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. Naimplementujte třídu `FibonacciIterator` tak, aby postupně vracela prvky Fibonacciho posloupnosti. Fibonacciho posloupnost je nekonečná posloupnost čísel začínající na $1, 1, 2, 3, 5, \ldots$, každé následující je součtem dvou předchozích. Vaše třída by měla mít malé nároky na paměť a procesorový čas.

2. Naimplementujte třídu `FilteringAdapter`. Ta by měla vrátit jen takové prvky z iterátoru `inner`, pro které vrátil predikát `predicate` hodnotu `true`. Počítejte s tím, že iterátor `inner` může reprezentovat velmi dlouhou nebo nekonečnou posloupnost objektů.

3. Podle návrhového vzoru strom naimplementujte následující tři iterátory:

   - iterátor, který spojí několik jiných iterátorů za sebe (třída `FlatteningIterator`),
   - iterátor, který z obou poditerátorů vybere prvek, nad prvním z nich zavolá predikát a v závislosti na vrácení hodnotě první nebo druhý prvek (třída `ConditionalIterator`) a
   - iterátor, který vrací prvky z poditerátoru tak dlouho, dokud predikát vrací `true` (třída `LoopIterator`).

4. Pomocí návrhového vzoru stav naimplementujte třídu `CloseableIterator`. `CloseableIterator` vrací prvky z poditerátoru zadaného v konstruktoru tak dlouho, dokud uživatel nezavolá `close`. Při zavolání `close` je ukazatel na poditerátor smazán.

5. Upravte předchozí kód tak, aby používal generické třídy. Generické parametry volte tak, aby uživateli umožňovaly co největší flexibilitu.

6. Naimplementujte metodu `Meta.createEventListener`, která pomocí dynamické proxy (třída `java.lang.reflect.Proxy`) vytvoří a vrátí instanci implementující rozhraní `EventSource` a rozhraní reprezentované parametrem `iface`. Vysledná instance by měla (po zaregistrování na nějakém zdroji událostí) přeposílat všechny události (transformované do instancí třídy `Event`) svým posluchačům. Můžete předpokládat, že všechny metody rozhraní `iface` vrací `void`.