعداد حرکت ها است

$$T(n) = \begin{cases} 1 & ; \quad n = 1 \\ T(n-1) + T(n-1) + 1 = 2T(n-1) + 1 & ; \quad else \end{cases}$$

$$T(n) = 2T(n-1) + 1$$

$$= 2(2T(n-2) + 1) + 1 = 2^2 T(n-2) + 3$$

$$= 2^2(2T(n-3) + 1) + 1 = 2^3 T(n-3) + 7$$

...

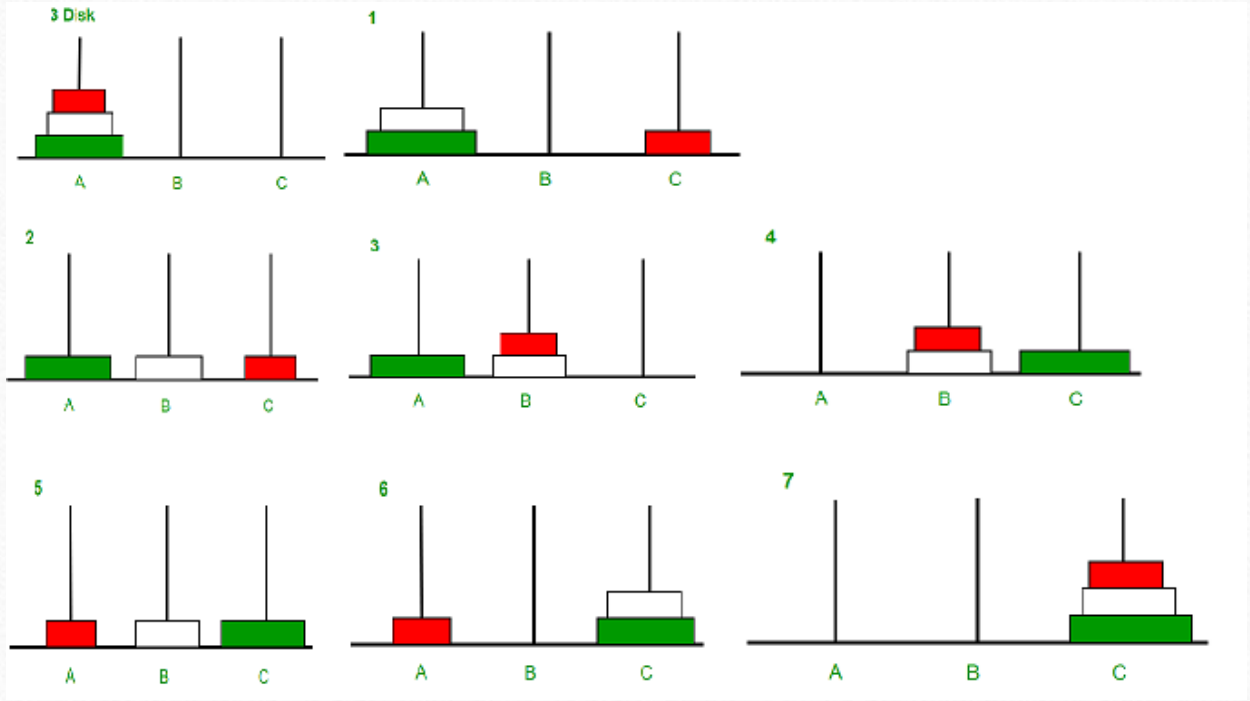
$$= 2^k\, T(n-k) + 2^k - 1 \xrightarrow{k=n-1} 2^{n-1}T(1) + 2^{n-1} - 1 = \mathbf{2^n - 1}$$

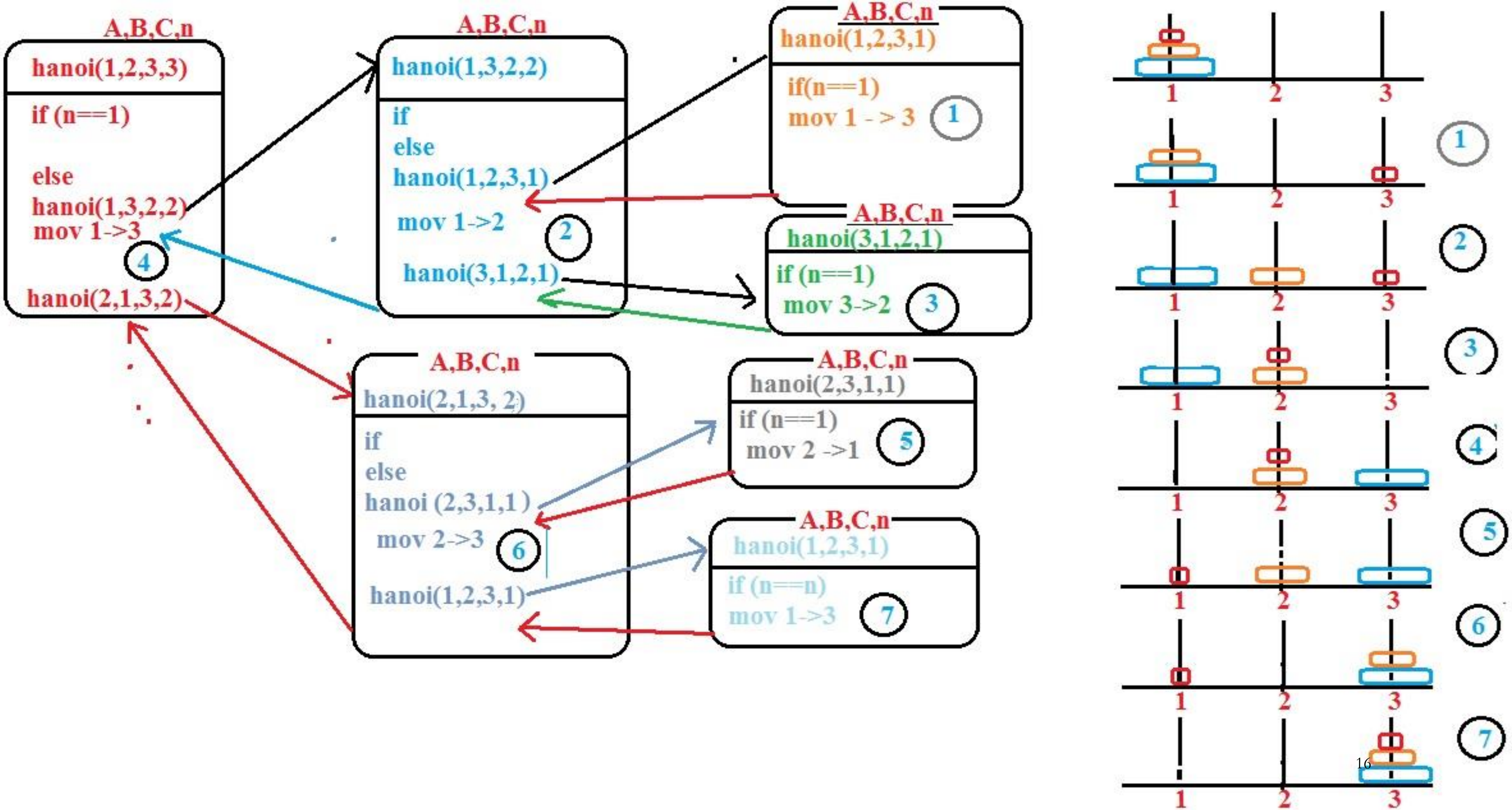
یک ثانیه برای هر جا به جایی :

$n=100 \quad => \quad 2^{100}-1 = 10^{30}$

نمایش گام به گام حل مسأله برج های هانوی برای 3حلقه :

Hanoi (A, B, C, 3)
- Hanoi (A, C, B, 2)
  - Hanoi (A, B, C, 1)
    - A --> C (1)
  - A --> B (2)
  - Hanoi (C, A, B, 1)
    - C --> B (3)
- A --> C (4)
- Hanoi (B, A, C, 2)
  - Hanoi (B, C, A, 1)
    - B --> A (5)
  - B --> C (6)
  - Hanoi (A, B, C, 1)
    - A --> C (7)

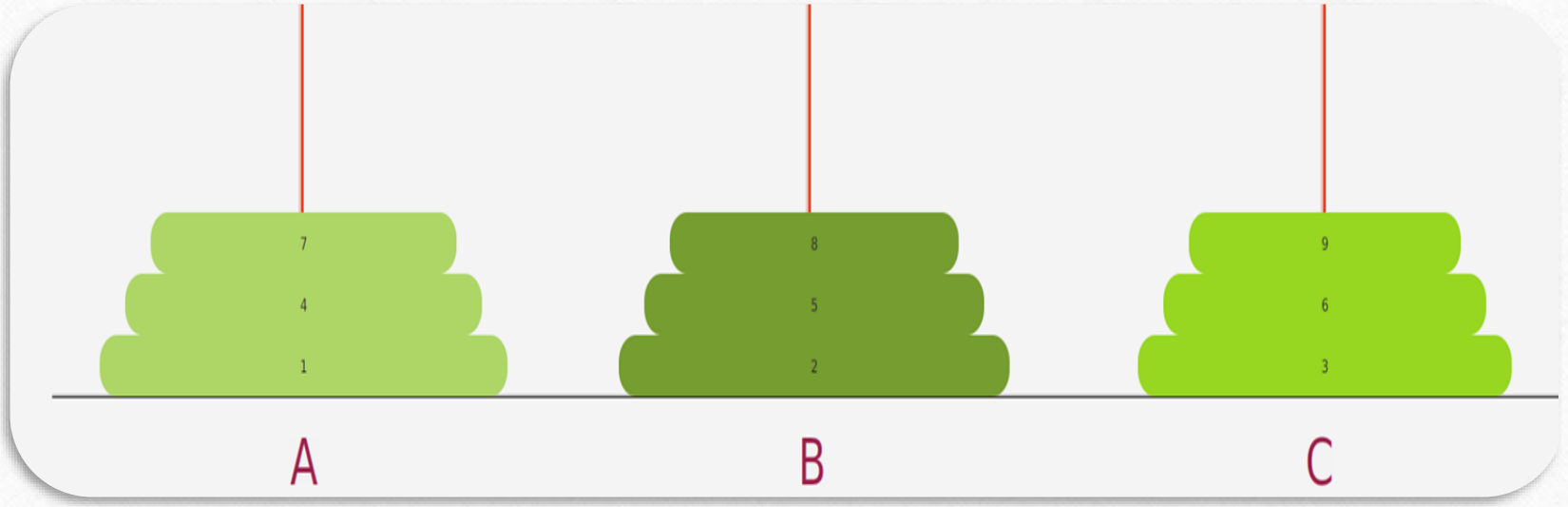



A,B,C,n
hanoi(1,2,3,3)
if (n==1)
else
hanoi(1,3,2,2)
mov 1->3
4
hanoi(2,1,3,2)
A,B,C,n
hanoi(1,3,2,2)
if
else
hanoi(1,2,3,1)
mov 1->2
2
hanoi(3,1,2,1)
A,B,C,n
hanoi(1,2,3,1)
if(n==1)
mov 1 - > 3
1
A,B,C,n
hanoi(3,1,2,1)
if (n==1)
mov 3->2
3
A,B,C,n
hanoi(2,1,3, 2)
if
else
hanoi (2,3,1,1)
mov 2->3
6
hanoi(1,2,3,1)
A,B,C,n
hanoi(2,3,1,1)
if (n==1)
mov 2 ->1
5
A,B,C,n
hanoi(1,2,3,1)
if (n==n)
mov 1->3
7
1 2 3
1
2
3
4
5
6
7

Ex Hanoi

- در این مساله هانوی پیشرفته، در هر میله به تعداد مساوی حلقه وجود دارد بطوری که اندازه حلقه ها متفاوت است و شیوه قرار گیری حلقه ها در میله ها، در شکل زیر نشان داده شده است.

- هدف یافتن الگوریتمی برای حل این مساله از برج های هانوی می باشد که قوانین آن همانند قوانین هانوی معمولی است .در پایان باید تمام حلقه ها به ترتیب اندازه در میله مقصد بر روی هم قرار گرفته باشند.



Ex Hanoi Algorithm

```
ExHanoi(A, B, C: pivot; n : integer)
{
    if n == 1
    {
        MOVE(C -> B);
        MOVE(A -> C);
        MOVE(B -> A);
        MOVE(B -> C);
        MOVE(A -> C);
    }
    else
    {
        ExHanoi(A, B, C, n - 1);
        Hanoi(C, A, B, 3*n - 2);
        MOVE(A -> C);
        Hanoi(B, A, C, 3*n - 1);
    }
}
```

تمرین برنامه نویسی

کد مساله ExHanoiرا به زبانی دلخواه پیاده سازی کنید.

*تعداد حلقه ها بصورت ورودی از کاربر دریافت شود.

*مجاز به استفاده از کدهای آماده و موجود در اینترنت یا دانشجویان ترم های قبل نمی باشید.

*گرافیک نمره اضافه دارد که شامل موارد زیر می باشد :

1.نمایش میله ها و حلقه های موجود در هر میله)اندازه ها رعایت شود.(

2.دکمه هایی برای کارهای :جا به جایی حلقه ها بطور خودکار، بطور دستی، توقف حالت خودکار.

3.وجود انیمیشن برای جا به جایی حلقه ها یک مثبت دارد.

قضیه اصلی یا MASTER THEOREM

قضیه اصلی برای حل روابط بازگشتی به شکل زیر به کار می رود :

$$T(n) = aT(\frac{n}{b}) + \mathbf{f(n)}$$

با فرض :

$$T\left(\frac{n}{b}\right) \equiv T\left(\left\lfloor\frac{n}{b}\right\rfloor\right) \equiv T\left(\left\lceil\frac{n}{b}\right\rceil\right)$$

مقادیرثابت $a \geq 1 , b > 1$

تابع مجانبی در بینهایت $+$ $f(n) : (+\infty)$

۱. اگر $\varepsilon > 0$ وجود داشته باشد بطوریکه $f_n = O\left(n^{(\log_b a \ - \varepsilon)}\right)$:

$$\Rightarrow T(n) = \ \theta\left(n^{\log_b a}\right)$$

۲. اگر $f_n = \theta\left(n^{\log_b a}\right)$:

$$\Rightarrow T(n) = \theta\left(n^{\log_b a} \log n\right) = \theta(f_n \log n)$$

۳. اگر $\varepsilon > 0$ وجود داشته باشد بطوریکه

$$T(n) = \theta(f_n) \Leftarrow af\left(\frac{n}{b}\right) \leq c\, f(n)\,; if\ c > 1\,, f_n = \Omega\left(n^{(\log_b a + \varepsilon)}\right)$$

به زبان ساده تر :

$$T(n) = a\, T\left(\frac{n}{b}\right) + f_n$$

$$\frac{\max(\overset{A}{n^{\log_b a}}, \overset{B}{f_n})}{\min(\underset{A}{n^{\log_b a}}, \underset{B}{f_n})} = D$$

$$D = \begin{cases} D > \theta(n^{\varepsilon}) \Rightarrow T(n) = Max(A\,, B) \\ else \Rightarrow T(n) = Max(A\,, B) \times \log n \end{cases}$$

یک نکته

از لحاظ order :

$$1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}=\log n$$

$$T(n) = 9T\left(\frac{n}{3}\right) + n$$

T(n)را بیابید .

$$\frac{\max(n^{\log_3 9}, n)}{\min(n^{\log_3 9}, n)} = \frac{n^2}{n} = n$$

$$\Rightarrow T(n) = max = O(n^2)$$

$$T(n) = T\left(\frac{2n}{3}\right) + 1$$

T(n) را بیابید .

$$\frac{\max(n^0, 1)}{\min(n^0, 1)} = 1$$

$$\Rightarrow T(n) = 1 \times \log n$$

$$T(n) = 3T\left(\frac{n}{4}\right) + n\log n$$

T(n)را بیابید .

$$n^{\log_4 3} = n^{0.7}$$

$$D = \frac{\max(n^{0.7}, n\log n)}{\min(n^{0.7}, n\log n)} = \frac{n\log n}{n^{0.7}} = n^{0.3}\log n$$

$$D > \Theta(n^{\varepsilon}) \Rightarrow T(n) = n\log n$$

$$T(n) = 2T\left(\frac{n}{2}\right) + n\log n$$

T(n)را بیابید .

$a = 2 \qquad b = 2 \qquad f_n = n\log n = B \qquad A = n^{\log_2 2}$

$$D = \frac{\max(A,B)}{\min(A,B)} = \frac{n\log n}{n} = \log n$$

$$\text{else} \rightarrow MAX \times \log n = n\log n \times \log n = n(\log n)^2$$

T(n)را بیابید .

```
sum( L : Array, start, end : int ) : int
{
        if start == end
                ret L[start];
        else
        {
                a = sum( L, start, ⌊(start+end)/2⌋);
                b = sum( L, ⌊(start+end)/2⌋ + 1, end);
                ret a + b;
        }
}
```

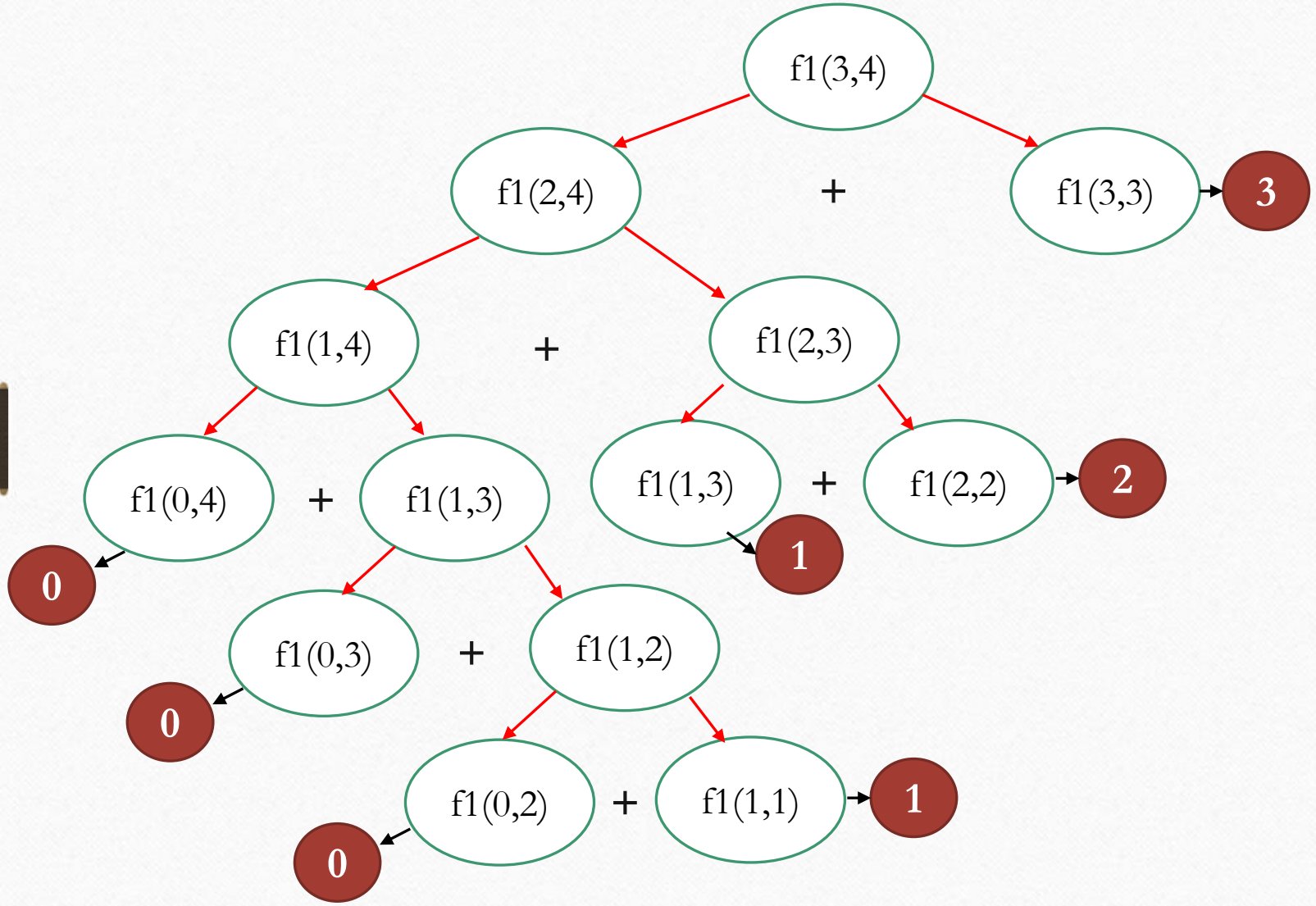
$$T(n) = 2T\left(\frac{n}{2}\right) + 1$$

$$D = \frac{\max(n, 1)}{\min(n, 1)} = n$$

$$\Rightarrow T(n) = n$$

خروجی برای فراخوانی $f1(3,4)$ چیست؟

```
f1( integer a, integer b) : integer
{
        if a < b and a ≥ 1 and b ≥ 1
                return (f1( a-1, b) + f1(a, b-1));
        else
                return a;
}
```


f1(3,4)
f1(2,4)
+
f1(3,3)
3
f1(1,4)
+
f1(2,3)
f1(0,4)
+
f1(1,3)
f1(1,3)
+
f1(2,2)
2
0
1
f1(0,3)
+
f1(1,2)
0
f1(0,2)
+
f1(1,1)
1
0

جواب نهایی

7

تمرین : خروجی برای فراخوانی $f1(3,4)$ را بیابید.

```
f1( int a, int b) : int
{
        if a = = 1 or b = = 1
                ret 1;
        if a < b
                ret (f2(b-1, a) × a);
        else
                ret a + b;
}
```

```
f2( int a, int b) : int
{
        if a = = 1 or b = = 1
                ret 1;
        if a ≥ b
                ret (f1(a-1, b) + b);
        else
                ret a – b;
}
```

پایان