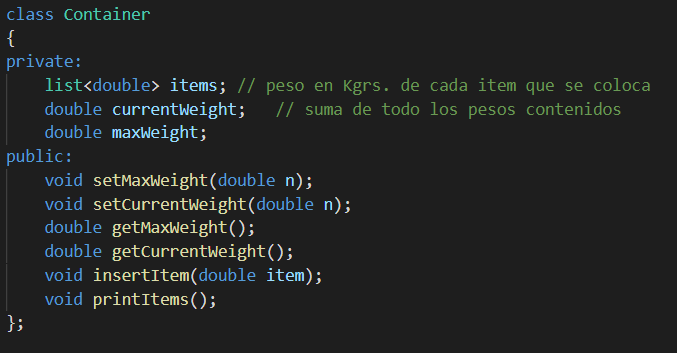
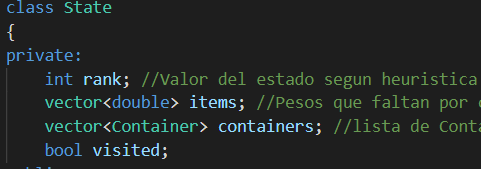
Modelamiento del problema.

Para modelar el problema se crean la clase Container y State.



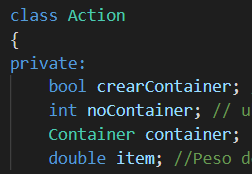


Dónde State posee un vector de ítems, que representan los valores de cada objeto que se desea almacenar en la maleta. Además cada estado presenta un vector de Container, el cual representa todas las maletas que se utilizan en ese estado para guardar los objetos.

Por parte de la clase Container, esta cuenta con un vector de items de tipo double. El cual representa todos los objetos almacenados en él. Finalmente posee variables de tipo double que representan el peso máximo que puede soportar y el peso actual que posee.

Creación de grafo.

Para lograr resolver este problema mediante el uso de grafos, se crea la clase action. La cual posee como parámetro todos las variables necesarias para crear un nuevo estado.



Mediante la función get\_actions(state&) se obtiene una lista que representa todas las posibles acciones que se pueden realizar en el estado que reciba la función. Y transition(state&,&action), es la función encargada de crear estos nuevos estados adyacentes.

Solución propuesta.

Para solucionar el problema, se toma cada objeto y se trata de colocar en cada uno de los containers existentes. Si no existe un container que pueda guardar el objeto, se creará uno nuevo y se colocará en él. Se repite este procedimiento hasta que cada ítem se encuentre guardado en un container.

Utilizando dos medios de búsqueda, búsqueda por profundidad y Best first, se busca encontrar los estados solución al problema. En la búsqueda por profundidad (DFD) se crean la totalidad de los estados y al final se compara cuál de todos ellos utiliza menos containers para guardar los objetos.

En Best first, se utiliza una heurística que rankea cada estado, es decir, le asigna un valor que representa que tan bueno es. Teniendo así que el estado con el numero ranking menor será el mejor estado solución. El ranking es creado mediante la suma de containers utilizados y la cantidad de ítems que faltan por colocar. Gracias a esto el primer estado final encontrado representará el mejor estado solución.

Conclusión

Al final de realizar diferentes pruebas, pudimos comprobar que utilizando Best first se llega a una solución de manera más rápida y certera.