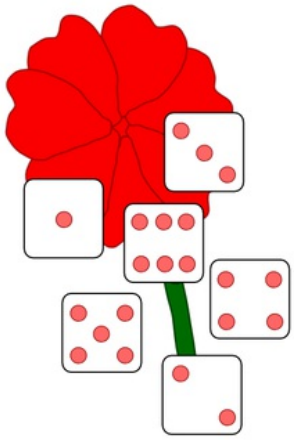
Petals around the Rose

The thinking game *Petals around the Rose* is played with five dice. It is played by a computer program or with real dice that are thrown by a *Potentate of the Rose*, a person who knows the secret behind the game. For each roll of the dice, there is a unique numerical solution. The players have to try to figure out the right solution through inductive reasoning. If they fail to guess the right solution, the correct solution is communicated by the *Potentate of the Rose*, and they must try to guess the right solution at the next throw.



The game has only three rules:

- The name of the game is Petals around the Rose, and this name is important.
- The answer is always a non-negative even integer.
- Anyone who knows the secret of the game, is allowed to give the right solution that corresponds to a particular throw, but the secret must never be revealed.

Assignment

In order not to reveal the secret of the game we have stored the correct answer to the question how many petals there are around the rose in the `flowerpetals.txt` file. Each line of this file contains six integers, which are separated by a single space from each other. The first five integers each time indicate the number of eyes for a possible roll of five dice, while the sixth number indicates to how many petals this throw corresponds. Because the number of petals is independent of the order of the dice (oops, now we have already revealed a part of the solution) we have sorted the first five numbers each in ascending order, so that we only need to include a single line in the file if the order of the dice is the only thing that differs.

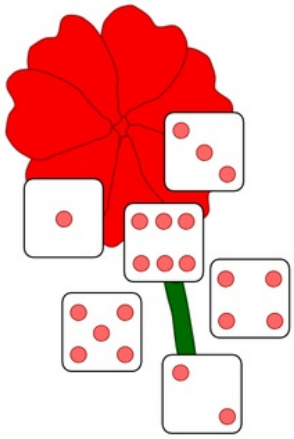
- Write a function `readSolutions` reads the solutions for the different combinations from a text file, and uses them to construct a dictionary that is returned by the function. The location of the text file must be passed as an argument to the function. For each line of the file, the first five numbers form the elements of a tuple that is used as the key in the dictionary that is to be returned by the function as a result, and is the sixth number the value that corresponds to this key.
- Write a function `potentate` that for a given roll of five dice indicates to how many petals around the rose the throw corresponds. Two arguments must be passed to this function. The first argument is a list of five integers between 1 and 6 (limits included). The elements of this list correspond to the number of eyes for a roll of five dice and are listed in random order. The second argument is a dictionary that contains the solution for every possible

throw of five dice (such as those returned by the function readSolutions).

Example

```
>>> solutions = readSolutions('flowerpetals.txt')
>>> solutions
{(1, 1, 1, 2, 3): 2, (5, 5, 5, 5, 6): 16, ..., (1, 3, 4, 4, 4): 2}
>>> potentate([3, 5, 5, 4, 3], solutions)
12
>>> potentate([6, 5, 3, 4, 2], solutions)
6
```

Het denkspel *Bloemblaadjes rond de Roos* wordt gespeeld met vijf dobbelstenen. Het wordt gespeeld door een computerprogramma of met echte dobbelstenen die worden geworpen door een *Potentaat van de Roos*, een persoon die het geheim achter het spel kent. Voor elke worp met de dobbelstenen is er een unieke numerieke oplossing. De spelers moeten de juiste oplossing proberen te achterhalen via inductief redeneren. Indien ze er niet in slagen om de juiste oplossing te raden, wordt de juiste oplossing meegedeeld door de *Potentaat van de Roos*, en moeten ze proberen om bij een volgende worp de juiste oplossing te raden.



Het spel kent slechts drie regels:

- De naam van het spel is *Bloemblaadjes rond de Roos*, en deze naam is belangrijk.
- Het antwoord is steeds een niet-negatief even geheel getal.
- Iedereen die het geheim van het spel kent, mag de juiste oplossing geven die correspondeert met een bepaalde worp, maar mag het geheim zelf nooit prijsgeven.

Opgave

Om het geheim van het spel niet te onthullen, hebben we het correcte antwoord op de vraag hoeveel bloemblaadjes er rond de roos staan opgeslaan in het bestand bloemblaadjes.txt. Elke regel van dit bestand bevat zes natuurlijke getallen, die telkens van elkaar worden gescheiden door één enkele spatie. De eerste vijf getallen geven telkens het aantal ogen voor een mogelijke worp met vijf dobbelstenen, terwijl het zesde getal aangeeft met hoeveel bloemblaadjes deze worp correspondeert. Omdat het aantal bloemblaadjes onafhankelijk is van de volgorde van de dobbelstenen (oeps, nu hebben we toch al een deel van de oplossing onthuld) hebben we de eerste vijf getallen telkens in oplopende volgorde gesorteerd, zodat we voor worpen waarbij enkel de volgorde van de dobbelstenen verschilt slechts één enkele regel in het bestand moeten opnemen.

- Schrijf een functie leesOplossingen die de oplossingen voor de verschillende combinaties inleest uit een tekstbestand, en daarmee een dictionary opbouwt die als resultaat door de functie moet teruggegeven worden. De locatie van het tekstbestand moet als argument aan de functie doorgegeven worden. Voor elke regel uit het bestand vormen de eerste vijf getallen de elementen van een tuple dat als sleutel gebruikt wordt in de dictionary die als resultaat door de functie moet teruggegeven worden, en vormt het zesde getal de waarde die correspondeert met deze sleutel.
- Schrijf een functie potentaat die voor een gegeven worp van vijf dobbelstenen aangeeft met hoeveel bloemblaadjes rond de roos deze worp correspondeert. Aan deze functie moeten twee argumenten doorgegeven worden. Het eerste argument is een lijst van vijf natuurlijke getallen tussen 1 en 6 (grenzen inbegrepen). De elementen van deze lijst corresponderen met het aantal ogen voor een worp met vijf dobbelstenen, worden in willekeurige volgorde opgelijst. Het tweede argument is een dictionary die de oplossing voor elke mogelijke worp met vijf dobbelstenen bevat (zoals die wordt teruggegeven door de functie leesOplossingen).

Voorbeeld

```
>>> oplossingen = leesOplossingen('bloemblaadjes.txt')
>>> oplossingen
{(1, 1, 1, 2, 3): 2, (5, 5, 5, 5, 6): 16, ..., (1, 3, 4, 4, 4): 2}
>>> potentaat([3, 5, 5, 4, 3], oplossingen)
12
>>> potentaat([6, 5, 3, 4, 2], oplossingen)
6
```