Atividade de Estrutura de Dados
Semana 2

1. Nesta versão do quebra-cabeça estendida da Torre de Hánoi, temos 4 pinos verticais e vários discos de vários tamanhos. Cada disco tem um buraco no centro para encaixar nos pinos.

   As regras do quebra-cabeça são as seguintes:

   - O quebra-cabeça começa com todos os discos colocados em um dos pinos. Eles são colocados em ordem do maior para o menor, de baixo para cima.
   - O objetivo do quebra-cabeça é mover todos os discos para o último pino.
   - Apenas um disco pode ser movido por vez, e os discos são sempre colocados em pinos.
   - Os discos só podem ser movidos para um pino vazio ou para um disco maior.

   Apresente o pseudocódigo que resolve esse quebra-cabeça.

   a) Se $n \leq 1$, resolva diretamente
   b) Mantenha 2 maiores discos no pino A e transfira $n - 2$ discos do pino A para o pino B.
   c) Transfira os dois maiores discos do pino A para o pino D sem pertubar os n-2 discos já instalados no pino B.
   d) Transfira n-2 discos do pino B para o pino D sem pertubar os maiores discos já instalados no pino D.

```
int hanoi4(int n, int start, int aux1, int aux2, int end){
    if(n == 0){
        return 0;
    } else if(n == 1){
        printf("Mova disco 1 do pino %d para pino %d\n", start, end);
        return 1;
    }else{
```

```
        int cnt = 0;
        cnt += hanoi4(n-2, start, aux2, end , aux1);
        printf("Mova disco %d do pino %d para pino %d\n", n-1, start, au
        printf("Mova disco %d do pino %d para pino %d\n", n  , start, en
        printf("Mova disco %d do pino %d para pino %d\n", n-1, aux2, end
        cnt += 3;
        cnt += hanoi4(n-2, aux1, start, aux2, end);
        return cnt;
    }
}
```

Algoritmo de Frame-Stewart:

a) Se $n \leq 1$, resolva diretamente

b) Escolha um $k \in [1..n]$

c) Mantenha k maiores discos no pino A e transfira $n - k$ discos do pino A para o pino B.

d) Transfira os k maiores discos do pino A para o pino D sem perturbar os n-k discos já instalados no pino B. Como um disco maior não pode ser colocado sobre um disco menor, então o pino B não pode ser usado, ou seja, utilizaremos a solução do problema com 3 pinos.

e) Transfira n-2 discos do pino B para o pino D sem perturbar os maiores discos já instalados no pino D.

Frame(Frame, 1941) e Stewart(Stewart & Frame, 1941) propuseram como escolha ótima para o valor de k, o maior inteiro tal que

$$\sum_{i=1}^{k} i \leq n$$

```
int findK(int n){
    int s = 0;
    int i = 1;
    while(s < n){
        s += i;
        if(s == n) return i;
        else if(s > n) return i-1;
        i++;
    }
}
```

```
int hanoi3(int i, int j, int start, int aux1,int end){
    int n = j-i+1;
    if(n == 0){
        return 0;
    } else if(n == 1){
        printf("Mova disco %d do pino %d para pino %d\n", i, start, end);
        return 1;
    }else{
        int cnt = 0;
        cnt += hanoi3(i, j-1, start, end, aux1);
        printf("Mova disco %d do pino %d para pino %d\n", j, start, end);
        cnt += 1;
        cnt += hanoi3(i, j-1, aux1, start, end);
        return cnt;
    }

}

int hanoi4(int i, int j, int start, int aux1, int aux2, int end){
    int n = j-i+1;
    if(n == 1){
        printf("Mova disco 1 do pino %d para pino %d\n", start, end);
        return 1;
    }else if(n==2){
        printf("Mova disco 1 do pino %d para pino %d\n", start, aux1);
        printf("Mova disco 2 do pino %d para pino %d\n", start, end);
        printf("Mova disco 1 do pino %d para pino %d\n", aux1, end);
        return 3;
    }else{
        int cnt = 0;
        int k = findK(n);
        int t = n-k;
        cout << "k" << k << endl;
        cnt += hanoi4(i, i+t-1, start, aux2, end, aux1);
        cnt += hanoi3(i+t, j, start, aux2, end);
        cnt += hanoi4(i, i+t-1, aux1, start, aux2, end);
        return cnt;
    }

}
```

n	$T_3(n)$	$T_4(n,2)$	Frame-Stewart
1	1	1	1
2	3	3	3
3	7	5	5
4	15	9	9
5	31	13	13
6	63	21	17
7	127	29	25

2. O número de Stirling do segundo tipo calcula o número de maneira de particionar n elementos em k subconjuntos não-vazios, denotado por $\left\{ {n \atop k} \right\}$. Lembrando que a cardinalidade da união dos subconjuntos deve ser igual a n. Por exemplo, o conjunto {a,b,c,d} pode ser particionado em 2 subconjuntos não-vazios de 7 maneiras diferentes

   - {a} e {b,c,d}
   - {a,b} e {c,d}
   - {b} e {a,c,d}
   - {a,b,c} e {d}
   - {b,c} e {a,d}
   - {a,b,d} e {c}
   - {a,c} e {b,d}

   a) Proponha uma definição recursiva para $\left\{ {n \atop k} \right\}$. [Dica: Escolha um objeto qualquer entre os n objetos disponíveis, ele pode formar um subconjunto unitário ou participar de algum outro subconjunto não-vazio.]

$$\left\{ {n \atop k} \right\} = \left\{ {n-1 \atop k-1} \right\} + k \left\{ {n-1 \atop k} \right\}$$

   b) De quantas maneiras eu consigo particionar n objetos em 0 subconjuntos não vazios, ou seja, qual é o valor $\left\{ {n \atop 0} \right\}$?

$$\left\{ {n \atop 0} \right\} = \begin{cases} 1 & , n == 0 \\ 0 & , \text{caso contrário} \end{cases}$$

   c) De quantas maneiras eu consigo particionar 0 objetos em 0 subconjuntos não vazios, ou seja, qual é o valor de $\left\{ {0 \atop 0} \right\}$? 1

   d) De quantas maneiras eu consigo particionar n objetos em 1 subconjuntos não vazios, ou seja, qual é o valor de $\left\{ {n \atop 1} \right\}$? 1

e) De quantas maneiras eu consigo particionar n objetos em n subconjuntos não vazios, ou seja, qual é o valor de $\left\{ {n \atop n} \right\}$?1

f) Preencha a seguinte tabela de número de Stirling de segundo tipo:

Preencha a seguinte tabela de números binomias:

```
int stirling(int n, int k){

    if(n == 0 && k == 0) return 1;
    if(k == 0) return 0;
    if(k == 1) return 1;
    if(k == n) return 1;
    else return stirling(n-1, k-1) + k*stirling(n-1, k);
}
```

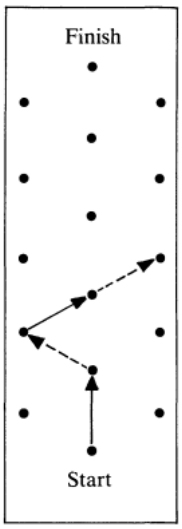
n/k	0	1	2	3	4
0	1	X	X	X	X
1	0	1	X	X	X
2	0	1	1	X	X
3	0	1	3	1	X
4	0	1	7	6	1

3. O jogo da escalada é um jogo para dois jogadores. Um lápis é colocado no ponto rotulado ”start” e o os jogadores se revezam para deslizar este lápis pela grade de pontos conforme as seguintes regras:

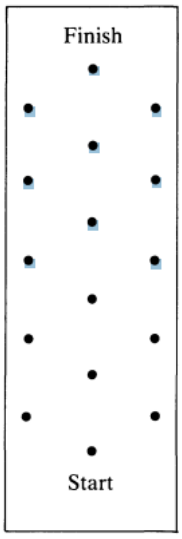
- A cada turno, o lápis só pode ser movido para um ponto mais alto que sua posição atual.
- Cada movimento pode, portanto, ocorrer apenas em um das três direções.

- O primeiro jogador que desliza o lápis para o ponto rotulando com ”finish” ganha o jogo.

a) Este diagrama seguinte mostra o início de um jogo, jogado entre Sara e Paulo. Os movimentos de Sarah são indicados por setas sólidas. Os movimentos de Paulo são indicados por setas pontilhadas. É a vez de Sarah. Ela tem dois movimentos possíveis. Mostre que a partir de um desses movimentos Sarah pode garantir que ela ganha, independente dos movimentos realizados por Paulo.



b) Se o jogo for jogado desde o início e Sara tem o primeiro movimento, então ela sempre pode ganhar o jogo se ela jogar corretamente. Explique como Sarah deve jogar para ter certeza de ganhando. [Dica: Para construir a estratégia vencedora para a Sara, utilize os rótulos W para uma ponto vencedor e L para um ponto perdedor. Analise se o processo de rotulação deve ser realizado a partir dos pontos próximos do Start ou do Finish.]



Referências

Frame, J. S. (1941). Solution to advanced problem 3918. *Amer. Math. Monthly*, *48*, 216–217.

Stewart, B. M., & Frame, J. (1941). Solution to advanced problem 3918. *The American Mathematical Monthly*, *48*(3), 216–219.