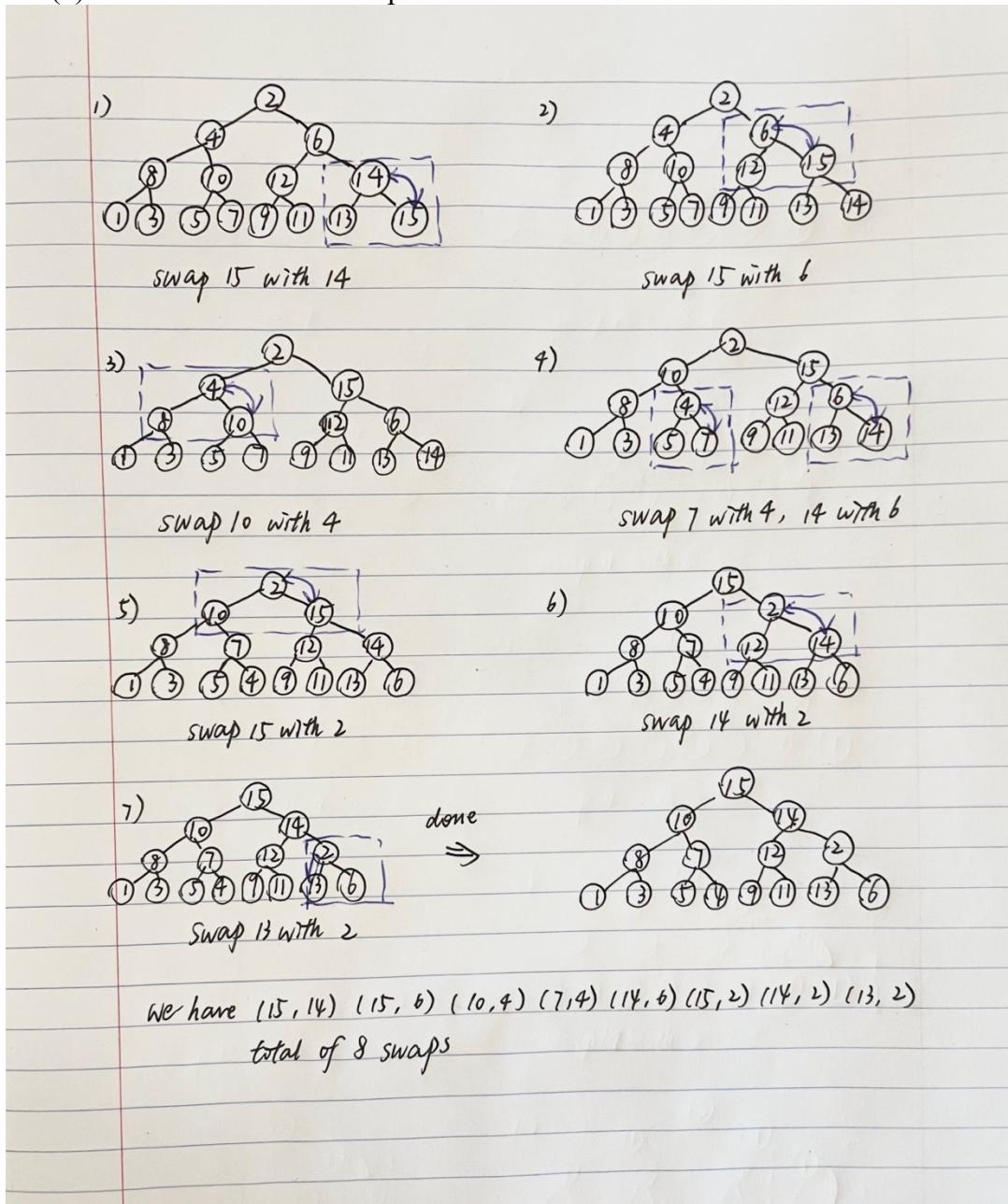


Problem 1

- (1) D
- (2) iii
- (3) $k = 5$
- (4) $512 + 1024 = 1536$
- (5) The total number of swaps is 8



$$(6) B + C + D + 0.2E = 15 + 24 + 30 + 7 = 76$$

Problem 2

(a)

Guess my word

normal



I'm thinking of an English word. Make guesses below and I'll tell you if my word is alphabetically before or after your guess.

hi
panda
school
sunny
system
tab

my word
is after: table

You got it!

(11 guesses in 1m 49s)

tackle

enter your name for the completion board
 submit

allow my guesses to be public

Come back tomorrow for a new word or [try a hard word?](#)

my word
is before: taco
tiger
zebra

(b)

We find the mid word and compare it with the search word to check if this word is before or after the mid word. We perform the task repeatedly until we find the exact word.

And this will be a good application for binary search problem, since $\log(2^k - 1) = k$ guesses.

```
1  function identifySecretWords (wordsList, k):  
2      l = 0, r = 2^(k) + 1  
3      mid = (r - l) / 2  
4      midWord = wordsList[mid]  
5      while (l < r) :  
6          if (searchWord < midWord):  
7              r = mid  
8          if (searchWord == midWord):  
9              return searchWord  
10         if (searchWord > midWord):  
11             l = mid + 1
```

(c)

Since $T(n)$ be the maximum number of guesses, by applying binary search algorithm, we get
 $T(n) = T(n/2) + O(1)$

By using Master theorem, we have $a = 1$, $b = 2$, $n^{\log_b a} = n^0 = 1$,
Therefore, we can get that $T(n) = \Theta(\log n)$.

(d)

The average guessing times = $\log_2(267,751) = 18.030532438$
Since $18.03 * 1 > 15$, I expect to lose money.

Problem3

(a)

Insertion sort will be faster since it's $O(n)$ if all elements are equal.

(b)

Quicksort is faster, since it doesn't do unnecessary element swaps. For Heapsort, even if all data is already ordered, you are going to swap 100% of elements to order the array.

(c)

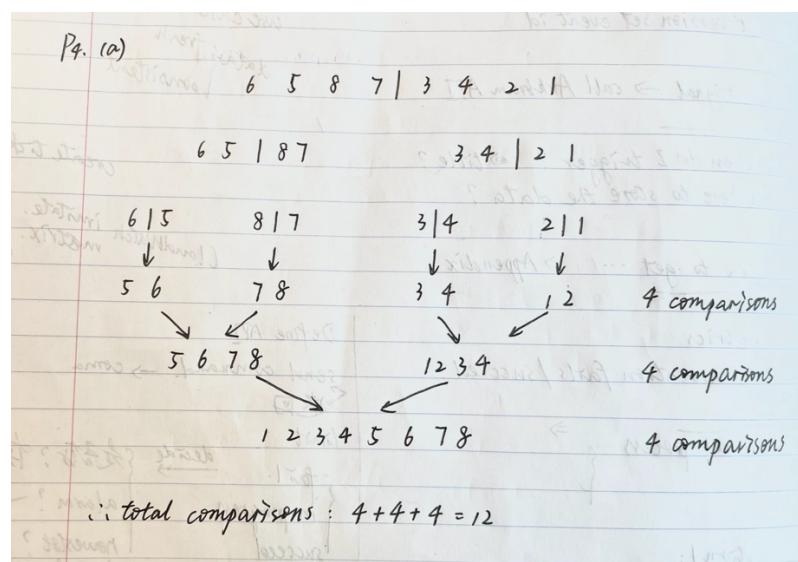
Bucket sort is faster, because it works best when the data are more or less uniformly distributed. The average time complexity for Bucket Sort is $O(n + k)$.

(d)

Counting sort is faster. Since the input range is fixed, time complexity is $O(9 + n)$

Problem4

(a)



(c)

P4. (c) Proof by induction.

Assume $M(n) = \frac{n \log n}{2}$ holds for $n=k$, we have $M(k) = \frac{k \log k}{2}$

Let $n=2k$,

$$\begin{aligned} M(2k) &= 2 \cdot M(k) + k \\ &= 2 \cdot \frac{k \log k}{2} + k \\ &= k \cdot \log k + k \\ &= 2k \cdot \left(\frac{\log k + 1}{2}\right) \\ &= 2k \cdot \frac{(\log k + \log_2 2)}{2} \\ &= 2k \cdot \frac{\log_2 2k}{2} \\ &= (2k) \cdot \frac{\log(2k)}{2} \end{aligned}$$

We found that $M(n) = \frac{n \log n}{2}$ holds for $n=2k$.

Therefore, we conclude that $M(n) = \frac{n \log n}{2}$ is valid.

Q.E.D.

(d)

Total number of inputs = 8,

P (exactly 12 comparisons) = 1,

Probability of getting single comparisons = $\frac{1}{2}$ (since divide the array into two halves)

By using binomial distribution:

$$\begin{aligned} P(X=x) &= \binom{n}{k} p^k q^{n-k} \\ &= \binom{12}{8} 0.5^4 0.5^8 \\ &= 495 * (1/4096) \\ &= 0.1208 \end{aligned}$$

Therefore, the probability pf getting exactly 12 comparisons in merge sort will be 0.1208

Problem 5

(b)

Greedy algorithm:

1. Color a vertex with color 1
2. Pick an uncolored vertex v . Color it with the lowest-numbered color that has not been used on any previously colored vertices adjacent to v . (If all previously used colors appear on vertices adjacent to v , this means that we must introduce a new color and number it.)
3. Repeat the previous step until all vertices are colored

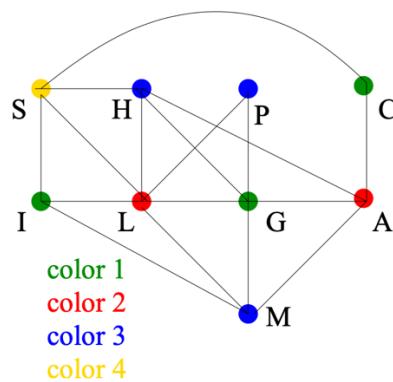
By applying this algorithm, we can get that $\chi(G) = 4$

(c)

No. The provided algorithm in (a) will not always use exactly $\chi(G)$ colors.

Examples:

For the following graph, if decide to color the vertices in order G, L, H, P, M, A, I, S, C. Then we can mark the graph with 4 colors.



However, if decide to color the vertices in order A, I, P, M, S, C, H, L, G. We have to mark the graph with 5 colors.

