Name: Ayser Jamshidi

Here are some basic rules for calculating the Big O for some $T(n)$ or an algorithm.

1. Only the highest degree of n matters. For example

$$T(n) = n^3 + 5n^2 + 10^7 \rightarrow O(n^3)$$

since once n becomes super-massively huge, the other terms just stop mattering.

2. Constant factors don't matter. $T(n) = 500n$ and $T(n) = 0.005n$ both $O(n)$. Again, as n becomes bigger, these constants stop mattering; what matters is the rate of growth. Example:

$$T(n) = 50n^3 - 2n^2 + 400 \rightarrow O(n^3)$$

3. Counting the number of nested loops usually works.

You can turn in this assignment physically to a TA or me. You can also scan and upload your answers.

_____ 0 points

For each of the following $T(n)$, write the corresponding Big O time complexity. Some series may require research.

1. (2 points) $T(n) = n^2 + 3n + 2$

1. O(n^2)

2. (2 points) $T(n) = (n^2 + n)(n^2 + \frac{\pi}{2})$

2. O(n^4)

3. (2 points) $T(n) = 1 + 2 + 3 + \ldots + n - 1 + n$

3. O(n)

4. (2 points) $T(n) = 1^2 + 2^2 + 3^2 + \ldots + (n-1)^2 + n^2$

4. O(n^2)

5. (2 points) $T(n) = 10$

5. O(1)

6. (2 points) $T(n) = 10^{100}$

6. O(1)

7. (2 points) $T(n) = n + \log n$

7. O(n)

8. (2 points) $T(n) = 12\log(n) + \frac{n}{2} - 400$

8. O(n)

9. (2 points) $T(n) = (n+1) \cdot \log(n) - n$

9. O(n log(n))

10. (2 points) $T(n) = \frac{n^4+3n^2+2n}{n}$

10. O(n^3)

_____ 20 points

11. (4 points) What is the time complexity to get an item from a specific index in an ArrayList?

O(1)

12. (3 points) What is the time complexity remove an item in the middle of an ArrayList?

O(n)

13. (3 points) Why?

Since we don't know if we're removing the item at the very end, which would be constant time, or anywhere else, we assume the worst case scenario: O(n) as this would be true if we removed element 1.

14. (3 points) What is the **average** time complexity to add an item to the end of an ArrayList?

O(1)

15. (3 points) What is the **worst case** time complexity to add an item to the end of an ArrayList? What if you have to or don't have to reallocate?

The worst case would be O(n) if we did have to reallocate.
If we didn't have to reallocate then it would be O(1).

16. (4 points) Taking this all into account, what situations would an ArrayList be the appropriate data structure for storing your data?

When you have an unknown set size an ArrayList will be useful as it will dynamically expand.

17. (10 points) The above puzzle, while from a children's puzzle book, is actually a very interesting graph theory problem, known as the Rudrata Path or Hamiltonian Path. What is the Rudrata Path problem and how does it correspond to the above puzzle?

The Rudrata Path problem are problems of determining whether we can visit each vertex only one time (called a Hamiltonian path). The puzzle given has a task where, when navigating the starways, we must be sure to only use them once, the same rules as a Hamiltonian path, otherwise we will turn into a creature called greep.

18. (10 points) Suppose we generalized the above puzzle so that there was any number of stars, rather than just 35. Let the number of stars in the above puzzle be defined as n. If there are n stars and a line between every possible pair of stars, how many paths would an algorithm need to check in order to check to find the solution?

O(n!). We start with n paths and then subtract 1 from n every time a vertex is used.

_____ 20 points

`bogosort` attempts to sort a list by shuffling the items in the list. If the list is unsorted after shuffling, we continue shuffling the list and checking until it is finally sorted.

19. (5 points) What is the worst case run time for `bogosort`?

O(infinity)

20. (5 points) Why?

There is no guarantee that the list given will ever actually be sorted, especially with a large list. The larger a list, the less likely it is for it to be sorted.

21. (5 points) What is the average case run time for `bogosort` (Hint: think about a deck of cards)?

O(n!)

22. (5 points) Why?

There are n! permutations in an array.

23. (20 points) For each of the methods you wrote in Lab 2, figure out the time complexity of the method you wrote. To turn in this portion, attach a printout of the code and specify the time complexity of each.

_____ 20 points

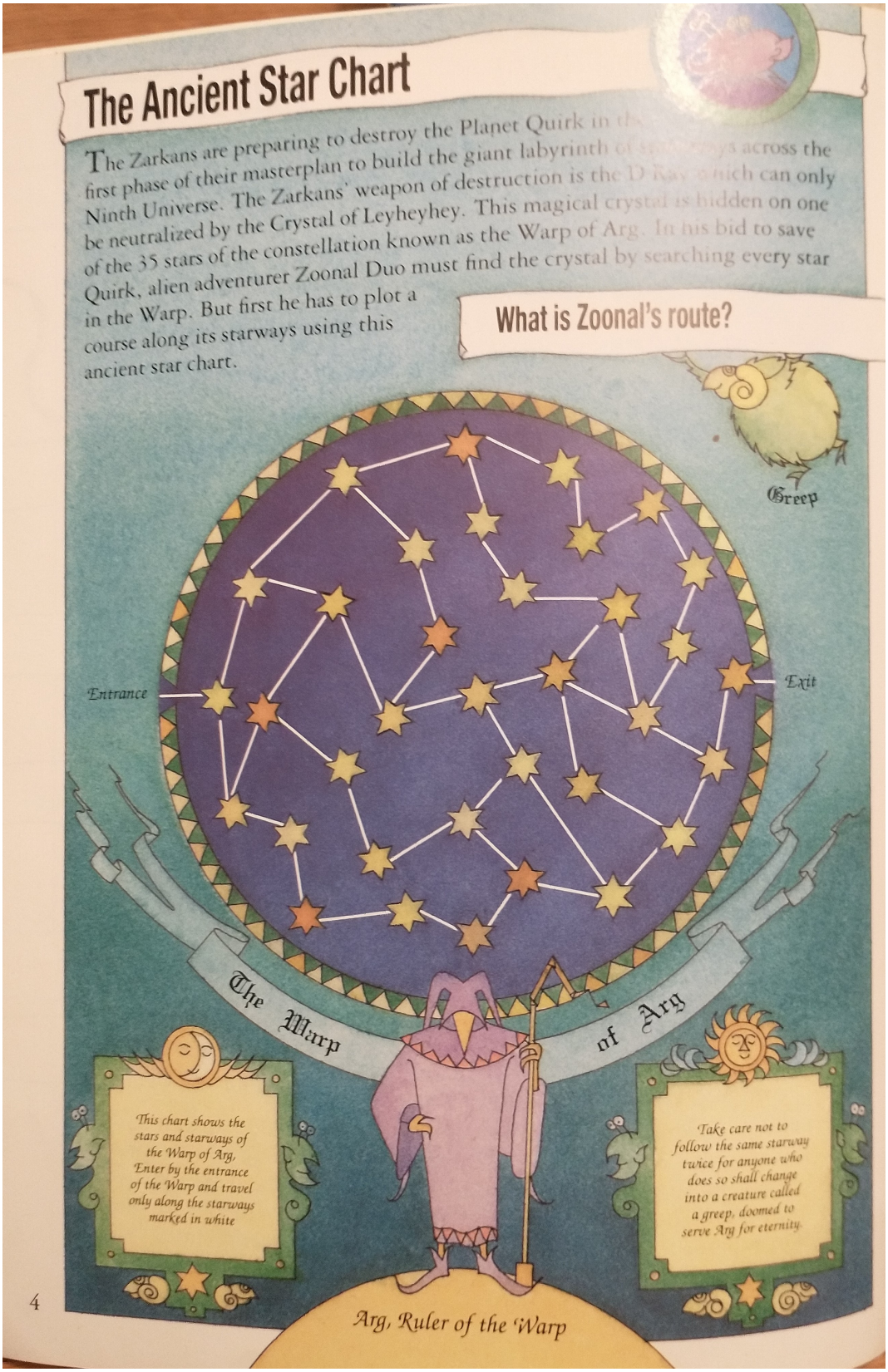

Figure 1: Please look at the above puzzle, taken from Sarah Dixon's *Map & Maze Puzzles* and answer the questions on the next page.

_____ 0 points