Normas del examen PdL: PEI2

- Tiempo máximo: 90 minutos.
- Sólo se contestará en las caras de las hojas que estén habilitadas para ello con cabecera de firma. Cualquier texto/respuesta/apunte en otro lugar será ignorado.
- Encima de su mesa sólo pueden estar dos bolígrafos y su documento de identificación como alumno (carné de Universidad/DNI).
- **No se puede doblar, grapar, unir, rasgar las hojas del examen.**
- Debe entregar **TODAS** las hojas del examen.
- Recuerde **identificar todas las hojas** del examen con su DNI/NIF/Pasaporte en la esquina superior derecha de las mismas, acompañadas de su firma si así lo desea.
- No se puede abandonar el aula hasta que pase el tiempo de examen.

Pregunta 1: 1 puntos. (tiempo estimado: 10 minutos)

TEST. Rodee la letra de la respuesta válida (sólo una por pregunta)

Pregunta 2: 1,5 puntos. (tiempo estimado: 20 minutos)

Dado un fichero en formato XML estándar diseñe la tabla de símbolos asociada, mostrándola de manera gráfica y como definición de estructuras en java, asociando cada definición con el elemento gráfico correspondiente.

Ejemplo XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!<attachment type="gif" name="picture.gif" /></body>
</note>
```

Pregunta 3: 2 puntos (tiempo estimado: 30 minutos)

Para el código se muestra a continuación

```
01          a = 1
02          b = 0
03 L0:      a = a + 1
04          b = p + 1
05          if (a > b) goto L3
06 L1:      a = 3
07          if (b > a) goto L2
08          b = b + 1
09          goto L1
10 L2:      a = b
11          b = p + q
12          if (a > b) goto L0
13 L3:      t1 = p * q
14          t2 = t1 + b
15          return t2
```

Determine los siguientes puntos

a) Los bloques básicos de las instrucciones y el gráfico de flujo de control
b) Identificar los bucles y determinar si están o no anidados (explicar por qué o por qué no).
c) Las variables vivas al final de cada bloque básico. Una variable se dice que está viva al final de un bloque básico dado o instrucción si se utiliza su valor más allá de ese bloque o instrucciones. Suponga que las variables se declaran en el primer uso. No es necesario determinar las variables vivas antes y después de cada bloque y justificar su respuesta para el valor presentado para el que contiene la instrucción bloque básico en la línea 11.
d) Escriba un código de función java equivalente al pseudocódigo mostrado (preste atención al ámbito de vida de las variables).
e) Transforme el código en tripletas.
f) ¿Qué podría hacer para optimizar este código?

Pregunta 4: 2 puntos (tiempo estimado: 20 minutos)

Dada la gramática:

Expr → Expr Op Expr | n | p

Op → suma | resta

Donde n es un número entero positivo y p una cadena de texto

a) Determine la gramática de atributos correspondiente para poder realizar operaciones matemáticas y concatenación de cadenas sin errores (no existe la concatenación de cadenas y números). Añada las reglas semánticas correspondientes.
b) Responda a la siguiente cuestión: Es una gramática S-Atribuida o L-Atribuida
c) Creen una gramática equivalente que permita en formato alternativo (Si era S-Atribuida, cree una equivalente L-Atribuida, o viceversa)

Pregunta 5: 3 puntos (tiempo estimado: 40 minutos)

Dado el siguiente código, genere el código resultante según el juego de instrucciones adjunto, teniendo en cuenta que debe realizarse un sistema metódico usable en cualquier situación como compilador.

```
function f1(numero param1, numero param2): numero
{
	numero temporal:=-1;
	if (param1 > param2){
		return temporal;
	}
	return param1+param2;
}

function f2(numero p1): numero
{
	return p1>0;
}

function main(): void
{
	numero b:=0;
	for (numero a:=0; a<=b; a++)
	{
		if (a>b) {
			a:=f1(a,b);
		} else {
			b:=f1(b,a);
		}
	}
	f2(12);
}
```

Juego de instrucciones (válido para el examen)

Nº	Nombre	Parámetros	Descripción	Coste
01	PUSH	Tipo,Valor	Establece una variable en la pila de tipo Tipo con Valor V.	10
02	LOAD	Registro,CPi	Carga en el Registro el valor almacenado en la pila apuntada por el puntero CPi	5
03	ADD	RegistroA,RegistroB	Suma los valores del RegistroA y el ResgistroB y deja el resultado en el RegistroR.	2
04	STO	CPi,Registro	Almacena en la posición de la pila indicada por CPi el valor del registro indicado.	5
05	INC	Registro	Suma 1 al registro indicado	1
06	CMP	RegistroA,RegistroB	Compara dos registros. Si son iguales: valor 0, si el registroA es mayor que el registroB: valor 1, si viceversa: valor -1.	1
07	PUT	Registro,Valor	Establece el registro indicado al valor Valor.	2
08	POP	Registro	Extrae la última variable de la pila y la sitúa en registro indicado.	7
09	GOTO	Dirección	Mueve el puntero del programa a la dirección indicada	2
10	JMPNE	Dirección	Mueve el puntero del programa a la dirección indicada si el registro RCMP es distinto de 0	3
11	JMPE	Dirección	Mueve el puntero del programa a la dirección indicada si el registro RCMP es igual a 0	3
12	JMPGT	Dirección	Mueve el puntero del programa a la dirección indicada si el registro RCMP es mayor que 0	3
13	JMPLT	Dirección	Mueve el puntero del programa a la dirección indicada si el registro RCMP es menor que 0	3
14	INIT	Dirección	Establece en la pila una nueva entrada que marca el inicio de una función, almacenando el puntero de pila de la función actual y el puntero de programa de la instrucción siguiente a esta, para saltar a la Dirección indicada por parámetro.	10
15	REST		Si la pila contiene en su última posición una entrada INIT, elimina la entrada restaurando el puntero de pila almacenado y saltando a la dirección almacenada.	10

Registros disponibles:
RA, RB, RC, RD, RE, RF, RR, RCMP

Direccionamiento de memoria:
Pila vía puntero CPi, que permite direccionamiento dinámico: CPi + 1, CPi – 3. CPi siempre apunta al inicio de la pila para el ámbito de la función actual.

Direccionamiento de código:
Cada tripleta tiene un número de instrucción que la identifica.