Software Engineering Home Test

The purpose of this test is to evaluate your technical abilities in a few aspects of software engineering. It also allows common grounds for discussion in the interview that takes place after you complete the test. The test is a home test and you are encouraged to use any resource at your disposal (online, consulting friends, etc). You will be asked to explain your solution in the interview, emphasizing on correctness, design and coding decisions.

General Notes:

- We require your submission to be in Java.
- Please don't use any LLM to generate your code.
- Code can be easily delivered by email, github or both to jon.v@taboola.com
- There is no time limit - complete the test and send it back at your earliest convenience.
- Use your best judgment! We leave a lot of freedom to candidates to answer the questions: if in doubt, follow the "do something reasonable" approach.
- Where needed, please email jon.v@taboola.com for clarifications.

Question 1:

JSON is a document format used to encode information that is both human-readable and machinereadable. JSON format is explained at http://json.org. Please write a JSON parser that accepts as an input JSON and provide as an output a proper data structure. Do not use any existing library to do the parsing.
As an example, if the input is a string that contains this content:

```
Unset
{
    "debug": "on",
    "window": {
        "title": "sample",
        "size": 500
    }
}
```

Then the parser may be a static function:

```
Unset
class JsonParser {
    public static Object parse(String json) {…}
}
```

And the output may be stored inside a Java Map structure:

```
Unset
Map<String, Object> output = (Map<String,
Object>)JsonParser.parse(input)
assert output.get("debug").equals("on")
assert (Map<String,
Object>(output.get("window")).get("title").equals("sample")
assert (Map<String,
Object>(output.get("window")).get("size").equals(500)
```

Table 2

Question 2:

The following class has coding, memory and runtime inefficiencies. Locate and fix as many as you can:

```
Unset
import java.util.Date;
import java.util.List;
public class Test {
    private Date m_time;
    private String m_name;
    private List<Integer> m_numbers;
    private List<String> m_strings;
    public Test(Date time, String name, List<Integer> numbers,
List<String> strings)
    {
        m_time = time;
        m_name = name;
        m_numbers = numbers;
        m_strings = strings;
    }
    public boolean equals(Object obj) {
        try {
            return m_name.equals(((Test) obj).m_name);
        } catch (Exception e) {
            e.printStackTrace();
```

```
        }
        return false;
    }
    public String toString() {
        String out = m_name + m_numbers.toString();
        return out;
    }
    public void removeString(String str) {
        m_strings.remove(str);
    }
    public boolean containsNumber(int number) {
        return m_numbers.contains(number);
    }
    public boolean isHistoric() {
        return m_time.before(new Date());
    }
}
```

Question 3:

You're tasked with writing a spec for a generic local cache with the following property: if the cache is asked for a key that it doesn't contain, it should fetch the data using an externally provided function that reads the data from another source (eg: a database).
What features do you think such a cache should offer? How, in general lines, would you implement it?