



UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

TP Individual N°1: Funcional

Paradigmas de Programación 2023

Apellido y Nombre	Legajo	Correo electrónico
Arias Natasha	1773811	naarias@frba.utn.edu.ar



Facultad Regional Buenos Aires

Universidad Tecnológica Nacional




Av. Medrano 951

Ciudad Autónoma de Buenos Aires

<https://www.frba.utn.edu.ar/>

Enunciado

Implementar las siguientes funciones en Haskell y hacer ejemplos de consultas

 esVocal
 estaEntre
 potencia

Consultas:

```
>esVocal 'a'  
>True
```

```
>estaEntre 3 4 5  
>True
```

```
>Potencia 2 3  
>8
```

Bonus: definir los tipos de datos, analizar si podría ser más genérico.

Resolución

```
9  estaEntre :: Int -> Int -> Int -> Bool  
10  estaEntre x y z = (z < y && z > x) || (z > y && z < x)  
11
```

```
1  esVocal :: Char -> Bool  
2  esVocal 'a' = True  
3  esVocal 'e' = True  
4  esVocal 'i' = True  
5  esVocal 'o' = True  
6  esVocal 'u' = True  
7  esVocal _ = False  
8
```

```
12  potencia :: Float -> Int -> Float  
13  potencia _ 0 = 1.0  
14  potencia 0 _ = 0  
15  potencia x y  
16      | y > 0      = x * potencia x (y-1)  
17      | otherwise = 1.0 / potencia x (-y)
```

**Nota: en el caso de la función potencia tome la decisión de que en lugar de ser Float → Float → Float sea Float → Int → Float debido a que, si la variable “y” es un float, con la solución planteada entraría en una recursión infinita. De esta manera se evita ese caso, tratando de abarcar la mayor cantidad de casos posibles para los cuales funcione la potencia.

La función que podría ser más genérica es estaEntre ya que es válida sin modificar nada para Char → Char → Char → Bool

Consultas:

```
GHCI, version 9.2.5: https://www.haskell.org/ghc/  :? for help
ghci> :load tpl.hs
[1 of 1] Compiling Main                ( tpl.hs, interpreted )
Ok, one module loaded.
ghci> esVocal 'e'
True
ghci> esVocal 't'
False
ghci> esVocal 'o'
True
ghci> esVocal 'k'
False
ghci> potencia 2 6
64.0
ghci> potencia 2 (-2)
0.25
ghci> potencia 3 5
243.0
ghci> potencia 0 1000
0.0
ghci> potencia 10000 0
1.0
ghci> estaEntre 3 5 4
True
ghci> estaEntre 5 3 4
True
ghci> estaEntre 2 3 3
False
ghci> estaEntre 5 9 10
False
```