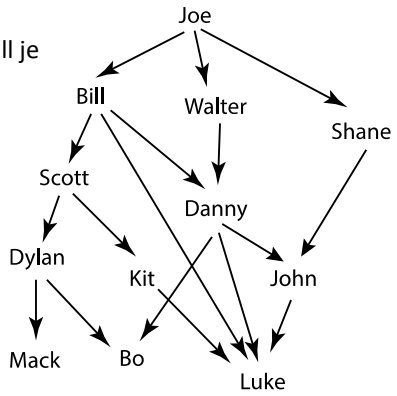
1. As usual, Bill

Sestavi razred `Bar` s primernim konstruktorjem in metodami

- `naroci(kdo, kaj)`, ki zabeleži, da je kavboj `kdo` naročil pijačo `kaj`. `kaj` je lahko tudi `None`; v tem primeru dobi pijačo, ki jo je doslej naročil največkrat. Predpostaviti smeš, da je `kdo` že naročal in da je pijača, ki jo je naročil največkrat, le ena.
- `najpogostejse(kdo)` vrne ime pijače, ki jo `kdo` najpogosteje naroča
- `bilanca(kdo)` vrne slovar, katerega ključi so imena osvežilnih napitkov, ki jih je v preteklosti naročal `kdo`, pripadajoče vrednosti pa povedo, kolikokrat je naročil posameznega, na primer, `{"whisky": 63, "gin": 16}`.

Nasvet: poglej v dokumentacijo Pythona, kaj ti nudi `collections.Counter`. Del dokumentacije je na drugi strani izpita.

2. The Fastest Gun in the West



Da se ne bodo stalno streljali, so se odločili s slepo municijo preskusiti, kdo je hitrejši. Bill je hitrejši od Scotta. Danny je hitrejši od Boja. In od Luke. In Johna. Rezultate zabeležijo v seznam parov imen: `[("Bill", "Scott"), ("Danny", "Bo"), ("Danny", "Luke"), ("Danny", "John")]` – prvi element terke je hitrejši od drugega. Testi vsebujejo daljši seznam; podatki so narisani na desni.

Napiši funkcijo `je_hitrejsi_od(premaga, kdo, koga)`, ki vrne `True`, če je kdo hitrejši od koga. Klic `je_hitrejsi_od(premaga, "Joe", "Bill")` vrne `True`. Klic `je_hitrejsi_od(premaga, "Walter", "Bo")` prav tako vrne `True`, ker je Walter hitrejši od Dannyja, Danny pa od Boja. Klic `je_hitrejsi_od(premaga, "Bo", "Walter")` vrne `False`, ker je v resnici Walter hitrejši od Boja. Klic `je_hitrejsi_od(premaga, "Shane", "Kit")` vrne `False`, ker tega ne vemo: oba sta počasnejša od Joea in oba sta hitrejša od Luka, ali je Shane hitrejši od Kita, pa ni znano.

Formalno: A je hitrejši od B, če je zmagal v dvoboju, ali pa je A zmagal v dvoboju z nekim C, ki je hitrejši od B.

3. Cowboy Polka

Kot v vsakem mestu na divjem zahodu je tudi tu hiša z rdečimi tapetami, v kateri gospe vsak večer vabijo na ples. Vsak kavboj ima prioritetno listo plesalk. Želje so podane v seznamu takšne oblike:

```
zelje = (("Joe", ("Ana", "Cilka", "Berta")),
         ("Bill", ("Berta", "Cilka", "Dani", "Ema")),
         ("Shane", ("Ema", "Cilka")),
         ("Walter", ("Ana", "Cilka", "Ema", "Fanny")),
         ("Mack", ("Cilka", "Berta", "Greta", "Fanny")))
```

Joe ima najraje Ano; če Ane ni, Cilko, sicer Berto. Bill ima najraje Berto, če nje ni, pa Cilko, sicer Dani, sicer Emo. In tako naprej. Plesalke izbirajo v takšnem vrstnem redu, kot so našteti: najprej Joe, nato Bill, Shane in Walter. Joe, Bill in Shane torej dobijo prvo izbiro, Walter Cilko (ker je Ano pograbil Joe), Mack pa Greto (ker sta Cilko in Berto pobrala Walter in Bill).

Napiši funkcijo `soplesalka(zelje, kavboj)`, ki dobi seznam želj, kot je gornji, in ime kavboja, kot rezultat vrne ime njegove soplesalke. Funkcija ne sme predpostavljati, da so imena kavbojev in plesalk natančno takšna kot v gornjem primeru.

4. Dead Men Don't Shoot

Bill grozi, da bo ustrelil Dannyja in potem še Kita, Walter kani potem ustreliti Dannyja (ki ga to sicer ne bo zmotilo, saj bo že mrtev), John pravi, da bo ustrelil Luka, Kit se priduša, da bo nato tudi on ustrelil Luka, potem pa še Joeja. Joe bo pospravil Dylana. To predstavimo tako: `[("Bill", "Danny"), ("Bill", "Kit"), ("Walter", "Danny"), ("John", "Luke"), ("Kit", "Luke"), ("Kit", "Joe"), ("Joe", "Dylan")]`.

Dobra novica za Joeja je, da ga Kit ne bo mogel ustreliti, ker bo Kita že prej pospravil Bill. Slaba novica za Dylana je, da bo Joe preživel in ga ustrelil.

Napiši funkcijo `streljanje(groznje)`, ki prejme seznam groženj, kot je gornji, in vrne število krst, ki jih je potrebno pripraviti. (Se pravi: število ustreljenih.)

OBRNI!

5. The True Glory of Cow Boys

Na koncu koncev pa so (bili) kavboji bolj ali manj priganjači živine. Napiši funkcijo `uspesnost(ime_datoteke)`, ki prejme ime datoteke s težami krav. Vsaka vrstica se nanaša na eno kravo: prva beseda je ime lastnika, druga teža krave. Funkcija naj vrne slovar, katerega ključi so imena kavbojev, vrednosti pa skupne teže njihovih krav.

Če datoteka vsebuje

```
Bill 578
Joe 641
Bill 385
Bill 684
Joe 483
Scott 684
```

mora funkcija vrniti `{"Bill":1647, "Joe": 1124, "Scott": 684}`. Pri tem je, recimo 1647 vsota 578 + 385 + 684.

`Counter` objects

A counter tool is provided to support convenient and rapid tallies. For example:

```
>>>
```

```
>>> # Tally occurrences of words in a list
>>> cnt = Counter()
>>> for word in ['red', 'blue', 'red', 'green', 'blue', 'blue']:
...     cnt[word] += 1
>>> cnt
Counter({'blue': 3, 'red': 2, 'green': 1})

>>> # Find the ten most common words in Hamlet
>>> import re
>>> words = re.findall(r'\w+', open('hamlet.txt').read().lower())
>>> Counter(words).most_common(10)
[('the', 1143), ('and', 966), ('to', 762), ('of', 669), ('i', 631),
 ('you', 554),  ('a', 546), ('my', 514), ('hamlet', 471), ('in', 451)]
```

class `collections.`**`Counter`**([*iterable-or-mapping*])

A `Counter` is a `dict` subclass for counting hashable objects. It is a collection where elements are stored as dictionary keys and their counts are stored as dictionary values. Counts are allowed to be any integer value including zero or negative counts. The `Counter` class is similar to bags or multisets in other languages.

`most_common`([*n*])

Return a list of the *n* most common elements and their counts from the most common to the least. If *n* is omitted or `None`, `most_common()` returns *all* elements in the counter. Elements with equal counts are ordered in the order first encountered:

```
>>>
```

```
>>> Counter('abracadabra').most_common(3)
[('a', 5), ('b', 2), ('r', 2)]
```