Turing

```
module Turing where

--Jarraian dauden adibide sinple hauen bidez Haskell-en
--Turing-en makinak funtzio bezala nola defini daitezkeen
--erakusten da.
--u_t_m, u_t_m2 eta u_t_m3 funtzioei datu bezala beste funtzio
--bat pasa diezaiekegu, Turing-en makinei beste makina baten
--kodea pasatzen diezaiekegun bezala.


f:: t -> Bool
f e = True

g:: Int -> Int
g x = 0

h:: Int -> Bool
h x
        |x > 0 = True
        |x < 0 = False
        |x == 0 = h x

u_t_m:: t1 -> t2 -> Bool
u_t_m m w = m w
--m w deiak ziklatu egiten badu u_t_m funtzioak ere ziklatu egingo du
--t1 mota ipintzean, hor edozer ipini dezakegula adierazten da. Funtzio
--bat ere ipini dezakegu.
--Adibidez u_t_m h 4
--Beraz lehenengo parametroa funtzio bat izan daiteke (hau da,
--Turing-en makina bat izan daiteke eta bigarren parametroa
--makina horri edo funtzio horri eman nahi diogun datua
--u_t_m h 4 deiak h funtzioa exekutatzen du 4 datuarekin.
--Nolabait, u_t_m funtzioak ordenagailu bat bezala jokatzen du.


u_t_m2:: t1 -> t2 -> Bool
u_t_m2 m w
        |m w == True = True
        |m w == False = False
--m w deiak ziklatu egiten badu u_t_m2 funtzioak ere ziklatu egingo du

--Orokortuz
u_t_m3:: t1 -> t2 -> t3
u_t_m3 m w = m w

--
luzera:: [t] -> Int
luzera z = length z

--Orain honako deia egin dezakegu: u_t_m3 luzera [True, False, True]
--Baina beste dei hau ere egin dezakegu: u_t_m3 luzera [5, 6, 2]

--
karratua:: Int -> Int
karratua x = x * x

--Orain honako deia egin dezakegu: u_t_m3 karratua 6
```