Design problems

Describe how you could solve each of the following problems using data structures we’ve learned so far. Specify the ADTs (Bag, Set, Stack, Queue, Deque, Priority Queue) you would use, how they would be used to support the required operations, and how they should be implemented to optimize efficiency.

1. You are the system manager for a large company, which has one high-speed printer. Two groups of people use this printer: engineers and managers. You’ve been asked to implement a printer queue for the printer, which has the following properties: it should print jobs in arrival order, but manager jobs must come first. In other words, the only time an engineer’s job should be printed is when there are no manager jobs waiting, but other than that, the jobs are printed in the order they arrived at the printer. How would you implement such a printer queue? Specify any ADTs you would use, their implementations, and how you would use them.

2. You are hired to help design software to help with a key airline operation: processing drink orders on a flight. Once the pilot gives the ok, airline staff walk from the back of the plane to the front taking everyone’s order. When they reach the front, they begin preparing those orders. Drinks are prepared in groups of 10, that being the number of cups that can fit onto a carrying tray. The first group whose drinks are prepared are those in the front of the plane (first class!). The second group is the next 10 people further back in the plane and so on. You are in charge of developing a new application that will let airline staff take orders and then display those orders 10 at a time in the appropriate order. How would you efficiently support these operations? Specify any ADTs you would use, their implementations, and how you would use them.

3. When designing your awesome animated role-playing game, you notice a problem with how moves are processed. Since a move (go to location X) can take some time to perform, the user will sometimes enter another move before the first one is complete. To avoid the confusion of multiple moves trying to execute at the same time, you want a data structure to store moves so they can be executed sequentially in the order they were originally entered. How would you efficiently support this? Specify any ADTs you would use, their implementations, and how you would use them.

4. The Justice League has had another run in with the Legion of Doom, led this time by Vandal Savage. The Legion’s newest scheme involves pelting the earth with a series of giant meteors. The plan is to cause widespread earthquakes and therefore destruction, ushering in a new age of civilization, with Vandal Savage in charge. Luckily, Batman has a contingency plan for every situation. He is going to have Superman fly up into the atmosphere and use his heat vision to destroy the meteors (or at least blow them up into smaller pieces) in the order that they are going to reach the ground. Batman needs a program into which he can enter meteors as they are discovered and which will output the meteors in the correct order so that he can tell Superman which one to destroy next. How would you efficiently support these operations? Specify any ADTs you would use, their implementations, and how you would use them.

5. You are working for the Guinness Book of World Records on the record for most books published in a single year. You are tasked with entering the titles of books written by an author during some year, and your boss will ask for the size of the list. Because records must be double and triple checked for accuracy, others will be entering titles for that author and year as well. Before the titles are counted, duplicate titles must be removed. The size will be checked against the current record holder (initially Barbara Cartland with 23 novels published in 1976), so neither the author nor year needs to be kept since the list will be scrapped if its length does not exceed the current record. How would you efficiently support this process? Specify any ADTs you would use, their implementations, and how you would use them.