Exame Especial de Linguagens de Programação
- DCC024 -

Nome: _____

"Eu dou minha palavra de honra que não trapacearei neste exame."

Número de matrícula: _____

As regras do jogo:

- A prova é sem consulta.
- Quando terminar, não entregue nada além do caderno de provas para o instrutor.
- Quando escrever código, a sintaxe correta é importante.
- Cada estudante tem direito a fazer uma pergunta ao instrutor durante a prova. Traga o caderno de provas quando vier à mesa do instrutor.
- A prova termina uma hora e quarenta minutos após seu início.

Alguns conselhos:

- Escreva sempre algo nas questões, a fim de ganhar algum crédito parcial.
- Se não entender a questão, e já tiver gasto sua pergunta, escreva a sua interpretação da questão junto à resposta.
- Serão avaliadas somente as sete melhores respostas. Então sinta-se livre para abandonar alguma questão devido ao tempo.
- A prova não é difícil, ela é divertida, então aproveite!

1. Podemos representar uma árvore binária em SML usando o tipo algébrico abaixo:

```
datatype 'a tree = Leaf | Node of 'a tree * 'a * 'a tree
```

Um nodo `Leaf` é um sentinela usado para indicar que chegamos ao fim de um caminho na árvore; já um nodo `Node` contém uma sub-árvore à direita, um item de dado, e uma sub-árvore a esquerda. Nesta questão você deve implementar uma função `revTree`, cujo tipo é `'a tree -> 'a tree`.

```
-revTree (Node(Node(Leaf, 1, Leaf), 2, Node(Leaf, 3, Leaf)));
val it = Node(Node(Leaf, 3, Leaf), 2, Node(Leaf, 1, Leaf)) : int tree
```

Uma dica para ajudar a relembrar a sintaxe de SML: a função abaixo determina se uma árvore é completa, isto é, ela verifica se a árvore não contém nodos em que um dos filhos seja outra árvore e o outro seja uma folha.

```
datatype 'a tree = Leaf | Node of 'd tree * 'd * 'd tree

fun isComplete Leaf = true
  | isComplete (Node (Leaf, _, Leaf)) = true
  | isComplete (Node (Leaf, _, Node(_, _, _))) = false
  | isComplete (Node (Node(_, _, _), _, Leaf)) = false
  | isComplete (Node (l, _, r)) = isComplete l andalso isComplete r
```

2. Considere a classe abaixo:

```
class MyInt {
  int i;
  MyInt(int k) {i = k;}
  void swap1(MyInt j) {
    MyInt tmp = j;
    j = new MyInt(i);
    i = tmp.i;
  }
  void swap2(MyInt j) {
    MyInt tmp = j;
    j.i = i;
    i = tmp.i;
  }
  void swap3(int j) {
    int tmp = j;
    j = i;
    i = tmp;
  }
}
```

Cada uma das próximas questões é completamente independente uma das outras. Estas questões devem ser respondidas com base nas definições abaixo:

```
MyInt m1 = new MyInt(3);
MyInt m2 = new MyInt(4);
```

(a) Qual é o valor de `m1.i` e `m2.i` depois da chamada `m1.swap1(m2)`?

(b) Qual é o valor de `m1.i` e `m2.i` depois da chamada `m1.swap2(m2)`?

(c) Qual é o valor de `m1.i` e `m2.i` depois da chamada `m1.swap3(m2)`?

(d) Qual é a política [1] que java adota para passar tipos primitivos (`int, float, char`, etc) como parâmetros de métodos?

(e) Qual é a política que java adota para passar objetos como parâmetros de métodos?

[1] As políticas existentes são passagem por: valor, referência, nome, expansão de matro, retorno, valor-retorno e avaliação preguiçosa

3. Cada uma das questões a seguir diz respeito ao programa abaixo:

```
class Wrapper<E> {
  private E o;
  Wraper() {this.o = null;}
  Wrapper(E o) {this.o = o;}
  E get() {return o;}
}
public class Test {
  public static void main(String a[]) {
    Wrapper<String> w = new Wrapper<String>();
    System.out.println(w.get().toString());
  }
}
```

(a) Que tipo de erro será causado pelo programa acima?

(b) Defina uma classe de exceção para representar este erro.

(c) Modifique o método `get` para disparar a exceção criada anteriormente.

(d) Modifique o método `main` para tratar esta exceção.

4. O problema do subconjunto de soma N é um problema NP-completo clássico. Dado um conjunto U de inteiros, e um número inteiro N, o problema pede que seja encontrado um subconjunto S de U cuja soma seja N. Por exemplo, caso $U = \{1, 2, 3, 4, 5\}$ e $N = 6$, temos $S = \{1, 2, 3\}, S = \{1, 5\}$ e $S = \{2, 4\}$. Dado que o problema do subconjunto de soma N é NP-completo, pouca esperança existe de que exista uma solução polinomial para ele. Logo, algoritmos que resolvem este problema baseiam-se em buscas exponenciais.

   Implemente este algoritmo em Prolog.

5. (a) Qual o uso de gramáticas em linguagens de programação?

(b) Mostre que a gramática abaixo é ambígua:

```
<s> ::= <s> <s>
     |  '(' <s> ')'
     |  '(' ')'
```

(c) Escreva uma gramática para o conjunto de todos os *strings* formados por uma seqüência de `a`'s com um ponto e vírgula entre eles. Por exemplo, `a;a;a` pertence a esta linguagem, porém `a;a;` não pertence.

(d) Escreva uma gramática para termos compostos em Prolog, como por exemplo `soma(L, S)` ou `parent(james, barbara)`. Assuma que um termo contenha somente átomos ou variáveis em seu interior. Você não precisa definir átomos e variáveis. Apenas assuma que os não-terminais `atom` e `var` já foram definidos.

6. Consider a definição da função `sum` abaixo:

```
fun sum nil = 0
  | sum (h::t) = h + sum t;
```

(a) Qual o tipo desta função?

(b) Redefina a função `sum`, de forma que a nova definição use `foldr` ou `foldl` e tenha somente uma linha.

(c) Qual a complexidade, em termos de tempo e espaço em memória, da função `sum` definida no enunciado desta questão?

(d) Redefina a função `sum` para que ela passe a ser uma função de *cauda rasa*.

(e) Qual a complexidade, em termos de tempo e espaço em memória, da função `sum` que você acabou de redefinir?

7. Considere uma linguagem desconhecida, com tipos primitivos inteiro e *string* tal que as seguintes igualdades sejam válidas:

   (a) `1 + 2 * 3 = 7`

   (b) `"1"+ "2"+ "3"= "123"`

   (c) `"1"+ 2 + 3 = "123"`

   (d) `1 + "2*3"` é um erro de tipo.

   Descreva um sistema de precedência, associatividade, sobrecarga e coerção implícita que suporte esta linguagem. Neste sistema, qual o resultado da avaliação de `"1"+ 2*3`?

8. Considere a classe `Reuse` abaixo:

```
1    class Reuse {
2      Reuse Reuse (Reuse Reuse) {
3        Reuse:
4        for (;;) {
5          if (Reuse.Reuse(reuse) == Reuse)
6            break Reuse;
7        }
8        return Reuse;
9      }
10    }
```

(a) A definição de `Reuse` na linha 3 é um rótulo. Caso o comando da linha 6 seja executado, então o fluxo do programa será desviado para a linha 3. Qual o significado de cada uma dentre as outras definições do nome `Reuse`?

(b) A ocorrência de `Reuse` na linha 6 é definida na linha 3. Para cada outra ocorrência da palavra `Reuse`, *que não seja uma definição*, mostre onde este nome é definido.