Compiladores – Prova #1

Nome:

Matrícula: Data:

Observações:

(a) A prova é individual e sem consulta, sendo vedado o uso de calculadoras e de telefones celulares.

(b) A interpretação dos comandos das questões faz parte da avaliação.

(c) A nota da prova será igual a 0 (zero) caso o estudante consulte algum material durante a prova, ou receba ou ofereça qualquer ajuda a outro estudante durante a prova.

(d) As questões podem ser resolvidas a lápis ou à caneta. Entretando, a resposta final deve ser **destacada** de forma clara (circulada, sublinhada, reforçada, indicada, etc...) para que a questão seja devidamente corrigida.

(e) O grampo da prova **não deve ser removido**. Caso o grampo seja removido, a nota da prova será igual a 0 (zero).

Parte A

1. (7 pontos) Complete a sentença: "A compilação pode ser dividida em _____ fases, as quais podem ser classificadas em duas partes: _____ e _____. Outra organização possível separa as _____ primeiras fases na interface de _____ e as _____ últimas fases na interface de _____."

2. Considere a gramática livre de contexto G abaixo:

$$S \rightarrow S \textbf{ concat } L \mid S \textbf{ replace } L\ L \mid \epsilon$$
$$L \rightarrow \texttt{a} \mid \texttt{b} \mid \texttt{c}$$

De acordo com as convenções de notação, responda as questões abaixo?

(i) (2 pontos) Quais são os não-terminais de G?

(ii) (2 pontos) Quais são os terminais de G?

(iii) (1 ponto) Qual é o símbolo de partida de G?

3. (8 pontos) **Assinale a alternativa correta.** Considere o seguinte código de máquina de pilha:

```
value-l a
push    5
push    2
ϕ
value-r b
⁖
```

Marque a expressão, em notação infixada, que é avaliada por este código. Assuma que as operações ϕ e ⁖ sejam binárias e que o ordem de extração dos operando da pilha seja a seguinte: primeiro o operando à direita, em seguida o operando à esquerda.

(A) `a := (5 ϕ 2) ⁖ b`

(B) `a := b ⁖ (2 ϕ 5)`

(C) `a := (b ⁖ 2) ϕ 5`

(D) `a := 5 ϕ (2 ⁖ b)`

4. (10 pontos) **Julgue os itens abaixo.** Em cada item, preencha os parêntesis com V (verdadeiro) ou F (falso).

( ) Em uma produção, o símbolo não-terminal que será produzido fica do lado direito da seta.

( ) O símbolo ϵ representa uma cadeia de tokens vazia.

( ) Uma gramática livre de contexto é ambígua se existe ao menos uma expressão que não possui árvore sintática.

( ) Em uma definição dirigida pela sintaxe, um atributo de um nó n é dito sintetizado se ele depende apenas dos valores dos atributos das folhas da árvore.

( ) Em um esquema de tradução, as ações semânticas são inseridas no lado direito da produção e são delimitadas por chaves.

Parte B

5. (15 pontos) Na linguagem de programação C o enunciado `do-while` possui a forma

```
do { cmd } while ( expr );
```

O laço inicia executando `cmd`. Em seguida, `expr` é avaliada: caso seja verdadeira, o laço reinicia com uma nova execução de `cmd`; caso contrário, o laço é encerrado. O significado do enunciado `do-while` é similar a

```
cmd ; while ( expr ) { cmd }
```

Construa um gabarito para a tradução dirigida pela sintaxe que traduz enunciados `do-while` em C para código de máquina de pilha.

6. (15 pontos) Considere a seguinte gramática, que gera sequências regulares de parêntesis:

$$S \to (\; S \;) \; S \mid \epsilon$$

Construa a árvore sintática para a sequência `(())()`.

7. Em relação aos analisadores gramaticais recursivos descendentes:

(i.) (8 pontos) Defina uma gramática não-ambígua que gere expressões formadas pelos valores lógicos `t` e `f` (verdadeiro e falso, respectivamente) e a conjunção (`^`). Assuma que a conjunção é uma operação binária associativa à direita.

(ii.) (12 pontos) Construa um analisador gramatical recursivo descendente para a gramática definida no item anterior.

Parte C

8. Uma constante inteira em base binária (token `NUM`) é uma cadeia não-vazia formada pelos caracteres `0` e `1` sendo que, exceto pelo número zero, nenhuma outra cadeia deve iniciar com o caractere zero. Um identificador (token `ID`) é uma cadeia não-vazia composta pelos caracteres `a` e `b`, nas quais dois caracteres consecutivos devem ser sempre distintos.

Considere a seguinte implementação de um analisador léxico que identifica este tokens. Assuma

(a) que o atributo de `NUM` é o seu valor em base decimal (por exemplo, para o lexema `101` geraria um token `NUM` cujo atributo seria igual a `5`),

(b) que o atributo de `ID` seja o número de caracteres que compõem o lexema (por exemplo, o lexema `abab` geraria um token `ID` cujo atributo seria igual a `4`),

(c) que a função `erro()` esteja devidamente implementada,

(d) que os tokens foram devidamente declarados, e

(e) que o código esteja sintaticamente correto segundo a linguagem C++.

```cpp
using token = std::pair<int, int>;

token_t scanner()
{
    while (not std::cin.eof())
    {
        auto c = std::cin.get();

        if (c == '0' or c == '1')
        {
            int valor = 0;

            while ((not std::cin.eof()) and (c = std::cin.get(), c == '0' or c == '1'))
            {
                valor *= 10;
                valor += c - '0';
            }

            std::cin.unget();

            return { NUM, valor };
        } else if (isalpha(c))
        {
            int len = 1, prev = c;

            while ((not std::cin.eof()) and (c = std::cin.get(), c == 'a' or c == 'b')
            {
                if (c == prev)
                    break;

                ++len, prev = c;
            }

            std::cin.unget();

            return { ID, len };
        } else
            erro();
    }

    return { EOF, -1 };
}
```

(i.) (`24 pontos`) Identifique três erros semânticos presentes nesta implementação. Para cada erro, indique o número da linha onde ele ocorre, descreva e justifique o erro e proponha uma correção para este erro. Esta proposição pode ser feita de forma descritiva, sem necessariamente mostrar o código C++ correspondente à correção.

(ii.) (`6 pontos`) Forneça uma cadeia de caracteres que a implementação acima reconheceria como um token válido mas que viola uma ou mais dentre as regras de formação de tokens apresentadas.

Nome:

Matrícula: Data:

Nome:

Matrícula: Data:

Nome:

Matrícula: Data:
