**CANTIDAD DE HOJAS:**

|  | | | **UNIVERSIDAD DE BELGRANO** | | |
| --- | --- | --- | --- | --- | --- |
| **Facultad de Ingeniería y**  **Tecnología Informática**  LENGUAJES DE PROGRAMACIÓN (0050100023LENGP)  Profesor: Mag. Ing. Pablo Pandolfo | | |
| Examen Parcial octubre 2020  ALUMNO: LU: FECHA:  CARRERA: | | | | | |
| **NOTA: EL EXAMEN ESCRITO ES UN DOCUMENTO DE GRAN IMPORTANCIA PARA LA EVALUACIÓN DE LOS CONOCIMIENTOS ADQUIRIDOS, POR LO TANTO, SE SOLICITA LEER ATENTAMENTE LO SIGUIENTE:**   * Responda claramente cada punto, detallando con la mayor precisión posible lo solicitado. * Sea prolijo y ordenado en el desarrollo de los temas. * Sea cuidadoso con las faltas de ortografía y sus oraciones. * No desarrollar el examen en lápiz. * Aprobación del examen: Con nota mayor o igual a 4 (cuatro) * Condiciones de aprobación: 60% correcto. * Duración de examen: 3 horas. | | | | |  |
|  |  | Ejercicio 1 [2 puntos]: Resuélvase mediante programación lógica el siguiente problema: Se consulta por ejemplo: parse([comienzo, norte, norte, este, fin]) y tiene que devolver true si es aceptado por el AF que reconoce las palabras del lenguaje representado por la ER: comienzo (norte | sur | este | oeste)\* fin  **delta(0, comienzo, 1).**  **delta(1, norte, 1).**  **delta(1, sur, 1).**  **delta(1, este, 1).**  **delta(1, oeste, 1).**  **delta(1, fin, 2).**  **inicio(0).**  **final(2).**  **transición(X, []) :- final(X).**  **transición(X, [A|B]) :- delta(X, A, Y), transición(Y, B).**  **parse(L) :- inicio(S), transición(S, L).**  Ejercicio 2 [2 puntos]: Diséñese en BNF la gramática que genera las palabras del siguiente lenguaje:   * Una **expresión** puede ser un átomo o una lista. * Un **átomo** puede ser un número o un símbolo. * Un **número** es una secuencia continua de uno o más dígitos decimales, precedido opcionalmente por un signo (+) o un signo (-). * Un **símbolo** es una letra seguida de cero o más caracteres (dígitos o letras) * Una **lista** es un par de paréntesis que abren y cierran. (con cero o más expresiones en medio separadas por punto y coma).   **<expresión> ::= <atomo> | <lista>**  **<atomo> ::= + <numero> | - <numero> | <numero> | <símbolo>**  **<numero> ::= 0 <numero> | … | 9 <numero> | 0 | … | 9**  **<símbolo> ::= <letra> | <letra> <letras o dígitos>**  **<letras o dígitos> ::= <letra> <letras o dígitos> | <numero> <letras o dígitos> | <letra> | <numero>**  **<letra> ::= A | … | Z | a | … | z**  **<lista> ::= () | (<expresiones>)**  **<expresiones> ::= <expresión> | <expresión> ; <expresiones>**  Ejercicio 3 [2 puntos]: Escríbase la gramática BNF para el siguiente Diagrama de Sintaxis, el símbolo de inicio es <statement>. Luego, derívese verticalmente, mediante el árbol de análisis sintáctico una palabra del lenguaje.  Diagrama  Descripción generada automáticamente  **<statement> ::= new <identifier> : <type>**  **<type> ::= init <expression> <type> | alias <expression> <type> | in <statement> ni**  Ejercicio 4 [1 punto]: Obténgase el lambda calculus de la expresión **suc 2** sabiendo que suc (máquina sucesora o siguiente) es λa. λb. λc. b(abc) y 2 es λf. λx. f(fx)  **λb. λc. b(λf. λx. f(fx) bc)**  **λf. λx. f(f(fx))**  Ejercicio 5 [1 punto]: Calcúlese {P} y {Q} y describa computabilidad de C de la siguiente función:  BEGIN  z:=0; u:=x; REPEAT z:=z+y; u:= u-1 UNTIL (u=0);  END  **P = {x>0}**  **Q = {z=x\*y Λ u=0}**  **C: Calcula el producto de dos números enteros.**  Ejercicio 6 [1 punto]: Resuélvase operacional y declarativamente el siguiente problema: “obtener los números impares de una lista de números”. ¿En cuál solución tengo mayor control sobre el algoritmo? ¿En cuál solución tengo mayor foco en lo que tengo que resolver? Justifique.  **type LISTA = array [1..longitudMaxima] of integer;**  **procedure impares (listaNumeros: LISTA, var listaImpares: LISTA)**  **begin**  **j <- 0;**  **recorrer i de 1 a longitudMaxima**  **begin**  **if (listaNumeros[i] es impar)**  **begin**  **listaImpares[j] = listaNumeros[i];**  **j <- j + 1;**  **end**  **end**  **end.**  **impares A = [ x | x <- A , odd x ]**  Ejercicio 7 [1 punto]: Complétese el programa y la tabla (máxima heterogeneidad):     | 01 | public class A { | | --- | --- | | 02 | public int a; | | 03 | private boolean b = true; | | 04 | **public void m() { int c = 4; }** | | 05 |  |      | **identificador** | **l-valor** | **r-valor** | **Alcance** | | --- | --- | --- | --- | | a (linea 2) | automática | basura | global | | **b (linea 3)** | **automática** | **true** | **dentro de la clase** | | **c (linea 4)** | **automática** | **4** | **local al método m** | | |  |