UNIVERSIDADE FEDERAL DA FRONTEIRA SUL (UFFS)

MAMIKAMI

Harold c. s. Becker
Leonado Silva

Chapecó
2015

Apresentação da Sintaxe da linguagem

A linguagem mamikami fornece uma sintaxe flexível, sendo possível ter espaçamento e quebras de linha em qualquer parte do código e não existe cometários. O código principal deve ser escrito dentro da função main sem conter parâmetros e dentro de um escopo, não é possível criar funções completas fora da main. Os 2 tipos de variáveis implementados são Double e String, sendo que não existe declaração de variáveis o interpretador irá fazer isso automaticamente, é necessário apenas atribuir um valor senão irá retornar uma exceção. É possível criar um encadeamento de ifs, seus parâmetros são dois valores e um operador ou apenas true ou false, sendo que o mesmo vale para o loop.

Declaração de Variáveis e Operações

A declaração de variáveis na linguagem mamikami é bem simples de ser usada, é necessário simplesmente informar o nome da variável e seu respectivo valor para ser definido o seu tipo, seguindo a sintaxe: Nome_variável = Valor, considerando que não é possível atribuir strings nas variáveis, também é possível atribuir a uma variável a outra variável na hora se sua criação, inclusive é possível mudar o tipo da variável em tempo de execução, considerando que a linguagem é case sensitive, ou seja, variáveis com letras minúsculas são totalmente diferentes de variáveis com o mesmo nome com letras maiúsculas.

As operações suportadas pela linguagem mamikami é a multiplicação, subtração, soma, divisão e módulo. Essas operações podem ser usadas no momento de instanciação das variáveis, considerando que existe precedência de operadores, e também é possível demarcar essa precedência usando parênteses.

Exemplos :

```
ex =  12;  //será convertido para 12.0 pois só existe o tipo double e não existe inteiro.

EX                  =                   17.0        ;

abobrinha
=
27.43
;

oi    =                                 44.0;

ex = oi;

ex = EX + 32;

Oi  =  oi / 2;

Oi = abobrinha - 7.43;

Oi =      2 * 2;

abobrinha = 2 % 2;
```

Funções de Entrada, Saída e Função Main

Para iniciar a programação na linguagem mamikami é preciso declarar um escopo chamado main(){}, é dentro dela que deve ser escrito todos os comandos que precisam ser executados, sua sintaxe é bem simples é apenas a palavra main, sem nenhum tipo de retorno, seguida de parênteses sem conter parâmetros e as chaves que irão conter o código.

Outros comandos úteis para a programação na linguagem são os de entrada e saída de dados, sendo possível escolher entre dois tipos de comando de saída e um de entrada, os comandos de saída são descritos de uma forma simples, o nome dos dois comandos de saída são printi (com "i" no final) e printiln(também contendo um "i" antes do "ln"), a diferença entre os dois tipos de saída é que o printiln irá fazer uma quebra de linha no final de todos os parâmetros passados, sendo possível realizar concatenções entre strings e valores.

O comando de entrada é usado para ler o que o usuário digitar no teclado, o escaneador tem como nome : escaneadordofuturo(), seus parâmetros contém apenas uma variável, essa variável pode ser criada diretamente dentro do escaneador sem fazer uma atribuição, ou usar uma variável que já foi declarada, tendo assim sua sintaxe definida da forma : escanadordofuturo(Variavel).

Exemplos:

```
main() {
        //comandos
}

main(){
        printi("olá mundo");
}

main(){
        printiln("olá mundo");
}

main(){
        printiln("olá" + "          mundo "          + ''1'');
        // irá imprimir os espaços que estiverem dentro das aspas e desconsiderará os
        que //estão fora
}

main(){
        escaneadordofuturo(i);
        printiln(i);
}
```

Condicionais

Existe uma condicional implementada que é o ifi (com “ i ” no final), seus parâmetros são dois valores e um operador lógico e seu retorno é um booleano que indica se a condição é verdadeira ou falsa, é possível se utilizar apenas do true ou false como parâmetro. Todo código que deve ser executado só se a condição for verdadeira deve estar entra chaves ({ }), tendo sua sintaxe definida dessa maneira: ifi(Nome_variavel Operador nome_variavel) {COMANDOS},caso seja necessário mais de uma decisão pode ser usado o comando elsi (com “ i ” no final) logo após o ifi e deve estar entre chaves({ }), os operadores que podem ser utilizados são os operadores “>”, “<”, “>=”, “<=”, “==”, “!”, “!=”, além de ser possível aninhar comandos usando as condicionais lógicas E(&&) e OU(||).

Exemplos:

```
main(){
        escaneadordofuturo(b);
        a =b    +4;
        ifi(a > b ){
                printiln(a);
        }elsi{
                printi (b);
        }
}

main(){
        b = 2;
        a = 3;
        ifi(a >3){
                printiln(a);
        }elsi{
                printiln(b);
        }

}
```

Laços de Repetição

Está implementada duas condições de repetição, os nomes deles são uaiou(while) e quatro(for), seus parâmetros são todos iguais ao das condicionais, logo é possível fazer testes da mesma maneira, porém com a diferença de que ele irá repetir a condição até que seja falsa. A linguagem permite fazer comando aninhados, ou seja, criar várias condicionais e laços dentro do laço de repetição principal.

Exemplo :

```
main() {
   i = 0;
   j = 1;
   uaiou(i < 3) {
         ifi(i == 0) {
               printiln("Zero");
         }
         ifi(i != 0) {
               printiln("Outro");
         }
         i = i + 1;
   }
   ifi(i == 3) {
         ifi(j == 0) {
               printiln("ok!");
         }
         ifi(j != 0) {
               printiln("erro");
         }
   }
}
```

```
main() {
         quatro(i = 10; i < 20; i = i + 1) {
               printiln(i);
         }
 }
```