A házi feladatot egy `Homework8` nevű modulként kell beadni. Figyeljetek arra, hogy a függvényeitek a module szóval egy "oszlopba" kerüljenek, azaz ne legyenek beljebb húzva! Minden definiálandó függvényhez adjuk meg a hozzá tartozó típus szignatúrát is! (Ezt most megadtam, a saját modulotokba is másoljátok be a definíciótok elé.)
Az alábbi feladatokat rekurzió és listagenerátorok NÉLKÜL írjátok meg! Használható függvények (Néhány függvény a `Data.List` modulban található):

```
head, tail, init, last, map, filter, take, drop, takeWhile, dropWhile, zip, span, (!!)
zipWith, unzip, (++), (:), repeat, replicate, group, groupBy, sort, sortOn, sum, length,
minimum, maximum, minimumBy, maximumBy, nub, nubBy, (.), ($), concat, concatMap,
elem, elemIndex, notElem, find, lookup, permutations, tetszőleges függvény NEM lista
típusú kifejezésekre (+, -, <, ==, not, fst, snd stb.)
```

Típusszinonímák definiálása

- Definiáljuk a `Salary` típusszinonímát ami egy ember fizetését reprezentálja. A típusszinoníma `Int`-el legyen ekvivalens.
- Definiáljuk az `Expenses` típusszinonímát ami egy ember költségeit reprezentálja. A típusszonínma `[Int]`-el legyen ekvivalens.

Függvények

- Definiáljunk egy függvényt, amely egy fizetések és költségek listáját kapja paraméterül. A függvény adja vissza azoknak az embereknek a maradék pénzét, akik kevesebbet költöttek mint a fizetésük. (`moneyRemaining :: [Salary] -> [Expenses] -> [Int]`)
- Definiáljunk egy függvényt, amely visszaadja azon emberek fizetését, akik ki tudnák fizetni a paraméterül kapott költséget legfeljebb 3 fizetésükből. (`max3 :: [Salary] -> Expenses -> [Salary]`)
- Definiáljunk egy függvényt, amely visszaadja a 3. legkisebb fizetésű ember fizetését azok közül, kiknek a fizetése nagyobb vagy egyenlő mint az átlag fizetés. (`third :: [Salary] -> Salary`)
- Definiáljunk egy függvényt, amely az 5. legnagyobb fizetésű embernél nagyobb költségű emberek fizetésének összegét visszaadja. **Megjegyzés:** A feladat szempontjából két ugyanannyi fizetésű ember 1 embernek számít. (`bigSpenders :: [Salary] -> [Expenses] -> Int`)

Tesztek:

```
moneyRemaining [1000, 2000, 3000] [[500, 500], [600, 600], [2000, 500, 600]] == [800]
moneyRemaining [1000, 2000, 1000, 2000] [[], [0], [1100, 900], [1100, 899, 1]] == [1000,
2000]
max3 [1000, 2000, 3000] [3000, 2000, 1000] == [2000, 3000]
max3 [(-1000)] [0] == []
third [1000, 2000, 3000, 4000, 5000, 6000] == 6000
third [1000, 2000, 3000, 4000, 5000, 6000, 7000] == 6000
third [1000, 2000, 3000, 4000, 5000, 7000] == 7000
bigSpenders [1000, 2000, 3000, 4000, 5000] [[], [1000, 2000], [100], [0], [1200]] == 7000
bigSpenders [0, 1000, 1000, 2000, 3000, 4000] [[0,0], [1], [], [], [], [0]] == 1000
```