UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE CIENCIAS

COMPILADORES

Alumno: Daniel Córdova

Alumno: Carlos Chauca

ÍNDICE

1. Nombre del compilador

El nombre del compilador es "CPY" porque éste compilador tomó como base los lenguajes C y Python, el objetivo de éste compilador es guiar a aquellas personas que recién se estan introduciendo en el mundo de la programación.

2. Estado del arte

3. Características del compilador

Lo primero que se planteó al crear éste compilador es como ayudar a las personas que quieren iniciarse en el mundo de la programación, tomando ésto como base decidimos empezar por lo más básico y elemental del del código, el pseudocódigo, a partir de ésto se creó un compilador que valide éste tipo de código, con el objetivo de que los programadores nuevos puedan validar sus ideas antes de empezar a programar en lengueges de más bajo nivel. Una de sus principales características es que tiene incluído la mayoría de los bucles conocidos y condicionales de la mayoría de lenguages para que pueda aceptar cualquier tipo de pseudocódigo, se definen tambien las operaciones básicas de la matemática y su correspondiente relación, validación de parentesis, llaves, etc; también se tiene que tener en cuenta que al final de cada asignación es "necesario" un punto y coma, ya que éste es uno de los mayores problemas con los que uno empieza al iniciar a programar, tenemos también implementada la creación de funciones, funciones recursivas y los bucles anidados, para que cualquier principiante entre en estos conceptos de la programación.

4. Reglas o patrones de la palabra reservada

4.1. Aritmética

ID = exp1 operador1 exp2;
ejemplo:
x = variable1 + 5 ;

4.2. Funciones

```
def nombreFuncion ( argumento )
{
    CuerpodelaFuncion ;
}
```

4.3. Condicional

```
SI ( argumento ) ;
{
    CuerpoCondicion ;
}
```

```
SI ( argumento ) ;
{
    CuerpoIF ;
}
SINO
{
    CuerpoELSE ;
}
```

4.4. Estructura While

```
WHILE ( argumento ) ;
{
    CuerpodelbucleWhile ;
}
```

4.5. Estructura For

```
FOR ( DESDE asignar ; HASTA argumento ; INCREMENTANDO asignar )
{
    CuerpodelbucleFor ;
}
```

4.6. Llamado a la Funcion

```
nombreFuncion ( elemento ) ;
```

5. Tokens usando bison

Se implementaron los siguientes tokens:

- ID: Hace referencia a un ID.
- NUM: Hace referencia a un número.
- ENTER: token usado para un salto de línea.
- COM: Hace referencia a una coma simple.
- IF: Token usado al momento de llamar una condicional "if".
- ELSE: Token usado al momento de llamar la condicional "else".
- OTHERCASE: Token usado al momento de llamar la condicional por defecto.
- IFELSE: Token usado al momento de llamar una condicional "ifelse".
- WHILE: Token usado al momento de llamar al bucle "while".
- FOR: Token usado al momento de llamar al bucle "for".
- DEF: Token usado al momento de llamar una función.
- SUM: Token para la suma.
- RES: Token para la resta.
- MUL: Token para la multiplicación.
- DIV: Token para la división.
- POW: Token para la potencia.
- AND: Token para el operador "and".
- OR: Token para el operador "or".
- PROG: Token inicial para el programa.
- VARS: Token en caso de asignar multiples variables.
- DESDE: Token para definir desde donde empieza a contar en el bucle "for".
- HASTA: Token para definir hasta donde empieza a contar en el bucle "for".
- INCREMENTANDO: Token para definir en cuanto aumenta el bucle "for".

6. Precedencia de la gramática

exp1 : exp1 operador1 exp2
exp2 : exp2 operador2 elemento
elemento: ID | NUM | '-' NUM ;

operador1: SUM | RES ;
operator2: MUL | DIV | POW ;

En las líneas superiores se observa la precedencia que existe entre los operadores matemáticos, se observa que la precedencia se aplica en el "operador2", el cual en caso de ser llamado, se ejecutaría antes que el "operador1".

7. Definir la sintaxis de las sentencias selectivas y repetitivas

7.1. Estructura IF:
IfEstructura: SI (condicion) { cuerpoIf }

7.2. Estructura IF and ELSE:

IfEstructura: SI (condicion) { cuerpoIf } SINO { cuerpoElse }

7.3. Estructura While:
WhileEstructura: WHILE (condicion) { cuerpoWhile }

7.4. Estructura For:
ForEstructura: FOR (DESDE asignar ; HASTA argumento ;
AUMENTANDO asignar) { cuerpoFor }

7.5. Estructura de una funcion:
FuncionEstructura: DEF nombreFuncion (argumentos) { cuerpoFuncion }

8. Método SHIFT - REDUCE