TP1 - LYM - RAUL RINCON - 202120414

miércoles, 7 de septiembre de 2022 06:46 p. m.

EBNF — "Mini JAVA"

1. La definición de las clases siempre esta definida con su super clase.
2. El constructor de la clase tiene un parámetro para cada uno de los atributos y siempre llaman el método `super()` como la primera instrucción. Solo los constructores deben usar asignaciones para definir los atributos de la clase.
3. El recibidor de la clase para el acceso a los atributos siempre lleva `this`.
4. Los métodos de la clase todos deben retornar una nueva instancia del objeto como su única instrucción.
5. las expresiones en el lenguaje corresponden a: la creación de un objeto (`new A()`), los llamados a métodos (`e.setfst(e2)`), el acceso a los atributos (`this.fst`), o el acceso a las variables (llamando su nombre `x`).

Ejemplos de los posibles programas que se pueden definir en el lenguaje se encuentran a continuación.

```
class A extends Object {
    A() { super(); }
}
```

Snippet 1: Definición de clase

```
class Pair extends Object {
  Object fst;
  Object snd;

  Pair(Object fst, Object snd) {
    super(); this.fst=fst; this.snd=snd;
  }
  Pair setfst(Object newfst) {
    return new Pair(newfst, this.snd);
  }
}
```

Snippet 2: Definición completa de clase

C := ("a-z" | "A-Z" | "0-9")

CID := ("A-Z" C*)

ID := ("a-z" C*)

OBJ := CID ID

A := OBJ ("," OBJ)*

ClassD := "class" CID "extends" CID Block

Block := "{" atributo* Statement* "}"

atributo = OBJ ";"

OBJN := CID "(" A? ")"

CONSB := "{" "super()" ";" (put)* "}"

Statement = OBJN consB | OBJ "(" A* ")" "{" "return" F ";" "}"

R ::= ID | CID

Call ::= R "(" (F ("," F)*)* ")"

Put ::= "this" "." ID "=" F ";"

F ::= "new" Call | (ID | "this") "." (Call | ID) | ID