1. Funkcija `sum` koristi linearnu rekurziju. Napisati funkciju tako da koristi terminalnu rekurziju.
```
def sum(f: Int => Double)(a: Int, b: Int): Double = {
  if ( a>b ) 0 else f(a) + sum(f)(a+1, b)
}
```

2. Na osnovu funkcije `sum` iz prethodnog zadatka, napisati funkciju `product` koja računa proizvod vrednosti u zadatom intervalu.

3. Napisati funkciju koja računa `n!` na osnovu funkcije iz prethodnog zadatka.

4. Generalizovati pristup.

5. Napisati funkciju koja pravi kerifikovanu verziju funkcije sa dva parametra tipa `Int` i čiji je rezultat tipa `Int`.
```
def curry(f: (Int, Int) => Int): Int => Int => Int
```

6. Napisati definiciju funkcije `id` koja vraća funkciju koja vraća vrednost jedinog parametra tipa `Int`.

7. Napisati funkciju `compose` koja za date dve funkcije `f` i `g`, koje preslikavaju `Int` u `Int`, vraća funkciju koja vrši kompoziciju funkcija `f` i `g`: `y = g(f(x))`.

8. Napisati funkciju koja vraća funkciju koja predstavlja `n` puta komponovanu funkciju `f` nad parametrom `x`: `f(f(f ... f(x) …))`.

9. Napisati funkciju iz prethodnog zadatka upotrebom funkcija `id` i `compose`.