MATA56 - Paradigmas de Linguagens de Programação
Prova 1

Aluno:

1. Você tem 110 minutos para fazer esta prova.
2. Você pode consultar qualquer material que estiver com você, mas não pode acessar a Internet ou consultar qualquer fonte externa, inclusive o seu colega ☺.
3. Você NÃO pode usar computador ou qualquer outro equipamento eletrônico.
4. Você só pode usar funções pré-definidas quando estas forem explicitamente mencionadas nos enunciados das questões, qualquer outra função auxiliar deve ser codificada por você.
5. Você pode usar cláusulas pedidas em outras questões para resolver uma dada questão. Por exemplo, você pode usar as cláusulas pedidas em Q1 para solucionar a questão Q2, mesmo sem ter resolvido Q1.
6. Utilize caneta. Assine a prova e as folhas de resposta logo no início do exame (i.e., AGORA mesmo).
7. Coloque a caneta de lado assim que o professor anunciar o final da prova.
8. Qualquer violação das regras acima implica em nota zero na prova.

Questão 1 (2 pontos) Escreva em Racket a função (**`intercala`** `v1 v2 N`) que produz uma lista de tamanho N intercalando os valores v1 e v2. Por exemplo:

```
> (intercala 'vermelho 'azul 5)
(vermelho azul vermelho azul vermelho)
```

Questão 2 (2 pontos) Escreva em Racket a função (**`empacote`** *`lista`*), que transforma uma lista em uma lista de listas, empacotando elementos iguais consecutivos em sublistas distintas. Por exemplo:

```
> (empacote '(a a a a a b b b b c c c a d e e e e e e))
'((a a a a a) (b b b b) (c c c) (a) (d) (e e e e e e))
```

Questão 3 (2 pontos) Escreva em Racket a função (**`codifica`** `L`), que codifica a lista L em uma lista de pares com os elementos consecutivos de L e o seu número de ocorrências. Por exemplo:

```
(codifica '(a a a a a b b b b c c c a d e e e e e e))
'((a . 5) (b . 4) (c . 3) (a . 1) (d . 1) (e . 6))
```

Você pode assumir a existência da função (`length l`) do Racket.

Questão 4 (2 pontos) Escreva em Racket a função (**`iguaisll`** *`generic-list generic-list`*) que recebe duas listas genéricas e retorna verdadeiro se e somente se elas forem exatamente iguais.

```
> (iguais-ll '((a b c) d (e (f) (g h (i)))) '((a b c) d (e (f) (g h (i)))))
#t
```

Questão 5 (2 pontos) Escreva em Racket a função (**`merge-sort2`** `list [function] [key]`**)** que recebe uma lista de elementos e os ordena, utilizando o algoritmo *merge sort*, uma função extratora de chave para comparação, e uma função de comparação. As funções de comparação e extração de chave são opcionais, nomeadas *f* e *k*, e por default assumem as funções "<" e "*identity*", respectivamente. Veja o exemplo:

```
> (merge-sort2 '(3 4 7 8 10 2 1 9 5 6))
(1 2 3 4 5 6 7 8 9 10)

> (merge-sort2 '((3 4) (7 8 10 2 1) (9 5 6)) #:f >= #:k length)
'((7 8 10 2 1) (9 5 6) (3 4))
```