Lista de chequeo - Plantilla

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Respuesta** | **Check** | **FECHA** | **Porcentaje** |
| Lista de participantes  (Restricciones: debe haber a lo sumo un estudiante MACC 1.0) | Miguel Valencia y Nicolás Rojas | OK | 17 de agosto  (1%) | 1 |
| Repositorio github | https://github.com/miguelvalzul/Logica-para-CC | OK | 24 de agosto  (1%) | 1 |
| Situación a representar  (Resumen) | Hallar un diploma detrás de una de cuatro puertas, sabiendo que tres de ocho afirmaciones sobre la ubicación del mismo son correctas. | OK | 31 de agosto  (2%) | 2 |
| Situación a representar  (archivo en repositorio) | Problema\_puertas.pdf | OK |
| Representación  (Resumen) | 4 átomos (V: el deploma está detrás de la puerta; F: el diploma no está detrás de la puerta).  El diploma solo puede estar detrás de una puerta => Fórmulas “Si…, entonces…” con antecedente puerta donde está el diploma y consecuente puertas donde no puede estar.  Exactamente 3 afirmaciones son verdaderas => cláusulas con 4 literales y fórmulas con conectivos binarios obtenidas de esos 4 literales. | OK | 14 de septiembre  (1%) | 1 |
| Representación  (archivo en repositorio) | Claves\_representacion.pdf | OK |
| Implementación gráfica de la representación  (uso en línea de comando) | $ python visualizacion.py puertas.csv  “puertas.csv” es un archivo donde cada línea, en caso de que haya varias, contiene la representación de cuatro puertas que pueden o no tener asociado un diploma. Cada puerta está representada por un literal. | OK | (1%) | 1 |
| Implementación gráfica de la representación  (archivo en repositorio) | visualizacion.py  Visualización de cuatro puertas de madera a partir de  una lista de literales. Cada literal representa una sola puerta. El literal es positivo si hay un diploma oculto detrás de la puerta en cuestión. Si es negativo, hay un monstruo.  Formato de la entrada: - las letras proposionales serán: 1, 2, 3, 4;  - solo se aceptan literales (ej. 1, -2, 3, -4, etc.)  Salida: archivo puertas\_%i.png, donde %i es un número natural | OK |
| Aplicación mediante tableaux  (uso en línea de comando) | $ python resolver\_tableau.py | OK | (3%) | 3 |
| Aplicación mediante tableaux  (archivo en repositorio) | resolver\_tableau.py | OK |
| Aplicación algoritmo DLL  (uso en línea de comando) | $ python resolver\_DPLL.py | OK | 9 de noviembre  (3%) | 3 |
| Aplicación algoritmo DLL  (archivo en repositorio) | resolver\_DPLL.py | OK |
| Sustentación |  |  | 21 de noviembre | 3 |