Mini EP1: A busca pelos primos que são representáveis pela soma de dois quadrados

Alfredo Goldman, Elisa Silva e Luciana Marques
MAC 0219-5742 – Programação Concorrente e Paralela 2021

Entrega até 26 de abril de 2021

1. Introdução

Neste primeiro mini EP buscamos avaliar a diferença de desempenho entre linguagens "rápidas" e "lentas", introduzindo formas simples de realizar as medidas.

A velocidade de uma linguagem está atrelada ao quão rápido ela realiza um determinado conjunto de ações em relação a outras linguagens. Isto difere da comparação de velocidade entre programas, onde o conjunto de ações pode ser diferente desde que a saída seja correta.

Nessa primeira tarefa vocês deverão implementar o crivo de Eratóstenes para encontrar primos menores que um certo N e depois usar o teorema de Fermat sobre a soma de dois quadrados para descobrir se o número primo pode ser escrito pela soma de dois quadrados.

Esse resultado é facilmente obtido ao testar se o resto da divisão do primo por 4 não é igual a 3. Para mais informações visite o link: http://eulerarchive.maa.org/docs/translations/E228en.pdf

Vocês terão a liberdade de escolher as linguagens "rápidas" e "lentas" seguindo a restrição de que a linguagem "rápida" deve ser uma linguagem compilada (preferencialmente para linguagem de máquina) e a linguagem "lenta" deve ser uma linguagem interpretada.

2. Tarefas

Sua primeira tarefa é definir as linguagens em que irá implementar o programa a ser estudado. A linguagem "lenta" deve ser uma linguagem interpretada e a linguagem "rápida" deve ser uma linguagem compilada. Algumas linguagens são complexas para serem classificadas como compiladas ou interpretadas, como Java, nestes casos compare a desempenho com a outra linguagem escolhida e considere ela "lenta" se demorar mais para executar ou "rápida" se executar em menos tempo (Java é "rápida" em comparação com Python e "lenta" em comparação com Rust).

Após definir as linguagens, implemente o pseudocódigo a seguir nelas:

```
01. # miniEP1                                                              |
02. # Nome, NºUSP                                                          |
03. # comando para compilar o programa, ex: cc prog.c                      |
04. var N = 1 << lerNumeroDaEntrada(); # << é a operação de bitshift       |
05. var primos = 0;                                                        |
06. var primosEspeciais = 0;                                               |
07. var crivo = cria lista de Booleanos com N+1 elementos # (*)            |
08. para var i = 2 até N: # inclusivo                                      |
09. >   crivo[i] = verdadeiro;                                             |
10. para var i = 2 até N: # inclusivo                                      |
11. >   se crivo[i] for verdadeiro:                                        |
12. >   >   primos = primos + 1;                                          |
13. >   >   se i % 4 != 3: primosEspeciais = primosEspeciais + 1;         |
14. >   >   para var j = i*2 até N com passo de tamanho i: # inclusivo     |
15. >   >   >    crivo[j] = falso;                                         |
16. imprima(primos, primosEspeciais);                                      |
```

(*) no caso de linguagens que indexam a partir do 1, pode criar uma lista com N elementos apenas.

Busque implementar usando os recursos da linguagem, mas sem alterar a natureza do pseudocódigo. No caso da linguagem não possuir um tipo Booleano, use o tipo inteiro com 1 para representar verdadeiro e 0 para representar falso. Não esqueça de incluir os três comentários do pseudocódigo no começo código-fonte contendo seu nome e como compilar o programa.

Para medir o tempo de execução recomendamos você usar o comando `time`. A maioria dos sistemas *Unix-like* já vêm com ele e no *Windows* ele pode ser obtido através da instalação do *Cygwin*. Seu uso é bem simples, basta adicionar `time` no começo do comando que irá executar como:

De `$python3 programa.py` para `$time python3 programa.py`.

Ao terminar a execução do programa ele irá imprimir na saída uma linha como essa:

```
./a.out  0.60s user 0.04s system 16% cpu 4.001 total
```

Desses tempos, você irá usar o tempo "user" (0,6 segundos no exemplo). Esse tempo ignora o tempo gasto pelas rotinas do sistema operacional.

Realize 5 medidas de tempo de execução para as entradas "24" e "26" para cada programa.

Como atividade extra, você poderá otimizar o programa, mantendo a solução sequencial e enviar seus resultados da versão otimizada no mesmo formulário que enviará as outras medidas.

Para verificar a corretude do seu programa, para a entrada "24" você deve obter como saída "1077871 538764" e para "26" você deve obter como saída "3957809 1978494".

Para ter uma referência, a versão de referência em C costuma demorar 1.16 segundos para a entrada de "26" e a versão de referência em Python 30 segundos para a mesma entrada. Então é esperado que possa observar tempos de execução maiores, como 1 minuto, bem como observar tempos menores.

3. Entrega

Envie os resultados obtidos com informações da sua máquina no seguinte formulário: https://forms.gle/QpnPRfJk3kfGPrYp6**.**

É necessário que entre com seu email USP, bem como será possível editar sua resposta depois do envio.

Envie o código fonte dos programas em um arquivo .ZIP no eDisciplinas da matéria.

Entrega até 26 de abril de 2021.

4. Critério de Avaliação

Os Mini EPs usam um critério de avaliação binária (ou 1 ou 0). Para tirar 1 envie o formulário com os resultados dos experimentos e submeta no eDisciplinas o código fonte dos seus programas.

Vale reforçar parágrafo II do artigo 23 do **Código de Ética da USP**:

> Artigo 23 - É vedado aos membros do corpo discente e demais alunos da Universidade:
> [...]
> II. lançar mão de meios e artifícios que possam fraudar a avaliação do desempenho, seu ou de outrem, em atividades acadêmicas, culturais, artísticas, desportivas e sociais, no âmbito da Universidade, e acobertar a eventual utilização desses meios.

Mini EPs plagiados receberão nota 0.

Se tiver dúvidas, envie uma mensagem no fórum do curso ou envie e-mails para elisa@silva.moe, lucianadacostamarques@gmail.com ou gold@ime.usp.br com *[miniEP1]* no assunto do e-mail. Divirta-se!