# Prova Final de CI164 – Introdução à Computação Científica

Prof. Daniel Weingaertner
**06.02.2017** – Turmas A e B

Regras Gerais:

1. A avaliação é **individual** e **sem consulta**. A nota máxima é 100.
2. A interpretação do enunciado faz parte da prova.
3. Todas justificativas para melhorias de desempenho devem explicar o motivo subjacente ao problema/solução. Ex.: não basta escrever "loop unrolling", mas é necessário explicar por que ele eventualmente melhora determinado código.

**Nome:** **GRR:** **Turma:**

## *Questão 1 (50 pontos)*

Seja $L=\{(x_i,y_i)\}\forall i\in[1,N]$ uma lista de pontos/coordenadas 2D, de tamanho $N$, e sejam $1\leq p_1<p_2\leq N$ duas posições nesta lista. Podemos utilizar a técnica de Regressão Linear por Quadrados Mínimos para calcular os coeficientes $\alpha$ e $\beta$ da função potência $pow(x)=\alpha x^{(-\beta)}$ que melhor aproxima os pontos da lista L no intervalo $[p_1,p_2]$ através das equações:

$$\beta=\frac{\sum_{i=p_1}^{p_2}(\log(x_i)-\overline{\log(x)})(\log(y_i)-\overline{\log(y)})}{\sum_{i=p_1}^{p_2}(\log(x_i)-\overline{\log(x)})^2} \quad \text{e} \quad \alpha=\frac{\sum_{i=p_1}^{p_2}(\log(y_i)-\beta\sum_{i=p_1}^{p_2}\log(x_i))}{p_2-p_1+1}$$

, onde $\overline{\log(x)}$ e $\overline{\log(y)}$ são, respectivamente, as médias de $\log(x_i)$ e $\log(y_i)$ no intervalo $[p_1,p_2]$.

**a)** Escreva uma função em linguagem C que receba como parâmetros a lista $L$, o tamanho da lista $N$, os pontos $p_1$ e $p_2$ e calcule, de forma eficiente, os valores de $\alpha$ e $\beta$.

A função "`double log(double x)`" calcula o logaritmo de um número.

**b)** Você deve definir uma estrutura de dados adequada para a lista L, **analisando/justificando** sua eficiência com relação ao seu algoritmo.

## *Questão 2 (50 pontos)*

**a)** Por que a versão B do código abaixo é MAIS EFICIENTE do que a versão A em uma arquitetura X64? **Justifique!**

| Versão A | Versão B |
|---|---|
| `double p[NUM_LIN*NUM_COL];`<br>`double q[NUM_LIN*NUM_COL];`<br>`for (long x=0; x<NUM_COL; ++x)`<br>`  for (long y=0; y<NUM_LIN; ++y)`<br>`  p[x+y*NUM_COL] = q[y+x*NUM_LIN];` | `double p[NUM_LIN*NUM_COL];`<br>`double q[NUM_LIN*NUM_COL];`<br>`for (long y=0; y<NUM_LIN; ++y)`<br>`  for (long x=0; x<NUM_COL; ++x)`<br>`  p[x+y*NUM_COL] = q[y+x*NUM_LIN];` |

**b)** Reimplemente a versão A do código acima utilizando a técnica de "loop unroll & jam" e responda se ela aumenta o desempenho deste código **justificando sua resposta**.

**c)** Reimplemente a versão A do código acima utilizando a técnica de "loop blocking" nas duas dimensões e responda se o "blocking" aumenta o desempenho deste código, **justificando sua resposta**.