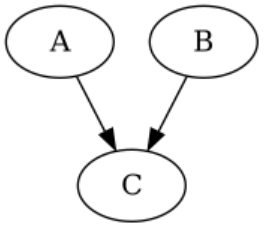
Řešení úlohy č. 2
Recepty

V úloze máš zadaný seznam receptů, a seznam požadavků. Tvým úkolem je spočítat, kolik nejméně surovin potřebuješ abys vyrobil všechny požadavky. Máš zaručeno, že řešení vždy bude existovat, a že bude vždy jednoznačné.

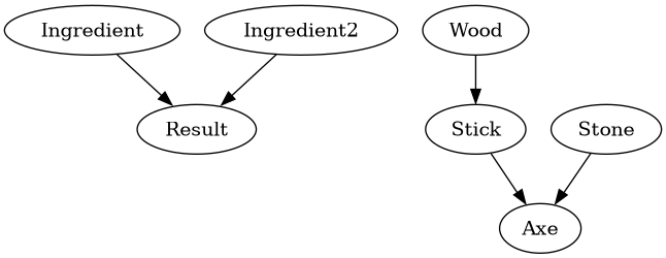
Formálně si můžeme zadání vyjádřit jako graf. Vrcholy grafu budou všechny vstupy a výstupy receptů, hrany budou definovány právě našimi recepty – pro recept, který vyrobí ze surovin `A` a `B` produkt `C`, vyrobíme dvě orientované hrany, `(A, C)` a `(B, C)`:



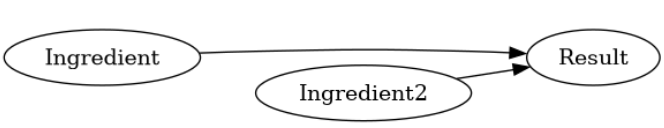
Takhle můžeme všechny recepty pospojovat dohromady, a vyrobit si tím stromovou reprezentaci našich receptů:

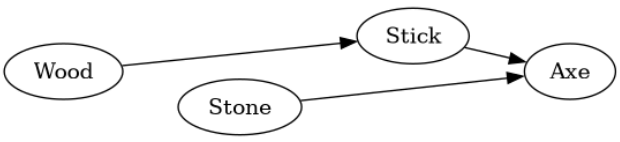
```
Recepty:

2
3 Ingredient
2 Ingredient2
2 Result

1
4 Wood
6 Stick

2
2 Stick
2 Stone
1 Axe
```

Stromy receptů:



Nyní na tento graf zavoláme algoritmus `topsort`, abychom si vytvořili topologické uspořádání tohoto grafu. Topologickým uspořádáním myslíme takové pořadí vrcholů tohoto grafu, ve kterém platí, že pro každou hranu `(a, b)` v tomto grafu, je ve výsledném uspořádání `a` uspořádáno před `b`:





Nejdříve si můžeme projet seznam požadavků, a každý požadavek si zapsat do příslušného vrcholu. Poté můžeme tento graf projít v opačném pořadí topologického uspořádání ("od konce k začátku") [1]), a postupně si vyplňovat potřebné počty v jednotlivých vrcholech.

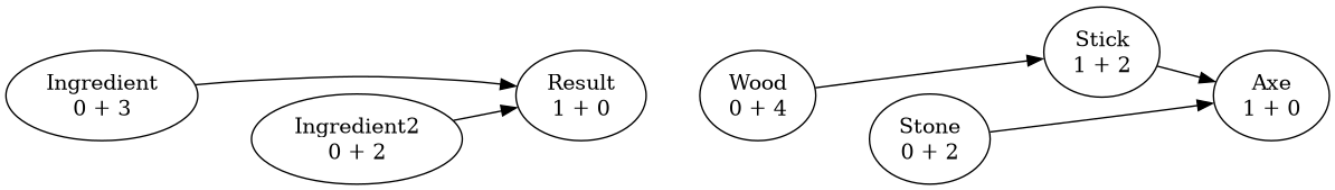
[1]) více v Průvodci labyrintem algoritmů - `http://pruvodce.ucw.cz/static/pruvodce.pdf#page=128`

Jelikož jdeme v topologickém uspořádání pozpátku, je zaručeno že když budeme vyplňovat potřebný počet položky `A`, budou už vyplněny všechny položky které obsahují `A` ve svém seznamu surovin. Je tudíž bezpečné pro každý vrchol `A` projít všechny jeho odchozí hrany, pomocí příslušných receptů si napočítat počet suroviny `A` který vyžaduji a zapsat ho do tohoto vrcholu. Je zde třeba dbát zřetel na to, abychom správně pracovali s recepty a zaokrouhlovali na celočíselný násobek až když máme plný součet kolik dané suroviny potřebujeme.

Například pro recepty nahoře a takovýhle seznam požadavků:

```
1 Result
1 Axe
1 Stick
```

není správné se podívat na recept na výrobu `Axe`, říct, že potřebujeme 2 `Stick` a tudíž 4 `Wood`, a poté se podívat na požadavek 1 `Stick` a přidat další 4 `Wood`. Naopak je nutné nejdříve sečíst 2 `Stick` pro výrobu `Axe` a 1 `Stick` pro výrobu, z čehož víme, že nám stačí 4 `Wood`. Vyplněný graf bude ve výsledku vypadat takto:



Zde si můžeme všimnout, že našimi hledanými surovinami budou všchny vrcholy, do kterých nevede žádná hrana, tj. v našem příkladu `Ingredient`, `Ingredient2`, `Wood` a `Stone`. Tyto suroviny si abecedně seřadíme a vypíšeme seznam `pocet surovina`:

```
4
3 Ingredient
2 Ingredient2
2 Stone
4 Wood
```

Analýza časové složitosti

Pro přehlednost, časovou složitost řešení budeme uvádět relativně k velikosti grafu, specificky k počtu vrcholů n (počet různých vstupů nebo výstupů receptů), a počtu hran m (součet počtu surovin přes všechny recepty). U první části se odkážeme na to, že algoritmus `topsort` běží v čase $O(n+m)$. Ve druhé části využijeme toho, že graf procházíme v opačném topologickém uspořádání, a tudíž každý vrchol navštívíme právě jednou. Z každého vrcholu také procházíme pouze jeho odchozí hrany, tudíž i tato část běží v čase $O(n+m)$. Lze tedy prohlásit, že celé řešení běží v čase $O(n+m)$.