**CANTIDAD DE HOJAS:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | **UNIVERSIDAD DE BELGRANO** | | |
| **Facultad de Ingeniería y**  **Tecnología Informática**  LENGUAJES DE PROGRAMACIÓN (0050100023LENGP)  Profesor: Mag. Ing. Pablo Pandolfo | | |
| Examen Recuperatorio noviembre 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 [4 puntos]: Resuélvase mediante programación lógica el siguiente problema que ayude a una agencia matrimonial, respondiendo consultas sobre qué parejas son compatibles. Definimos a una pareja como un par (mujer, varón), no al revés. Incluir en el programa la siguiente información: Las mujeres melancólicas son compatibles con los varones serenos. Las mujeres decididas son compatibles con los varones reflexivos. Las mujeres soñadoras son compatibles con los varones Decididos. Juan es sereno y decidido. María es melancólica. Úrsula es decidida. Juana es soñadora. Pedro es reflexivo. José es melancólico. Según la información descripta, la pareja (María, Juan) es compatible, mientras que la pareja (Úrsula, Juan) no lo es. Agregar al programa la siguiente información: Cualquier pareja formada por un decidido y un melancólico es compatible. Según esta nueva información, la pareja (Úrsula, José) es compatible, mientras que si nos remitimos al punto a. no lo es. Agregar al programa la posibilidad de responder a consultas sobre si una persona es deseable. Decimos que una persona (varón o mujer) es deseable si es compatible con, por lo menos, dos personas distintas. Según la información descripta, Juan es deseable mientras que Juana no lo es  **mujer(maria).**  **mujer(ursula).**  **mujer(juana).**  **hombre(juan).**  **hombre(pedro).**  **hombre(jose).**  **sereno(juan).**  **decidido(juan).**  **decidido(ursula).**  **melancolico(maria).**  **melancolico(jose).**  **sonadora(juana).**  **reflexivo(pedro).**  **distintoSexo(M,H):-mujer(M),hombre(H).**  **compatible(M,H):-distintoSexo(M,H),melancolico(M),sereno(H).**  **compatible(M,H):-distintoSexo(M,H),decidido(M),reflexivo(H).**  **compatible(M,H):-distintoSexo(M,H),sonadora(M),decidido(H).**  **compatible(M,H):-distintoSexo(M,H),melancolico(M),decidido(H).**  **compatible(M,H):-distintoSexo(M,H),melancolico(H),decidido(M).**  **esDeseable(Persona):-compatible(Persona,M), compatible(Persona,H), M = H.**  Ejercicio 2 [3 puntos]: Diséñese una GIC en notación BNF para describir la sintaxis de:  En el lenguaje C la estructura del prototipo de una función es la siguiente:  tipo nombre de la función (lista de parámetros);  Donde:  tipo: es un tipo válido de datos: char, int, float, void.  nombre de la función: es un identificador, que comienza con una letra y puede seguir con letras y números.  lista de parámetros: es una lista de parámetros separados por comas, de alguna de las siguientes formas (puede estar vacía): Tipo nombre o Tipo \*nombre o Tipo nombre []  La lista de parámetros puede estar vacía y el nombre del parámetro es opcional (puede o no estar en el parámetro).  Son símbolos terminales: int, char, float, void, paréntesis, corchetes, coma, punto y coma.  Ejemplo de cadenas:  int fcalculo (char y, char \*vy, float z, float vz []);  int fcalculo (int x, int \*vx , char, char \*, float, float []);  void mostrar ();  **<prototipo> 🡪 <tipo><nombre>(<listaparam>); | <tipo><nombre>();**  **<listaparam> 🡪 <param> | <listaparam>, <param>**  **<param> 🡪 <tipo><nombre> | <tipo>\*<nombre> | <tipo><nombre>[] | <tipo>| <tipo> \* | <tipo>[]**  **<tipo>🡪 float | int | char | void**  **<nombre> 🡪 <letra> | <nombre><letra> | <nombre><dígitos>**  **<dígito> 🡪 0 | 1 | 2 | … | 9**  **<letra> 🡪 a | b | c | … | z**  Ejercicio 3 [2 puntos]: Diséñese el Diagrama de Sintaxis para la siguiente producción:  <decl\_proc> ::= **procedure** <identificador> {<lista\_param\_form>}\* **;** <bloque>  | **proceduce** <identificador> {<lista\_param\_form>}\* ; **forward**  Diagrama  Descripción generada automáticamente  Ejercicio 4 [1 puntos]: Grafíquese el Árbol de Análisis Sintáctico para la palabra:  C := D – E \* F  GIC:  <sent\_asig> ::= <var> := <expresion>  <expresion> ::= <expresion> + <termino> | <expresion> - <termino> | <termino>  <termino> ::= <termino> \* <factor> | <termino> / <factor> | <factor>  <factor> ::= ( <expresion> ) | <var> | <num>  <var> ::= A | B | C | D | ... | Z  <num> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9  Diagrama  Descripción generada automáticamente | |  |