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 metodu `static BinaryDigit Main.add(BinaryDigit first, BinaryDigit second)`, která vrátí součet čísel reprezentovaných proměnnými `first` a `second`.

2. V implementaci čísel použijte návrhový vzor nulový objekt (tzn. upravte existující kód tak, aby v atributu `next` nikdy nebyla hodnota `null`).

3. Napište implementaci metody `static UnaryDigit Main.convertToUnary( BinaryDigit num)`, která vrátí unární reprezentaci čísla `num`. Nevytvářejte jednotlivé uzly této reprezentace naráz, ale jen "na požádání".

4. Dopište metodu `static BinaryDigit Main.xor(BinaryDigit first, BinaryDigit second)`, která vrátí po bitech provedený exkluzivní součet čísel `first` a `second`. Pokud je jedno z čísel kratší, doplňte ho nulami. Při implementaci použijte dvojité volání (double dispatch).

5. Napište metodu `static int Main.getValue(BinaryDigit num)`, která vrátí hodnotu čísla `num`. Svoji implementaci založte na použití metody `Main.fold`, vyhněte se použití operátoru `instanceof` a metody `Object.getClass`.

6. Instance implementující rozhraní `Node` reprezentují dokonale vyvážené stromy, tzn. stromy, ve kterých má každý list stejnou hloubku. Doimplementujte jeho metody ve třídách `Leaf` a `InnerNode`, snažte se o efektivní implementaci.