Departamento de Tecnologías de la Información

Área de Ciencias de la Computación e Inteligencia Artificial

Procesadores de lenguajes

Examen de junio

EJERCICIO 1 (2 puntos)

Considere la gestión de la memoria en tiempo de ejecución.

(a) ¿En que consiste la memoria de pila?

(b) Describa la estructura del registro de activación de una función.

(c) Describa el proceso de llamada a una función.

(d) Describa el proceso de retorno de una función.

EJERCICIO 2 (2 puntos)

La siguiente figura muestra una expresión regular formada por los símbolos **a**, **b** y **o**.

(b (a b)*) (o b (a b)*)*

Obtenga el Autómata Finito Determinista asociado, indicando el conjunto de expresiones regulares punteadas que describen cada estado del autómata.

EJERCICIO 3 (2 puntos)

La siguiente gramática representa la sintaxis de las cláusulas import en Java:

Cláusula → **import** *CoP* **semicolon**

CoP → *Clase*

CoP → *Paquete*

Paquete → *Clase* **dot star**

Clase → **id**

Clase → *Clase* **dot id**

Construya el autómata reconocedor de prefijos viables y la tabla de análisis SLR de la gramática planteada. Utilice para ello la tabla incluida en la última página.

EJERCICIO 4 (2 puntos)

La siguiente gramática permite describir los pedidos de una cafetería.

Pedido → *Lista*

Lista → *Término SigueLista*

SigueLista → **plus** *Término SigueLista*

SigueLista → λ

Término → **producto**

Término → **num prod** *Término*

Término → **lpar** *Lista* **rpar**

Para representar estos pedidos se pretenden utilizar listas de términos. Cada uno de estos términos representa un producto del pedido y la información incluye el número de unidades de ese producto (por ejemplo, tres cafés) y se describe por medio de un objeto de la clase *Term* (que se describe en la página siguiente).

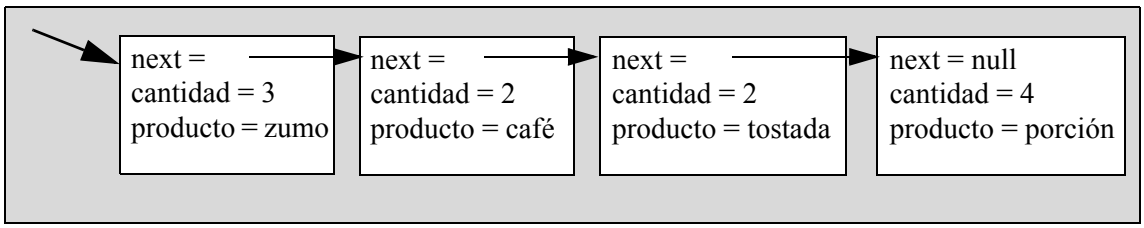
(a) Calcule los conjuntos Primeros, Siguientes y Predicción de la gramática.

(b) Construya la tabla de análisis sintáctico LL(1) para la gramática

(c) Modifique la gramática anterior para que el símbolo *Pedido* devuelva una lista de términos formada por objetos de la clase *Term*. Por ejemplo, para la entrada

3 * zumo + 2 * (café + tostada + 2 * porción)

la lista de términos a generar es la siguiente:



Estructura de datos a utilizar

```
public class Term {

   private double cantidad;
   private String producto;
   private Term next;

   public Term(double c, String p) {
     this.cantidad = c;
     this.producto = p;
     this.next = null;
   }

   public double getCantidad() { return this.cantidad; }
   public void setCantidad(double c) { this.cantidad = c; }

   public String getProducto() { return this.producto; }
   public void setProducto(String p) { this.producto = p; }

   public Term getNext() { return this.next; }
   public void setNext(Term t) { this.next = t; }

}
```

EJERCICIO 5 (2 puntos)

La siguiente figura muestra la descripción sintáctica de la instrucción WHILE en el formato de la herramienta JavaCC.

```
void InstWhile() :
{}
{
  <WHILE> <PARAB> Condicion() <PARCE> Instruccion()
}
```

Se pretende enriquecer la gramática anterior para construir el código intermedio de 3 direcciones correspondiente a la instrucción *while*. Para ello, la función *InstWhile()* deberá devolver un objeto *Inst* cuyo campo *code* contenga dicho código.

Considere que la función *Condicion()* devuelve un objeto de la clase *Condition* con los siguientes campos: *code*, código intermedio que describe la condición; *label_true*, etiqueta a la que salta el control en el caso de que la condición sea cierta; *label_false*, etiqueta a la que salta el control en el caso de que la condición sea falsa. Considere, asímismo, que la función *Instruccion()* devuelve un objeto de la clase *Inst* que contiene el campo *code* con el código asociado a la instrucción. Por último, se dispone del método *getNewLabel()* que devuelve una nueva etiqueta (es decir, no utilizada en ningún punto del código).

Estado	import	id	dot	star	semicolon	$	Cláusula	CoP	Paquete	Clase