Final Examination

NTNU 2302 / Algorithms and Data Structures

Session II • May 12, 2022

This is the final examination of the algorithm an data structure course. It contains four parts, each having 5 questions worth 2.5 points. You have four hours.

You can describe algorithms using plain English and bullet points, pseudocode, or some well-known programming syntax (i.e., C, Python, Java, etc). Choose what you are the most comfortable with.

Note that you are *not* required to figure out "optimal" algorithms (if any exist for these problems). Rather you have to show consistency through your answers, by providing a working solution and computing its complexity/efficiency properly.

Good luck.

1 Basic Knowledge

1. Can "testing" establish the correctness of an algorithm? Why?
2. Consider the following Java function. Explain (in words) what would happen (and why) if one invokes it with $n = -3$ as argument.

```
int factorial (int n) {
  if (n==0) return 1;
  return n * factorial(n-1);
}
```

3. What is an "array" . How does it differ from a linked-list?
4. Quick-sort runs in $O(n \log n)$. Is it correct to say that it also runs in $O(n^2)$. Explain your reasoning.
5. Consider the graph shown below on Figure 1. In what order will the nodes be reached during a depth-first traversal, starting from Node A, and processing neighbor nodes in alphabetical order?

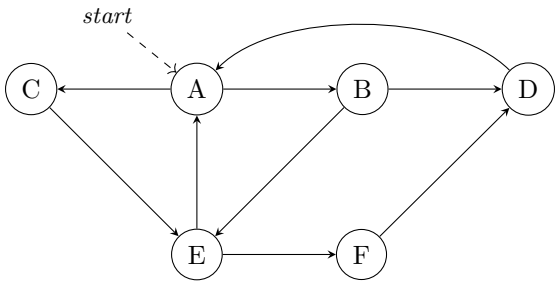

Figure 1: Sample directed graph, with 6 nodes.

2 Algorithm Analysis

The "Caesar cipher" is an old encryption algorithm used during the Roman empire. The idea is to *shift* all letters by a fixed number along the alphabet, either forward or backward. To encode and decode, we must know the secret key, which is the length of this shift. Here are a few examples assuming a 26 letters alphabet from 'a' to 'z',

- The word "abc" gets encoded as "bcd" when the key/shift is 1. The character 'a' becomes 'b' when shifted by 1, the character 'b' becomes 'c', and, 'c' becomes 'd'.
- The word "zoo" shifted by 2 gets encoded as "bqq". The second letter after 'z' is 'b' (we assume that the alphabet forms a circle). The second letter after 'o' is 'q'.
- The word "algorithm" gets encoded as "dojrulwkp" if the key/shift is 3. The third character after 'a' is 'd', the third character after 'l' is 'o' and so on and so forth.

The following Java function is one possible implementation of the Caesar cipher. It uses the ASCII character encoding, which preserves the alphabet ordering. For example, in ASCII, 'a' is encoded as 97, 'b' as 98, 'c' as 99, etc. In Java, converting a "char" to an "int" yields an ASCII code.

```
static char[] caesarCipher(char[] givenWord, int shift) {
  char[] result = new char[givenWord.length];
  for(int index=0 ; index<givenWord.length ; index++) {
    int asciiCode = (int) givenWord[index];
    int encoded = asciiCode + shift;
    if (encoded > (int) 'z') {
      encoded -= 26;
    }
    result[index] = (char) encoded;
```

```
    }
    return result;
  }
```

Questions

1. Explain how the execution unfolds given the word "zone" and a shift of 4.
2. What is the *size of the problem*, that is, what drives the runtime and memory consumption.
3. What is the best case scenario? What is the execution time efficiency/complexity. Explain your reasoning.
4. What is the worst case scenario? What is the execution time efficiency/complexity. Explain your reasoning.
5. What is the average case scenario? What is the execution time. Explain your reasoning. (Assume that every letter is equally probable).

3 Algorithm Design

Consider linked-list as shown in Figure 2, where each node points to the next one. As we build such list, it is possible to build "loops" as shown on Figure 3. We would like to design a procedure to check whether the pointers that make up the list forms a loop or not.

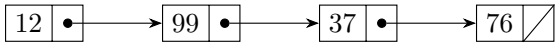

Figure 2: A "regular" linked-list.

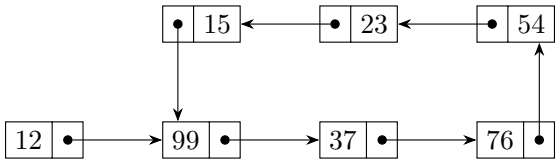

Figure 3: A "invalid" linked-list that includes a loop.

For the sake of simplicity, we will assume the existence of the following procedure that helps manipulate the nodes of these lists:

- `Node getNext (Node n) {...}` returns the "next" Node or null if it is not defined.

This task is about a procedure `boolean hasLoop (Node first) {...}`, which returns true only if the given list contains a loop.

Questions

1. Propose an "iterative" algorithm to detect such a loop. Feel free to add information to the node structure, if you feel it helps.
2. What is the time efficiency of your algorithm? Explain your reasoning.
3. What is the space efficiency of your algorithm? Explain your reasoning.
4. Convert your "iterative" algorithm into a "recursive" one.
5. Recursive algorithms leverage the "call stack" to store parameters of each active calls. What is the space efficiency of your recursive solution? Explain your reasoning.