Resolva cada exercício em Folhas Separadas

1. Um engenheiro informático está a desenvolver um sistema aproveitando o paralelismo das arquiteturas atuais em que várias instâncias de uma certa função podem ser executadas em paralelo. Tradicionalmente, na chamada de uma função os seus argumentos são guardados numa pilha (de argumentos), neste novo sistema existirá uma lista de pilhas, uma para cada chamada. A lista está ordenada por ordem crescente do número de ordem das pilhas. Para isso, cada pilha é inserida na lista de acordo com o seu número de ordem. No caso de já existir uma pilha com esse número, então os endereços dos argumentos da pilha nova são adicionados à existente (o primeiro argumento no topo, o segundo, sobre o primeiro e assim sucessivamente).

```
class F_Argumentos {   //classe representativa da pilha de argumentos
   private:
      unsigned int num_ordem;       // Nº de ordem da pilha
      stack<string> stack_args;    // stack com endereços de argumentos

   public:
       F_Argumentos() ;
       F_Argumentos(const F_Argumentos& fa) ;
       ~F_Argumentos() ;
       …  } ;

class Invoc_par: public list<F_Argumentos> {
    public:
       Invoc_par () ;
       Invoc_par (const Invoc_par& ip) ;
       ~Invoc_par ();
       …  } ;
```

Acrescente às classes todos os métodos necessários para realizar as seguintes operações, tendo em atenção as classes mais adequadas aos métodos que vai desenvolver:

a) Eliminar caso exista, uma pilha de argumentos, dando o seu número

b) Guardar os argumentos de uma chamada.

2. Faça a análise da complexidade temporal T(n) do código abaixo. Justifique.

```
long int metodo (int m, long int n)
{
   long int x=m;
   int r=0;

   while (x > 1 && x <= n) {
      x=x*m ;
      r=r+1 ; }

   return r ;  }
```

Resolva cada exercício em Folhas Separadas

3. A intersecção de conjuntos pode ser representada através de um grafo, em que cada vértice guarda um conjunto e existe uma aresta a ligar dois vértices se da intersecção entre os conjuntos representados por esses dois vértices resulta um conjunto não vazio. A aresta contém o número de elementos resultantes da intersecção.

a) Defina a classe NÃO TEMPLATE ConjCaract de forma a descrever um grafo não dirigido que represente a intersecção de vários conjuntos de caracteres. Para maior simplicidade represente o conjunto de caracteres por strings (não é necessário indicar os métodos da classe).

b) Elabore um método que recebe um vector de strings e cria o respetivo grafo. Considere o método str_intersection que devolve o número de caracteres resultante da intersecção de duas strings: `int str_intersection (string s1, string s2)`

c) Adicione à classe um método que calcula o número total de interseções de um grafo

4. Defina na classe template tree<TN> um método que devolva pelos parâmetros os valores (chaves) máximo e mínimo da árvore.

5.

a) Ilustre como são armazenados os valores da sequência 30-20-40-10-12 numa árvore AVL e indique a(s) rotaça(ões) efetuada(s).

b) Apresente os elementos da árvore acima segundo uma travessia em pre-ordem.