
2. Exercise 9.3-7 in the book.

3. Consider the following data structures:

• stack
• queue
• linked list
• hash table
• binary search tree
• heap

For each of the following situations, select one of the above data structures that would be most appropriate for storing the relevant data. Justify your choice by explaining how the strengths of the data structure matches the needs of the application.

(a) Keeping track of changes made to a document in order to implement an undo operation.

(b) Allowing a router to deal with bursts of traffic without dropping packets.

(c) Providing an interface to a collection of large records that allows users to maintain their own smaller collections. User collections need to allow fast adding and removing of records. For example, personal libraries in an online streaming service.

(d) Storing the actual record data in the previous situation.

(e) Implementing a garbage collector for a programming language. The garbage collector keeps track of allocated chunks of memory and frees a chunk when there are no more existing references to it.

4. A previous student (Horwitz) took this class and proposed the following sort:

```
function HSORT(LIST)
    if LIST.length = 0,1 then
        return LIST
    end if
    if LIST.length = 2 then
        Compare the two items, swap them if necessary
        return list
    end if
    HSORT(leftmost two-thirds of LIST)
    HSORT(rightmost two-thirds of LIST)
    return LIST
end function
```

(a) Argue convincingly either that HSORT does or that it does not work.

(b) In any case, determine the time complexity of HSORT.