EXERCÍCIO PROGRAMA 2

Data de entrega: 9/5/2002

Problema. *Escreva um programa que recebe um texto como entrada e devolve o trecho repetido mais comprido neste texto.*

Por exemplo, se o texto de entrada for

```
3.14159265358979323846264338327950288419716939937510582097
94459230781640628620899862803482534211706798214808651328230
66470938446095505822317253594081284811174502841027019385211055
59644622948954930381964428810975665933446128475648233786783
1652712019091456485669234603486104543266482133936072602491412
737245870066...
```

então a saída de seu programa deve ser `0348`. No *King James' Bible* do Projeto Gutenberg (removendo os números dos versos e ignorando quebras de linha), temos o seguinte texto repetido:

```
, offered: His offering was one silver charger, the weight whereof was an
hundred and thirty shekels, one silver bowl of seventy shekels, after the
shekel of the sanctuary; both of them full of fine flour mingled with oil
for a meat offering: One golden spoon of ten shekels, full of incense: One
young bullock, one ram, one lamb of the first year, for a burnt offering:
One kid of the goats for a sin offering: And for a sacrifice of peace
offerings, two oxen, five rams, five he goats, five lambs of the first year:
this was the offering of Ahi
```

onde quebramos em linhas para facilitar a apresentação. (O arquivo processado tem 4169466 caracteres.)

Você deve tentar descobrir um bom algoritmo para resolver este problema. O algoritmo ingênuo, discutido em um exercício em sala, tem comportamento pelo menos quadrático, que o torna inútil para tratar textos longos.

Entregue não só os programas (por correio eletrônico para `yoshi@ime.usp.br` e `daniel@linux.ime.usp.br`) como também um relatório. Envie seu relatório eletronicamente também.

1. Para os testes de seu programa, use entradas variadas. O Projeto Gutenberg é uma boa fonte de exemplos (grandes). Você também pode usar cadeias de caracteres gerados aleatoriamente.
2. Você pode usar o comando `time` para determinar o tempo gasto pelo seu programa, como no EP1.
3. Use o truque do `o` ("o pequeno") visto em sala para contar *o número de comparações entre caracteres* que o seu programa executa. Use o `GNUPLOT` para verificar como cresce este número conforme o texto de entrada cresce.
4. Explique no relatório o funcionamento do seu sistema e discuta seus resultados.