INF 05008 - Fundamentos de Algoritmos 2020/2
Lista de exercícios 3
Expressões Condicionais e Expressões Simbólicas (Capítulos 4 e 5)

Instruções:

- Escreva todas as expressões e funções pedidas nos exercícios no ambiente de sua preferência (*DrRacket* ou *WeScheme*).
- Salve seu programa em um arquivo e envie pelo *Moodle*. O nome do arquivo deve ser do tipo:

  lista3-NomeDoAluno-TurmaDoAluno.rkt

  onde em NomeDoAluno você deve colocar o seu nome e em TurmaDoAluno você deve colocar a sua turma (A, B, C ou D). **Não coloque espaços nem acentos no nome do arquivo**.
- Preste atenção no **nome das funções** e na **ordem dos argumentos** pedidos nos exercícios.
- Para facilitar, utilize o **modelo** de arquivo fornecido no *Moodle*.
- Em todas as questões, **deve ser colocada a documentação completa**, ou seja, contrato, objetivo, exemplos e testes.
- Nesta lista, vocês devem colocar um teste para cada caso possível do bloco condicional. Por exemplo, se uma função possui 5 casos possíveis no bloco condicional, você deve ter no mínimo 5 testes daquela função, um para cada caso.
- Nesta lista, todas as vezes que um bloco condicional for utilizado, você deve descrever em português como comentário cada um dos casos. Por exemplo:

```
(cond
    ;; se n for maior que 10, soma n com um
    [(> n 10).  (+ n 1)]
    ;; se n for menor ou igual a 10, multiplica n por dois
    [(<= n 10) (* n 2)])
```

- Nas questões que retornam uma string, preste bem atenção na string que deve ser retornada. Tudo deve estar igual ao pedido (pontuação, acentos, letras maiúsculas, etc)
- É sempre bom você quebrar os problemas maiores em problemas menores. Isso deixa seu código mais elegante e legível, além de facilitar a resolução do problema.
- Esta lista possui um exercício extra. Ele vale pontos extras para esta lista (não acumula para as próximas).
- Fique atento ao **prazo** para a entrega da lista.

Exercícios

Exercício 1. Imagine que você está programando um jogo. Nesta etapa você está programando a interação com o teclado. Seu personagem pode andar pelo mundo utilizando as clássicas teclas *W, A, S e D*. Implemente uma função chamada `movimenta_personagem` que recebe como argumento um símbolo e retorna uma string que representa a ação que ele deveria realizar, conforme descrito na tabela abaixo:

Símbolo de Entrada	String de saída
'w	``Andando para frente''
'a	``Andando para esquerda''
's	``Andando para tras''
'd	``Andando para direita''
*	``Parado''

O asterisco na tabela representa qualquer outro símbolo que não esteja na tabela.

Obs.: Para comparar os símbolos nesta questão, utilize a função `symbol=?`

Exercício 2. Escreve uma função chamada `esta_dentro`, que, dada uma circunferência (seu centro e seu raio) e um ponto no plano cartesiano, diz se o ponto está dentro da circunferência ou não (`#true` se estiver dentro e `#false` caso contrário).

Obs.: Se o ponto estiver exatamente na borda da circunferência, a saída deverá ser `#true`.

Dica 1: Como um ponto no plano cartesiano possui duas coordenadas, cada ponto é representado por dois números na entrada. Logo, a função terá 5 argumentos.

Dica 2: Para saber se um ponto está dentro de uma circunferência, você pode calcular a distância do ponto até o centro da circunferência. Se a distância for maior que o raio, então o ponto está fora. Caso contrário, o ponto está dentro.

Exercício 3. A função anterior pode ser codificada utilizando ou não a função `cond`. Se você fez utilizando este comando, tente agora fazer sem utilizá-lo. Se você não utilizou, tente agora fazer utilizando. Dê o nome dessa nova função de `esta_dentro_v2`.

Depois de codificada a função, responda: se eu quisesse que, ao invés de um booleano, a saída fosse uma string, do tipo: "O ponto está dentro da circunferência" ou "O ponto está fora da circunferência", ainda daria pra codificar a função das duas maneiras? Por quê?

Obs.: Dê a resposta escrita em um comentário no seu código, logo abaixo do código desta questão.

Exercício 4. Suponha que você tenha três contas para pagar no mês e deseja saber se seu salário será suficiente para pagar tudo. Escreva uma função chamada `gastos_mes` que, dado seu salário, o aluguel, conta de luz e de internet, retorna uma string dizendo quantas contas foram possíveis pagar.

Faça uma função auxiliar para gerar a string de saída a partir de um número de entrada. A saída deve estar no seguinte formato, de acordo com a quantidade de contas pagas:

```
Neste mês, foi possível pagar 3 conta(s).
Neste mês, foi possível pagar 2 conta(s).
Neste mês, foi possível pagar 1 conta(s).
Neste mês, foi possível pagar 0 conta(s).
```

Obs.: Seu programa deve **maximizar a quantidade de contas pagas**, independente do valor de cada uma delas.

Ex.: `(gastos_mes 1000 500 900 300) -> ‘‘Neste mês, foi possível pagar 2 contas.’’`

Exercício 5. Codifique a função `max_tres` que, dados três números, retorna o maior deles. O uso das funções booleanas `and`, `or` e `not` está proibido neste exercício. Utilize blocos condicionais aninhados (um dentro o outro) para resolver.

Obs.: Para resolver este exercício, compare os números utilizando as funções $>, <, \geq, \leq$. Não utilize funções de `max` e `min`.

Exercício extra. Suponha que seu você está cansado de operar na bolsa e quer automatizar o processo de compra e venda de ações, baseado no quanto a ação variou no dia. Depois de anos, você chegou à um conjunto de regras que pensa ser bom o suficiente para o trabalho:

- Se a ação subiu até 10%, não faz nada.
- Se a ação subiu entre 10% e 100%, compra X ações.
- Se a ação subiu mais de 100%, vende todas as ações.
- Se a ação caiu até 20%, não faz nada.
- Se a ação caiu mais de 20%, vende Y ações.

Onde X e Y são calculados pelas seguintes fórmulas:

$$X = poder\_total\_compra \cdot porcentagem\_subiu$$
$$Y = qtde\_acoes \cdot porcentagem\_caiu$$

onde *poder_total_compra* é a quantidade máxima de ações possíveis de comprar com o dinheiro disponível (pode ser um número não inteiro), *qtde_acoes* é a quantidade de ações que você possui atualmente. As porcentagens na fórmula estão na escala de 0 a 1.

Escreva uma função chamada `opera_na_bolsa` que recebe o preço de abertura de uma ação, o preço atual dela, a quantidade de ações possuídas atualmente e uma quantidade de dinheiro disponível para gastar nesta ação; e retorna a quantidade de ações compradas ou vendidas,

de acordo com as regras descritas. Números positivos significam ações compradas e negativos, ações vendidas.

Obs.: Para os pontos extremos, utilize a seguinte definição matemática para os intervalos das regras: $[0, 10)$ $[10, 100)$ $[100, \infty)$ $[-20, 0)$ $(-\infty, -20)$

Obs.: Não existe compra ou venda de frações de uma ação