**Container ship – Design**

Data structures:

1. Ship
   1. Container Plan [][] # 2D matric of stacks of containers
   2. Port Route [] # the path of the ship to the ports
   3. Int weight # starts with 0
2. Container
   1. Int weight
   2. Port destPort
   3. String uniqueId
3. Port
   1. String port (5 English letter code)
4. Container Map (dictionary)
5. Stack for unloaded containers to be loaded back
6. Array of linked lists of stacks for loading to the lowest floor

Functions:

1. Main()
2. Init\_ship(Port route[])
3. getShipPlan()
4. getInstructionsForCargo(container Instructions[])
5. Stowage(ship, port)
6. Optimize()
7. Load(ship, container)
8. Unload(ship, container)
9. Weight\_balance(ship)
10. Simulation ?

Flow:

1. Main function calls:
   1. init\_ship to initialize the ship with weight = 0 and the given route
   2. for ( each port )
      1. stowage(ship, current\_port)
2. stowage(ship, current\_port) calls optimize() to find the number of loads/unloads, optimize calls weight\_balance(ship) to make sure the optimal solution gives a balanced ship, then loads/unloads according to the output of the optimize function

Variables can be passed as parameters or be a global variable which can be shared with all the functions