1 – história da linguagem

2 – Domínio de aplicação

3 – Paradigmas suportado

4 – Variáveis e tipos de dados

5 – Comando de controle

6 – Escopo

7 – Exemplo pratico

data Arvore a = Nulo | No (Arvore a) a (Arvore a)

deriving (Show)

criar\_arvore :: (Ord a) => [a] -> Arvore a

criar\_arvore [] = Nulo

criar\_arvore (x:xs) = criar\_arvore\_aux (No Nulo x Nulo) xs

where

criar\_arvore\_aux arvore [] = arvore

criar\_arvore\_aux arvore (x:xs) = criar\_arvore\_aux (inserir arvore x) xs

inserir :: (Ord a) => Arvore a -> a -> Arvore a

inserir Nulo x = No Nulo x Nulo

inserir (No arv1 v arv2) x

| (v == x) = No arv1 v arv2

| (v < x) = No arv1 v(inserir arv2 x)

| (v > x) = No (inserir arv1 x) v arv2

main = do

let vetor = [5,4,3,9]

(print.criar\_arvore) vetor