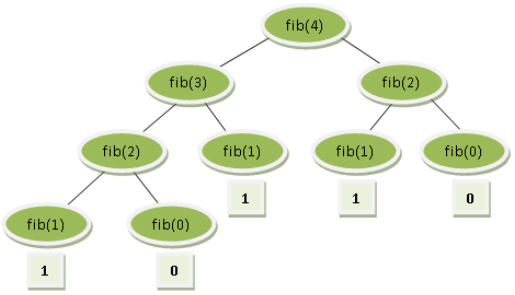
LISTA PARA A PROVA DE ALGEST 2017.1

1) Quase todo estudante de Ciência da Computação recebe em algum momento no início de seu curso de graduação algum problema envolvendo a sequência de Fibonacci. Tal sequência tem como os dois primeiros valores 0 (zero) e 1 (um) e cada próximo valor será sempre a soma dos dois valores imediatamente anteriores. Por definição, podemos apresentar a seguinte fórmula para encontrar qualquer número da sequência de Fibonacci:

```
fib(0) = 0
fib(1)= 1
fib(n) = fib(n-1) + fib(n-2);
```

Uma das formas de encontrar o número de Fibonacci é através de chamadas recursivas. Isto é ilustrado a seguir, apresentando a árvore de derivação ao calcularmos o valor fib(4), ou seja o 5º valor desta sequência:



Desta forma,

- fib(4) = 1+0+1+1+0 = 3
- Foram feitas 8 calls, ou seja, 8 chamadas recursivas.

A primeira linha da entrada contém um único inteiro **N**, indicando o número de casos de teste. Cada caso de teste contém um inteiro **X** (1 ≤ **X** ≤ 39) .

Determine Ω e O do problema e a complexidade do algoritmo proposto.

2) O fatorial de um número inteiro positivo N, denotado por N!, é definido como o produto dos inteiros positivos menores do que ou iguais a N. Por exemplo 4! = 4 × 3 × 2 × 1 = 24.

Dado um inteiro positivo N, você deve escrever um programa para determinar o menor número k tal que N = a_1! + a2! + ... + a_k!, onde cada a_i, para 1 ≤ *i* ≤ *k*, é um número inteiro positivo.

Por exemplo, para N = 10 a resposta é 3, pois é possível escrever N como a soma de três números fatoriais: 10 = 3! + 2! + 2!. Para N = 25 a resposta é 2, pois é possível escrever N como a soma de dois números fatoriais: 25 = 4! + 1!.

A entrada consiste de uma única linha que contém um inteiro **N** (1 ≤ **N** ≤ 10^5).

Seu programa deve produzir uma única linha com um inteiro representando a menor quantidade de números fatoriais cuja soma é igual ao valor de **N**.

Proponha um algoritmo para a resolução do problema e diga sua complexidade.

3) João está trabalhando em uma mina, tentando retirar o máximo que consegue de diamantes "<>". Ele deve excluir todas as particulas de areia "." do processo e a cada retirada de diamante, novos diamantes poderão se formar. Se ele tem como uma entrada .<...<<..>>....>....>>>., três diamantes são formados. O primeiro é retirado de <..>, resultando .<...<>....>....>>>. Em seguida o segundo diamante é retirado, restando .<.......>....>>>. O terceiro diamante é então retirado, restando no final .....>>>., sem possibilidade de extração de novo diamante.

Deve ser lido um valor inteiro **N** que representa a quantidade de casos de teste. Cada linha a seguir é um caso de teste que contém até 1000 caracteres, incluindo "**<**,**>**, **.**"

Você deve imprimir a quantidade de diamantes possíveis de serem extraídos em cada caso de entrada.

Determine Ω do problema e a complexidade do algoritmo proposto.Diga se o algorítmo em questão é ótimo ou não.

4)Considerando a entrada de valores inteiros não negativos, ordene estes valores segundo o seguinte critério:

- Primeiro os Pares
- Depois os Ímpares

Sendo que deverão ser apresentados os pares em ordem crescente e depois os ímpares em ordem decrescente.

A primeira linha de entrada contém um único inteiro positivo **N** ($1 < \mathbf{N} < 10^5$) Este é o número de linhas de entrada que vem logo a seguir. As próximas **N** linhas conterão, cada uma delas, um valor inteiro não negativo.

Apresente todos os valores lidos na entrada segundo a ordem apresentada acima. Cada número deve ser impresso em uma linha, conforme exemplo abaixo.

Determine Ω e O do problema e a complexidade do algoritmo proposto.

5) O Professor solicitou que você escreva um programa que converta uma expressão na forma infixa (como usualmente conhecemos) para uma expressão na forma posfixa. Como você sabe, os termos in (no meio) e pos (depois) se referem à posição dos operadores. O programa terá que lidar somente com operadores binários +,-,*,/,^, parênteses, letras e números. Um exemplo seria uma expressão como: (A*B+2*C^3)/2*A.

O programa deve converter esta expressão (infixa) para a expressão posfixa: AB*2C3^*+2/A*

A primeira linha da entrada contém um valor inteiro **N** (**N** < 1000), que indica o número de casos de teste. Cada caso de teste a seguir é uma expressão válida na forma infixa, com até 300 caracteres.

Para cada caso, apresente a expressão convertida para a forma posfixa.

Determine Ω e O do problema e a complexidade do algoritmo proposto.