



Laboratório de Linguagens de Programação  
Prof. Andrei Rimsa Álvares

# Trabalho Prático I

## 1. Objetivo

O objetivo desse trabalho é desenvolver um interpretador para uma nova linguagem de programação que embute ideias de programação orientada a objetos: *SmallOO*. Ela possui quatro (4) tipos que podem ser indexados: inteiros, string, objetos e funções. Objetos possuem propriedades que podem armazenar qualquer um dos tipos disponíveis. Funções possuem retorno e podem ser chamadas com qualquer quantidade de argumentos.

## 2. Contextualização

A seguir é dado um exemplo de utilização da linguagem que simula um objeto que mantém informações de um arranjo de inteiros:

---

```
/* iniciar o array com nenhum elemento e tamanho 0. */
array.size = 0;

/* adicionar um novo elemento ao arranjo. */
array.add = function {
    system.set(self, ("element" + self.size), args.arg1);
    array.size = array.size + 1;
}

/* obter um elemento através de seu índice. */
array.get = function {
    return system.get(self, ("element" + args.arg1));
}

/* imprimir o arranjo. */
array.print = function {
    system.print("[ ");
    i = 0;
    while (i < self.size) {
        system.print(system.get(self, ("element" + i)) + " ");
        i = i + 1;
    }
    system.println("]");
}

/* entrar c/ numeros no arranjo */
n = system.input("Entre c/ a qtd de numeros: ");
c = 0;
while (c < n) {
    x = system.random(0, 9);
    array.add(x);
    c = c + 1;
}

/* criar uma copia do arranjo. */
array2 = array;

/* adicionar mais um numero aleatorio a ele. */
y = system.random(0, 9);
array2.add(y);
```



## Laboratório de Linguagens de Programação

### Prof. Andrei Rimsa Álvares

```
/* adicionar o método ordenar no novo arranjo utilizando o metodo bolha. */
array2.sort = function {
    a = 0;
    while (a < self.size) {
        b = a + 1;
        while (b < self.size) {
            x = system.get(self, ("element" + a));
            y = system.get(self, ("element" + b));
            if (y < x) {
                system.set(self, ("element" + a), y);
                system.set(self, ("element" + b), x);
            }
            b = b + 1;
        }
        a = a + 1;
    }
}

/* imprimir o arranjo original */
array.print();

/* ordenar o novo arranjo e depois imprimir. */
array2.sort();
array2.print();
```

---

*array.soo*

Um programa em *SmallOO* possui variáveis e propriedades de objetos que conseguem armazenar tipos primitivos (inteiros), outros objetos e funções. Todo tipo é atribuído ou passado por parâmetros por **cópia**. A linguagem possui escopo global para as variáveis. Funções dentro de objetos podem referenciar suas próprias propriedades (atributos ou métodos) através da palavra reservada **self**, e acessar os parâmetros passados na chamada através da palavra reservada **args**. Existe um objeto especial chamado **system** que possui as funções especiais:

- **print(msg)**: imprime msg (opcional) na tela, onde msg deve ser inteiro ou string.
- **println(msg)**: similar a função print, mas com nova linha.
- **read(msg)**: imprime msg (opcional) como print e retorna um número inteiro ou string obtido do teclado.
- **random(n, m)**: retorna um número aleatório inteiro entre n e m (incluso), onde n e m são inteiros e n deve ser maior que m.
- **get(obj, name)**: obtém o valor da propriedade name do objeto obj, onde obj deve ser um objeto e name uma string. Pode retornar qualquer um dos quatro tipos.
- **set(obj, name, value)**: define valor da propriedade name do objeto obj com o valor value, onde obj deve ser um objeto, name uma string e value qualquer um dos quatro tipos.
- **abort(msg)**: imprime msg (opcional) como print e aborta a execução.
- **type(x)**: retorna o tipo de x como uma string.
- **length(str)**: retorna o tamanho da string str, zero caso contrário.
- **substring(str, i, f)**: retorna a substring de str a partir de i de tamanho f.



Laboratório de Linguagens de Programação  
Prof. Andrei Rimsa Álvares

Operações aritméticas só podem ser feitas em inteiros, a não ser o operador + que pode ser usado para concatenar duas strings. A linguagem possui comentários de múltiplas linhas onde são ignorados qualquer sequência de caracteres entre /\* e \*/. A linguagem possui as seguintes características:

**1) Comandos:**

- a. **if**: executar comandos baseado em expressões condicionais.
- b. **while**: repetir comandos enquanto a expressão for verdadeira.
- c. **atribuição**: guardar um valor em uma variável.
- d. **chamada**: chamada de funções.

**2) Valores:**

- a. **Inteiro**: sequência de dígitos.
- b. **String**: uma sequência de caracteres entre aspas duplas.
- c. **Objetos**: armazenam propriedades que podem ser atributos (inteiros, strings e outros objetos) e métodos (funções).
- d. **Funções**: procedimentos que podem tomar opcionalmente qualquer quantidade de argumentos e retornam um valor. Caso não seja especificado deve-se retornar 0.
- e. **Lógico**: operações de comparações que obtém um valor lógico (não podem ser armazenados).

**3) Operadores:**

- a. **Inteiro**: + (adição), - (subtração), \* (multiplicação), / (divisão), % (resto inteiro).
- b. **String**: + (concatenação)
- c. **Lógico**: == (igual), != (diferença), < (menor), > (maior), <= (menor igual), >= (maior igual), ! (negação).
- d. **Conector**: & (E lógico), | (OU lógico).

### 3. Gramática

A gramática da linguagem *SmallOO* é dada a seguir no formato de Backus-Naur estendida (EBNF):

```
<code>      ::= { <statement> }
<statement> ::= <if> | <while> | <cmd>
<if>        ::= if '(' <boolexpr> ')' '{' <code> '}' [else '{' <code> '}' ]
<while>     ::= while '(' <boolexpr> ')' '{' <code> '}'
<cmd>       ::= <access> ( <assign> | <call> ) ';'
<access>    ::= <var> { '.' <name> }
<assign>    ::= '=' <rhs>
<call>      ::= '(' [ <rhs> { ',' <rhs> } ] ')'

<boolexpr>  ::= [ '!' ] <cmpexpr> [ ('&' | '!' ) <boolexpr> ]
<cmpexpr>   ::= <expr> <relop> <expr>
<relop>     ::= '==' | '!=' | '<' | '>' | '<=' | '>='
```



Laboratório de Linguagens de Programação  
Prof. Andrei Rimsa Álvares

```
<rhs>      ::= <function> | <expr>
<function> ::= function '{' <code> [ return <rhs> ] '}'
<expr>     ::= <term> { ('+' | '-') <term> }
<term>     ::= <factor> { ('*' | '/' | '%') <factor> }
<factor>   ::= <number> | <string> | <access> [ <call> ]

<var>      ::= system | self | args | <name>
```

## 4. Instruções

Deve ser desenvolvido um interpretador em linha de comando que recebe um programa-fonte na linguagem *Small00* como argumento e executa os comandos especificados pelo programa. Por exemplo, para o programa *array.soo* deve-se produzir uma saída semelhante a:

```
$ ./sooi
Usage: ./sooi [Small00 File]
$ ./sooi array.soo
Entre c/ a qtd de numeros: 5
[ 1, 9, 3, 7, 2 ]
[ 1, 2, 3, 6, 7, 9 ]
```

O programa deverá abortar sua execução, em caso de qualquer erro léxico, sintático ou semântico, indicando uma mensagem de erro. As mensagens são padronizadas indicando o número da linha (2 dígitos) onde ocorreram:

| Tipo de Erro | Mensagem                              |
|--------------|---------------------------------------|
| Léxico       | Lexema inválido [ <i>lexema</i> ]     |
|              | Fim de arquivo inesperado             |
| Sintático    | Lexema não esperado [ <i>lexema</i> ] |
|              | Fim de arquivo inesperado             |
| Semântico    | Operação inválida                     |

Exemplo de mensagem de erro:

```
$ ./sooi erro.msh
03: Lexema não esperado [;]
```

## 5. Avaliação

O trabalho deve ser feito em grupo de até dois alunos, sendo esse limite superior estrito. O trabalho será avaliado em 15 pontos, onde essa nota será multiplicada por um fator entre 0.0 e 1.0 para compor a nota de cada aluno individualmente. Esse fator poderá estar condicionado a apresentações presenciais a critério do professor.

Laboratório de Linguagens de Programação  
Prof. Andrei Rimsa Álvares

Trabalhos copiados, parcialmente ou integralmente, serão avaliados com nota **ZERO**, sem direito a contestação. Você é responsável pela segurança de seu código, não podendo alegar que outro grupo o utilizou sem o seu consentimento.

## **6. Submissão**

O trabalho deverá ser submetido até as 23:55 do dia 23/04/2018 (segunda-feira) via sistema acadêmico em pasta específica. Não serão aceitos, em hipótese alguma, trabalhos enviados por e-mail ou por quaisquer outras fontes.

