Comparable<T> szerepe

Ahhoz, hogy egyetlen rendezőalgoritmus általános adattípusokon tudjon működni, két dologra van szükség: egy rendezésre, amely egy bizonyos adattípuson működik, és egy módszerre, amely minden más adattípust erre a rendezhető típusra tud konvertálni, mely módszert a rendezőalgoritmus bemenetként fogad. Ez onnan látszik, hogy ha akár minden egyes adattípust lehetséges értékeinek sorrendiségét beépítve tudná, akár egyetlen adattípusra konvertálna, de a konvertálási módszert beépítve tárolná, akkor egyetlen új adattípus létrehozása elég lenne ahhoz, hogy a rendezőalgoritmust át kelljen írni, hogy ismét általánossá váljon.

A java.util.Collections.sort ilyen módon működő általános adattípuson működő algoritmus. A beépítetten rendezhető adattípus természetes módon az egész számok, melyek között egyértelmű a sorrendiség (az egyenlőség kivételével, de ez a metódus alapértelmezetten Timsort-ot használ, ami stabil, így tartja a sorrendiséget egyenlő elemek között), valamint nagyon gyorsan összehasonlíthatóak. A konvertálást a „Csapat” adattípus esetében úgy végezzük el, hogy a csapatok értékét hasonlítjuk össze, melyek számok, így hasonlóan egyszerűen összehasonlíthatóak.

Mivel a java legfőképpen objektumorientált programozási nyelv, értelmes választás, hogy a konvertálást maga a Csapat adattípus végezze el, és hogy annak érdekében, hogy a Collections.sort tudja paraméterként fogadni ezt a típust, egy interface implementációval tegye ezt, mely egyben garantálja is, hogy a Csapat típus valóban el is végezze a konvertálást.