PRACTICA 1 – INTELIGENCIA ARTIFICIAL

MEMORIA

Búsqueda en amplitud simple por coste uniforme

Las estructuras de datos implementadas para hacer este tipo de búsqueda han sido:

1. **std::set<int> visitado**
   He usado un set para meter los nodos visitados ya que es una estructura bastante fácil de usar y de determinar si un elemento está dentro de esta.

2. **std::vector<int> padre**
   Con este vector se puede determinar el nodo padre a la hora de buscar el camino desde v2 hasta v1. En principio se inicia todo a -1.

3. **std::queue<int> cola**
   La cola es necesaria para tomar un registro en el orden correspondiente de los nodos generados y que posteriormente habrá que observar.

4. **std::vector<std::vector<int> mat_ad**
   Esta es la matriz de adyacencia obtenida del fichero de entrada.

5. **std::stack<int> camino**
   Al final del algoritmo se recorrerá el vector de padre empezando en v2 hasta encontrar todos los nodos que conecten v2 y v1 y mientras tanto esto se pusheará en la pila para más tarde darle la vuelta e imprimir el camino desde v1 a v2.

TABLA DEL COMPORTAMIENTO DEL PROGRAMA:

	Nodo_Origen	Nodo_Destino	Camino	Coste	Nodos Visitados	Nodos Generados	C7
1	1	10	1 -> 4 -> 5 -> 7 -> 8 -> 9 -> 10	695	13	13	ID1
2	2	7	2 -> 3 -> 4 -> 5 -> 7	448	10	14	ID1
3	1	8	1 -> 4 -> 5 -> 7 -> 8	504	9	10	ID2
4	3	9	3 -> 4 -> 5 -> 7 -> 8 -> 9	616	10	11	ID2
5	1	6	1 -> 2 -> 5 -> 6	299	10	12	ID3
6	3	7	3 -> 4 -> 8 -> 9 -> 10 -> 7	568	19	25	ID3
7	7	10	7 -> 10	120	3	3	ID4
8	1	8	1 -> 8	140	4	5	ID4