Exercício 1

A fatoração de um inteiro em primos, se feita de modo ingênuo, costuma ser custosa pois os algoritmos básicos de geração de números primos não são eficientes. Uma forma de agilizar o processso consiste em utilizar uma lista de números primos pré-computados.

Construa três programas:

1-a. `gen-primes.rkt`: que gera todos os números primos até um valor `n`. O programa deve ser chamado assim (supondo que `n=1000` e estou salvando para o arquivo `primes.csv`):

```
$ ./gen-primes.rkt 1000 primes.csv
```

1-b. `factorize.rkt`: usa uma lista pré-computada de números primos para fatorar um inteiro. Deve ser chamado assim (`2783` é o número a fatorar e `factors.csv` é a lista de fatores):

```
$ ./factorize.rkt 2783 primes.csv factors.csv
```

1-c. `unique.rkt`: recebe um arquivo de fatores e elimina os fatores duplicados (e.g. se os fatores de um número são `2 2 3 3 4`, mantém só `2 3 4`). Restrição: o algoritmo de eliminar duplicidade só percorrer uma única vez a lista de fatores. Exemplo de chamada (`unique-factors.csv` é o arquivo de saída do filtro):

```
$ ./unique.rkt factors.csv unique-factors.csv
```

Exercício 2

2-a. Construa o programa `random-sorted-lists.rkt <n> <k> <m>` que gere n listas de inteiros aleatórios, no intervalo $[0, k)$, cada lista contendo m elementos. Cada lista deve ser salva em um arquivo separado, com nomes sequenciais, no diretório atual. Por exemplo:

```
$ ./random-sorted-lists.rkt 3 100 10
$ ls
1.csv 2.csv 3.csv
$ cat 1.csv
8 26 27 38 40 50 53 55 69 93
```

2-b. Construa o programa `merge.rkt <dir> <res>` que combine todas as listas ordenadas do diretório <dir> em um única lista no arquivo <res>, a qual também deve estar ordenada. Por exemplo:

```
$ ls data/
1.csv 2.csv
$ cat data/1.csv
46 90 95
$ cat data/2.csv
3 85 92
$ ./merge.rkt data data/res.csv
$ ls data/
1.csv 2.csv res.csv
$ cat res.csv
3 46 85 90 92 95
```