Během zkoušky aktivně komunikujte se zkoušejícími, nebojte se zeptat. Nad zadáním přemýšlejte, často je těžší zjistit *co* řešit než *jak*. Kód pište čistě a průběžně ho vylepšujte. Nesnažte se vyřešit všechny příklady naráz — hodnotit budeme především podle kvality vašich myšlenek a vašeho kódu.

1. Pomocí acyklického jednosměrně zřetězeného spojového seznamu naimplementujte rozhraní `List`.
2. Upravte implementaci třídy `List` tak, aby maximálně využívala návrhový vzor nulový (prázdný) objekt.
3. Napište adaptér `Listu` `ListConcatenation`, který reprezentuje objekt odpovídající spojení dvou jiných seznamů (samozřejmě aniž by kopíroval jejich prvky).
4. Metodu `List.size` upravte tak, aby vracela instanci třídy `MutableInteger`. Tato instance by vždy měla reprezentovat aktuální velikost seznamu, ne velikost seznamu v okamžiku volání metody `size`.
5. Do rozhraní `List` doimplementujte podporu událostí "změna délky seznamu"a "změna hodnoty prvku na daném indexu".
6. Napište adaptér, který daný seznam zadaptuje na strom tvořený instancemi třídy `Node`. Adaptace by měla probíhat podle následujících pravidel:
   - Kořen stromu je uložen na indexu 0.
   - Uzel, jehož hodnota je uložená na indexu i má levé dítě uložené na indexu $2i$ a pravé na indexu $2i + 1$.
   - Pokud daný index v poli neexistuje, uzel také neexistuje a jeho rodič musí v příslušném volání vrátit `null`.
   - Hodnota uložená v nějakém uzlu stromu se rovná součtu hodnoty v poli na příslušném indexu a hodnot všech jeho potomků.