Lista de Linguagens de Programação – 1

Nome: _____ Matrícula: _____

1. Uma linguagem é chamada *Turing Completa* se ela permite simular, em *software*, um computador tão poderoso quanto a *Máquina de Turing*. Neste curso estudaremos linguagens que são Turing completas. Porém vocês já viram alguns exemplos de linguagens que não são Turing Completas.

   (a) O que uma linguagem precisa ter, para que ela seja considerada Turing Completa?

   (b) Descreva uma linguagem que não seja Turing Completa. Dica: vocês já estudaram linguagens assim no curso *Fundamentos da Teoria da Computação* (FTC).

   (c) Descreva um problema que a linguagem acima não conseguiria resolver.

   (d) HTML é uma linguagem Turing Completa? E SQL?

2. Brain Fuc*, projetada por Urban Müller em 1993, é talvez a menor linguagem de programação considerada Turing Completa. Brain Fuc* contém oito comandos diferentes, que manipulam o estado de uma fita. A fita, que simula a memória, consiste de uma sequência de células contendo números inteiros. Neste exercício você vai fazer um pouco de Brain Fuc*ing. Embora não seja parte do exercício, um interpretador de Brain Fuc* não usa em torno de uns 200 caracteres. Talvez você queira escrever um.

   (a) Liste os oito comandos (símbolo mais descrição curta) usados em Brain Fuc*.

   (b) Escreva um programa em Brain Fuc* que zere o conteúdo da célula corrente.

   (c) Escreva um programa em Brain Fuc* que copie o conteúdo da célula corrente para a célula à esquerda.

   (d) Escreva um programa que inicialize a célula corrente com o número três, mova o cursor para a próxima célula, inicialize esta posição com o número quatro, e então some o conteúdo das duas células, colocando o resultado na segunda célula.

   (e) Escreva um programa que leia a entrada, copiando os caracteres lidos para a fita, e então percorra a fita, imprimindo os caracteres lidos na ordem em que eles foram informados.

3. As linguagens de programação podem ser divididas em dois grupos principais: as linguagens *imperativas* e as linguagens *declarativas*.

   (a) Algumas linguagens favorecem a programação declarativa, enquanto outras favorecem a programação imperativa. Por outro lado, pode-se programar declarativamente ou imperativamente na maioria das linguagens de programação. Escolha um certo algoritmo, com o qual você esteja familiar, e o escreva em pseudo-código, de forma imperativa e declarativa.

   (b) Descreva uma vantagem da abordagem declarativa sobre a abordagem imperativa. Não é preciso ater-se ao exemplo da questão anterior.

   (c) Agora faça o oposto: mostre uma vantagem da abordagem imperativa sobre a sua contra-parte declarativa.

4. Um *Quine* (leia quáine) é um programa que produz uma cópia do próprio código fonte como a única saída. Um resultado importante em computação é que qualquer linguagem Turing completa permite a escrita de quines. Note que um quine que é um quine mesmo não recebe nenhuma entrada.

   (a) Escreva um quine em sua linguagem de programação favorita.

   (b) Escreva um quine em Bash. Note que o seguinte programa não é um quine, pois ele lê seu próprio código da entrada:

   ```
   #!/bin/sh
   cat $0
   ```

   (c) Você conseguiria escrever um quine em Português? Bom, Português não é necessariamente "executável", mas podemos sempre usar um pouquinho de imaginação... escreva um texto que mande seu melhor amigo escrever um texto, por exemplo :)