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, ainda 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 while.

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 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 e divisão. Essas operações podem ser usadas no momento de instanciação das variáveis.

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;
```

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 te como nome : escaneadordofuturo(), seus parâmetros contém apenas uma variável, essa variável pode ser criada diretamente dentro do escaneador sem fazer uma atribuição, tendo assim sua sintaxe definida da forma : escanadordofuturo(Variavel_ainda_não_criada).

Exemplos:

```
main() {
        comandos
}

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

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

main(){
        printiln("olá" + "          mundo       " + ''1'');// irá imprimir os espaços
}

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, e 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};

Exemplos:

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