**I AM HERE: 31:55**

# חלק ב' – תשובות לשאלות

## 1.

## 2.

1. כאשר כותבים API – צריך לתכנן אותו כך שיהיה קל לשימוש גם ללא קריאה בתיעוד בכל פעם. זו דוגמא לעיקרון בספרטניזציה שאומר שצריך למזער את התיעוד.
2. ~~עקרון חשוב בספרטניזציה הוא הקריאות של הקוד – ולפי הסרטון, API טוב צריך לאפשר לקוד שמשתמש ב-API להיות קריא.~~
3. When writing the initial spec of the API – one should keep it as short as possible (in order to allow doing modifications quikly). This follow the key rule of Spartanization – making everything as short as possible.
4. An API should be “as small as possible but no smaller”, and when in doubt whether a given functionality is needed – we should leave it out.
5. Minimize the accessibility of everything, e.g. making classes and members as private as possible.

# TMP

## 3.

Using Checked Exceptions is bad because it requires the user to deal with each such exception (catch/declare it). It turns out 99% of the times, the user wouldn’t want to deal with exceptions thrown by library code, and instead would prefer to simply get an immediate indication of the failure.

~~In the video – we learn that having checked exceptions in an API is bad because it makes implementation “leak” into the interface (if later you want to change the implementation such that different exceptions are relevant – it’s a problem).~~