**Tipos de datos primitivos**

ent (entero de 32 bits)

carc (Unicode de 16 bits)

logico (true, false)

**Operadores aritméticos**

+ suma

- resta

\* multiplicación

/ división

% módulo

++ incremento en 1

-- decremento en 1

**Operadores aritméticos**

= asignación

+= asignación y suma

-= asignación y resta

/= asignación y división

\*= asignación y multiplicación

**Operadores Lógicos**

& AND lógico booleano \* izquierda a derecha

| OR inclusivo lógico booleano \* izquierda a derecha

&& AND lógico izquierda a derecha

| | OR lógico izquierda a derechag

**Operadores relacionales**

== igual a izquierda a derecha

=! diferente a izquierda a derecha

> mayor que izquierda a derecha

< menor que izquierda a derecha

>= mayor o igual que izquierda a derecha

<= menor o igual que izquierda a derecha

() paréntesis izquierda a derecha

**Precedencia de operadores**

Mas alta

()

++ --

\* / %

> >= < <=

=

Mas baja

**Comentarios**

Comentario de línea

## código

Comentario sección de código

/#

Código

#/

**Bloque de Decisión**

If

evaluando (condición) {

Cuerpo;

}

If else

evaluando (condición) {

Cuerpo;

} mas {

Cuerpo;

}

If else if

evaluando (condición) {

Cuerpo;

} mas evaluando(condición){

Cuerpo;

}

**Bloque de Iteración**

sea (ent var, var < var2.length, var++){

cuerpo;

}

**Bloque Switch**

ent var = 0;

considerando ( var ) {

op 1: imprima(“ numero 1”);

op 2: imprima(“numero 2);

otro: imprima(“ hay : “ ´+ var + “ numeros”);

}

**Arreglo unidimensional**

Declaración de arreglo:

Tipo\_de\_dato UniArreglo nombre\_var;

Tipo\_de\_dato UniArreglo nombre\_var=nuevo Tipo[size];

Asignación de valor

Tipo\_de\_dato UniArreglo nombre\_var={elemento1, elemento2,… ,elementoN }

**Arreglo Bidimensional**

Declaración de arreglo:

Tipo\_de\_dato BiArreglo nombre\_var;

Tipo\_de\_dato BiArreglo nombre\_var=nuevo Tipo[size][size];

Asignación de valor

Tipo\_de\_dato BiArreglo nombre\_var={elemento1, elemento2,…},{elemento1,elemento2,…,elementoN}

**Main**

Fama Main(){

}

**Impresiones**

Fama.print(contenido);

Fama.println(contenido);

**Funciones**

fun nombre\_funcion(parametro1, parametro2, ….parameroN){

Cuerpo;

ret valor;

}

**Recursividad**

fun nombre\_función (parametros){

Evaluando(valor==0){

ret 1;

} mas {

ret valor operador\_aritmetico nombre\_función(valor);

}

}