UNIVERSIDADE FEDERAL DE SÃO JOÃO DEL REI
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO
ALGORITMOS E ESTRUTURAS DE DADOS III

2º semestre de 2019
Professor: Leonardo Chaves Dutra da Rocha

Trabalho Prático 1
Data de Entrega: 06 de Setembro 2019.

Este trabalho tem por objetivo exercitar primitivas básicas da Linguagem C e iniciar a discussão sobre problemas complexos e a sua solução.

Conforme veremos ao final dessa disciplina, os principais e mais utilizados algoritmos de criptografia são baseado na teoria dos números primos. Os números inteiros podem ser representados como um produto de potência de números primos. Por exemplo, $126 = 2^1 * 3^2 * 7$. Resumidamente, algoritmos como RSA utilizam o conceito de chaves públicas e privadas. Essas, por sua vez, são compostas, em sua essência, da multiplicação de números primos muito grandes. A decomposição desses números levaria muitos e muitos séculos de processamento intenso para que pudesse ser realizada. Mas fique tranquilo, por ora, você não precisa se preocupar exatamente como funciona o processo de criptografia.

Nesse trabalho vamos trabalhar com o conceito de inteiros Xulambs! Um inteiro é Xulambs se pode ser escrito como um produto de **dois ou mais primos distintos, sem repetição**. Por exemplo, $15 = 3 * 5$, $14 = 2 * 7$ e $21 = 3 * 7$ são inteiros Xulambs. Porém $4 = 2^2$, 5, 7, $8 = 2^3$ e $18 = 2 * 3^2$ não são inteiros Xulambs.

Considerando como entrada um arquivo contendo um inteiro N ($1 \leq N \leq 10^{12}$) em cada linha. Você deve fazer um programa que produza um arquivo de saída contendo um inteiro por linha que represente o número de divisores Xulambs de cada valor N passado no arquivo de entrada. Veja exemplo de entrada e saída:

ENTRADA

```
1
8
18
252
```

SAÍDA

```
0
0
1
4
```

Seu programa deve apresentar as seguintes características:

1. O programa deve ser implementado em módulos. Deve ainda ser definido um arquivo de protótipos e definições contendo as estruturas de dados e protótipos das funções dos vários módulos. A compilação do programa deve utilizar o utilitário Make.

2. Todas as estruturas de dados devem ser alocadas dinamicamente, assim como devem ser desalocadas após o processamento. As rotinas de teste de colisão também devem ser implementadas usando estruturas de alocação dinâmica.

3. O programa deve receber dois parâmetros pela linha de comando, utilizando a primitiva getopt:

   (a) arquivo de entrada

(b) arquivo de saída

O programa implementado deve ser avaliado para várias entradas, utilizando as funções `getrusage` e `gettimeofday`. Deve-se também distinguir entre os tempos de computação e tempos de entrada e saída. Comente sobre os tempos de usuário e os tempos de sistema e sua relação com os tempos de relógio.

Avaliação

Deverão ser entregues:

- listagem das rotinas;
- descrição breve dos algoritmos e das estruturas de dados utilizadas;
- análise da complexidade das rotinas;
- análise dos resultados obtidos.

Distribuição dos pontos:

- execução
  execução correta: 60%
- estilo de programação
  código bem estruturado: 20%
  código legível: 20%
- documentação
  comentários explicativos: 30%
  análise de complexidade: 30%
  análise de resultados: 40%

A nota final é calculada como a média harmônica entre execução (E) e documentação (D):

$$\frac{D * E}{\frac{D+E}{2}}$$