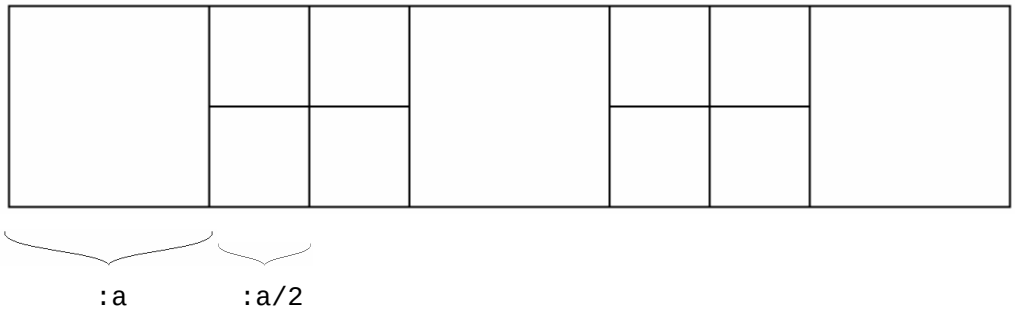
1. zadatak COKO 30 bodova

Napišite naredbu `COKO :a :n` koja briše ekran i crta redak čokolade. Redak čokolade se sastoji od `:n` kvadrata (kockica čokolade) stranice `:a`, koji međusobno dijele jednu stranicu (vidi sliku). Parni kvadratići trebaju biti podijeljeni na četiri manja kvadrata jednakih dimenzija.

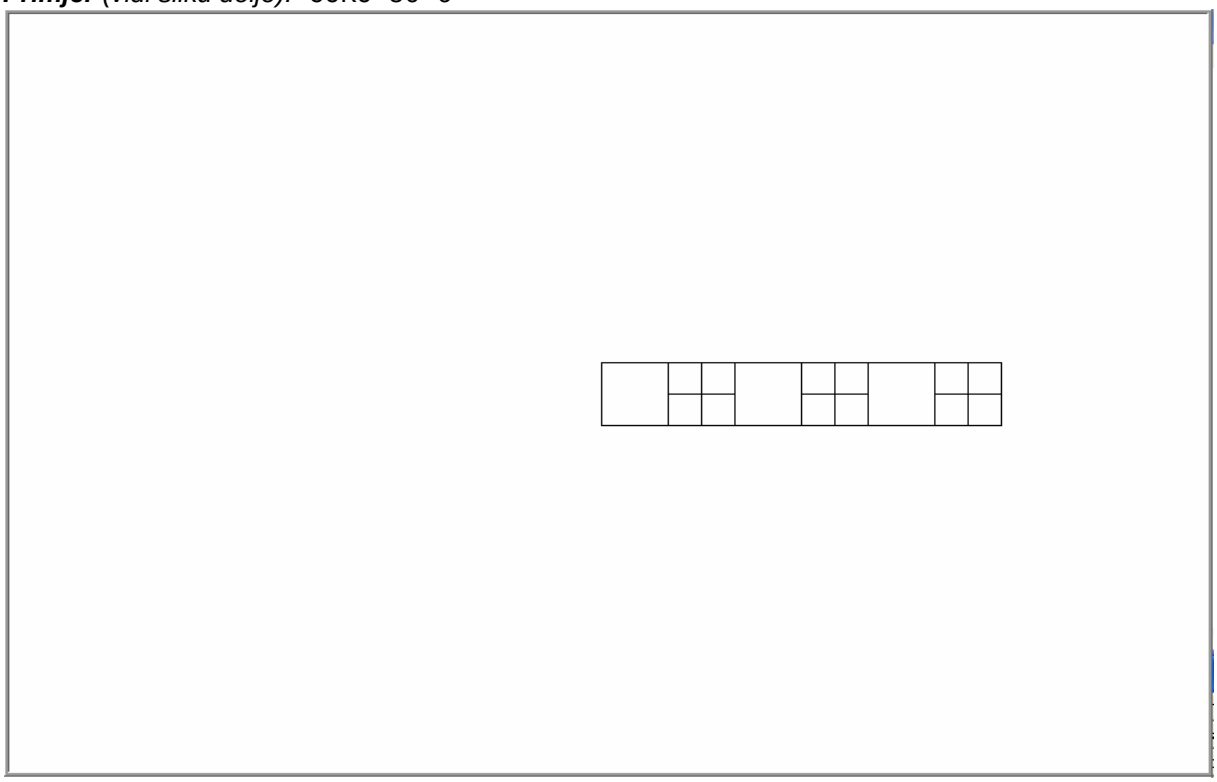
Na slijedećoj slici je primjer za `COKO 100 5`.



Pozicija lika na ekranu nije bitna.

Parametri `:a` i `:n` će biti prirodni brojevi.

Primjer *(vidi sliku dolje):* `COKO 50 6`

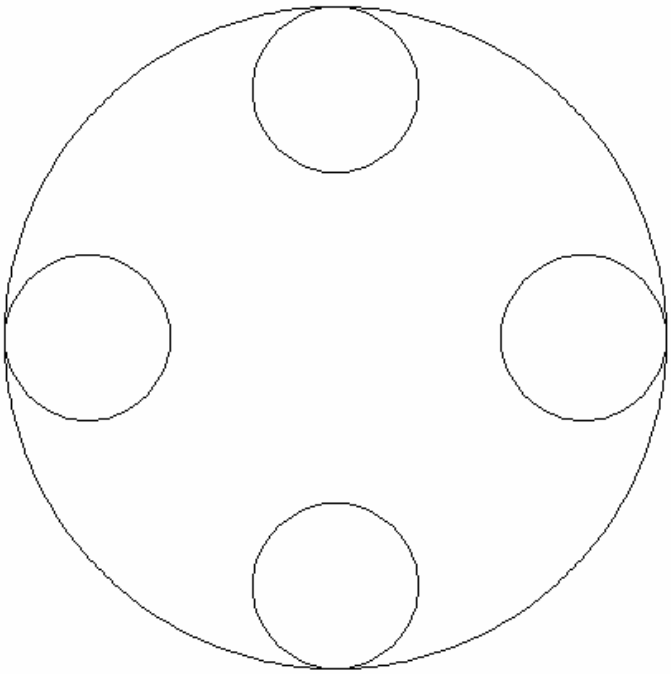


Program snimite pod imenom **COKO.LGO**.

2. zadatak KRUGOVI 40 bodova

Napišite naredbu `KRUGOVI :r :s :n` koja crta kružnicu radijusa `:r`. Unutar te kružnice se mora nalaziti `:n` manjih kružnica radijusa `:s`. Unutarnje kružnice moraju dirati vanjsku i moraju biti pravilno raspoređene, tj. njihova središta su vrhovi pravilnog `:n`-terokuta.

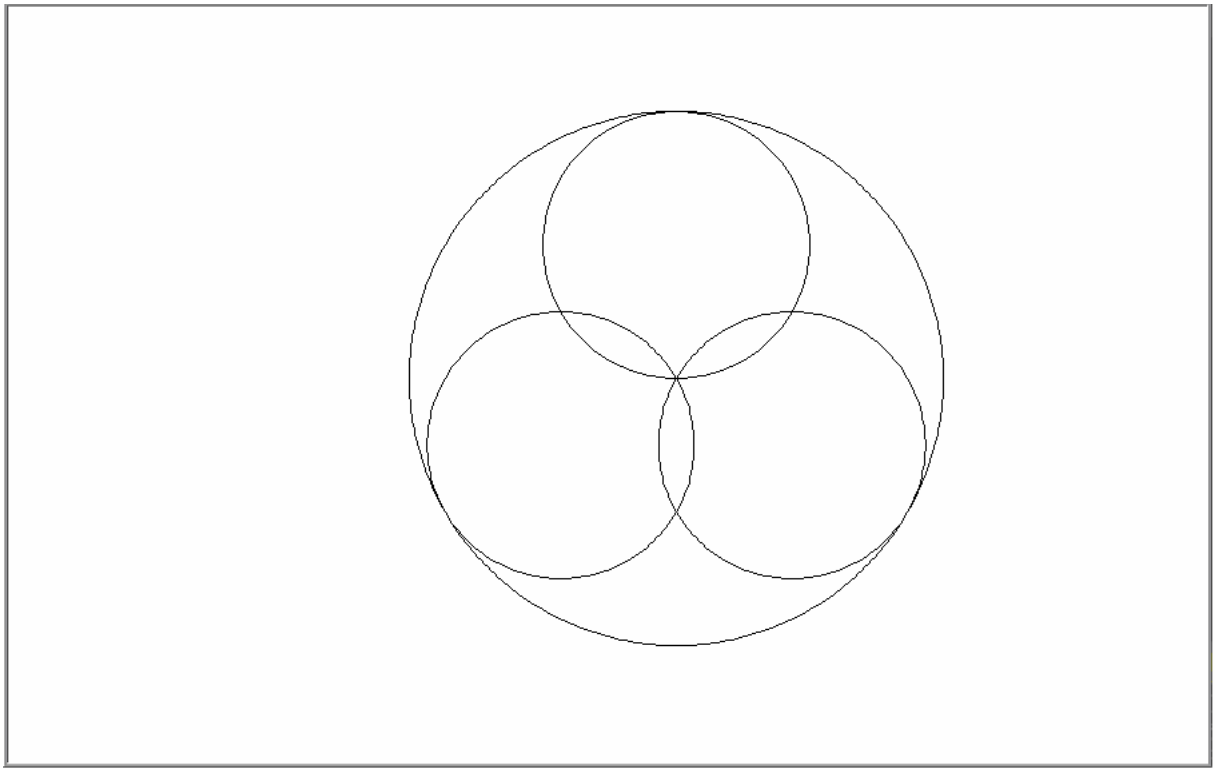
Na slici desno je primjer za `KRUGOVI 200 50 4`.



Pozicija lika na ekranu nije bitna.

Radijusi `:r` i `:s` su prirodni brojevi pri čemu je `:s` manji od `:r`. Parametar `:n` je nenegativan broj.

Primjer *(vidi sliku na drugoj stranici):* `cs KRUGOVI 200 100 3`



Program snimite pod imenom **KRUGOVI.LGO**.

3. zadatak TEZISTE 60 bodova

Napišite naredbu `TEZISTE :n :d` koja briše ekran i crta pravilni `:n`-terokut sa stranicom duljine `:d`.

Težišnica je dužina koja spaja vrh s polovištem nasuprotne stranice.

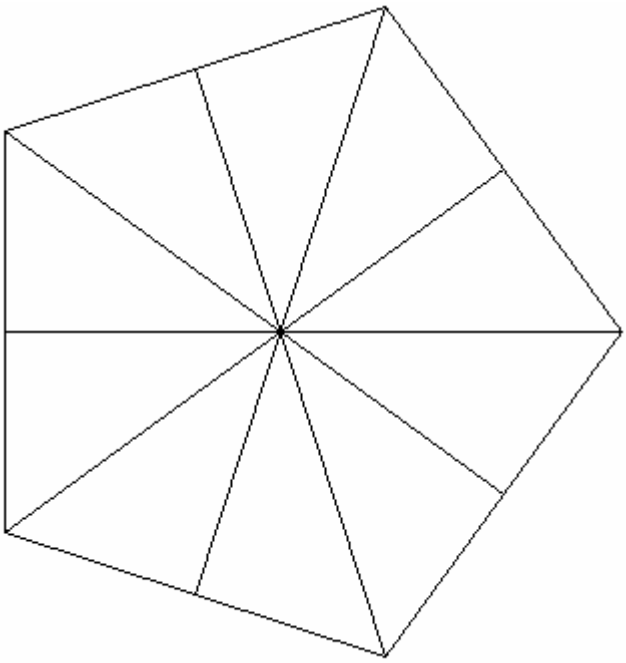
U nacrtani mnogokut potrebno je upisati sve njegove težišnice.

Mnogokut će uvijek imati neparan broj vrhova, kako bi postojala točno jedna nasuprotna stranica svakom vrhu.

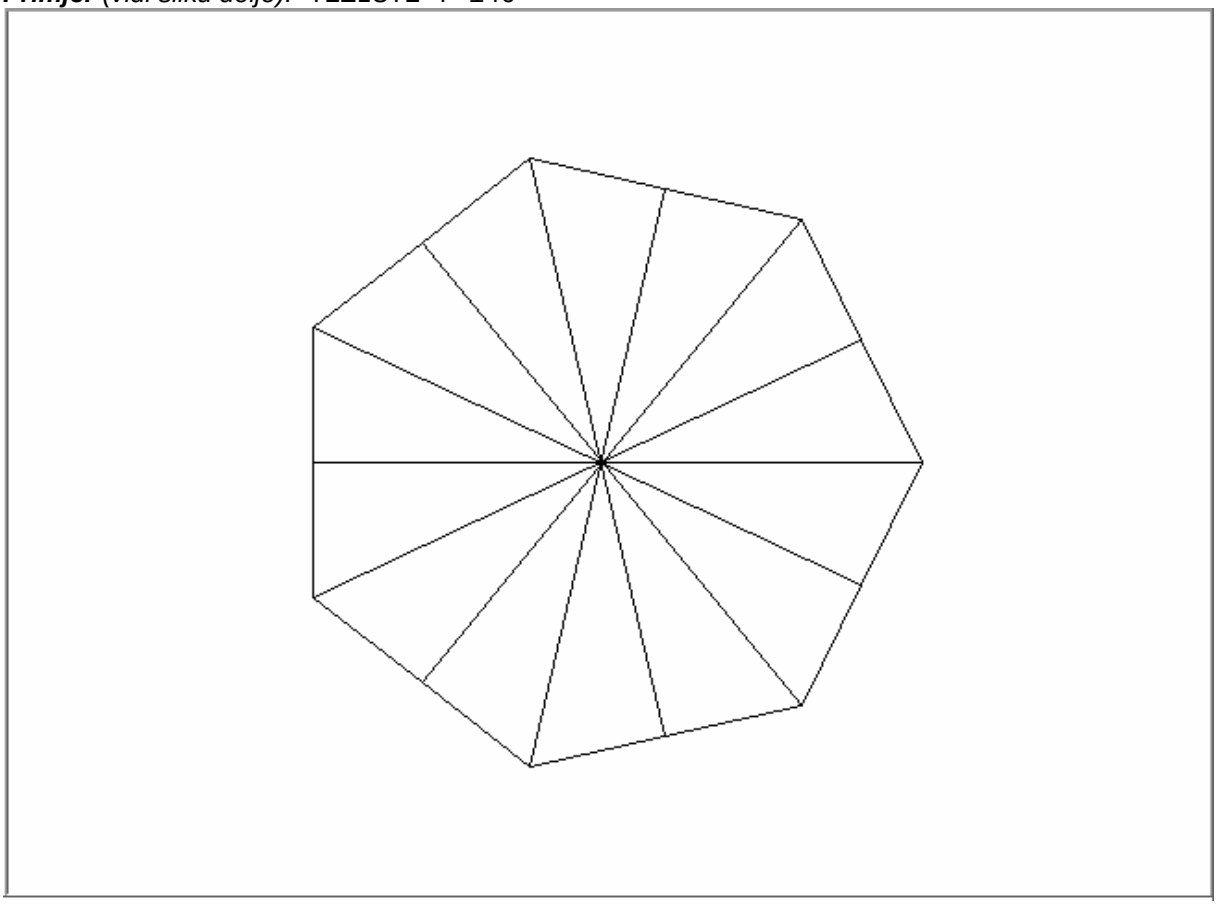
Na slici desno je primjer za `:n` = 5 i `:d` = 180.

Pozicija lika na ekranu nije bitna.

Broj `:n` će biti prirodan broj veći ili jednak 3, a broj `:d` će biti prirodan broj.



Primjer *(vidi sliku dolje):* `TEZISTE 7 140`



Program snimite pod imenom **TEZISTE.LGO**.

4. zadatak SUMA 70 bodova

Napisati funkciju `SUMA :L` koja prima listu `:L` kao parametar. Ta lista može sadržavati brojeve i druge liste unutar sebe, a one mogu sadržavati brojeve i još neke liste i tako u nedogled. Svi brojevi će biti cijeli i po apsolutnoj vrijednosti manji od 100.

Suma liste je zbroj svih elemenata te liste, dakle zbroj svih brojeva i suma podlisti. Na primjer:

lista [1 2 3] ima sumu 6,

lista [-1 [2] 3] ima sumu 4, i

lista [1 [2 [[3]] 4] [5 6] 7] ima sumu 28.

Iz zadane liste `:L` vadimo sve moguće podliste. Na primjer, iz liste [1 2 [4 5 [6]] [-3 -4] 7] možemo izvaditi slijedeće podliste:

[1 2 [4 5 [6]] [-3 -4] 7] – ovo je sama zadana lista,

[4 5 [6]] – prva podlista iz zadane liste,

[6] – jedina podlista prethodne, i

[-3 -4] – druga podlista iz zadane liste.

Za svaku od ovih podlista možemo izračunati sumu te liste. Dobiju se slijedeće sume:

lista [1 2 [4 5 [6]] [-3 -4] 7] ima sumu 18,

lista [4 5 [6]] ima sumu 15,

lista [6] ima sumu 6, i

lista [-3 -4] ima sumu -7.

Funkcija treba vraćati kao svoj rezultat **najveću** od svih ovih suma.

Primjeri:

```
? PR SUMA [2 -3 [5] -4 [4 6 [-8 4] 17] [[-1 -6] 8] 1]
25                          Cijela lista ima najveću sumu, dakle rješenje je suma svih brojeva.

? SUMA [1 -2 1 [4] -1]
Result: 4                   Imamo dvije podliste: [1 -2 1 [4] -1] i [4], suma prve je 3, a druge 4.

? SHOW SUMA [ 1 [-2 [-3 [-4 [5] ] ] ] ]
5                           Najveća suma se dobije iz liste [5].
```

Napomena:
U drugom primjeru MSW Logo ispisuje poruku *«You don't say what to do with»* umjesto *«Result:»*.

Program snimite pod imenom **SUMA.LGO**.