**Principio de Simplicidad**

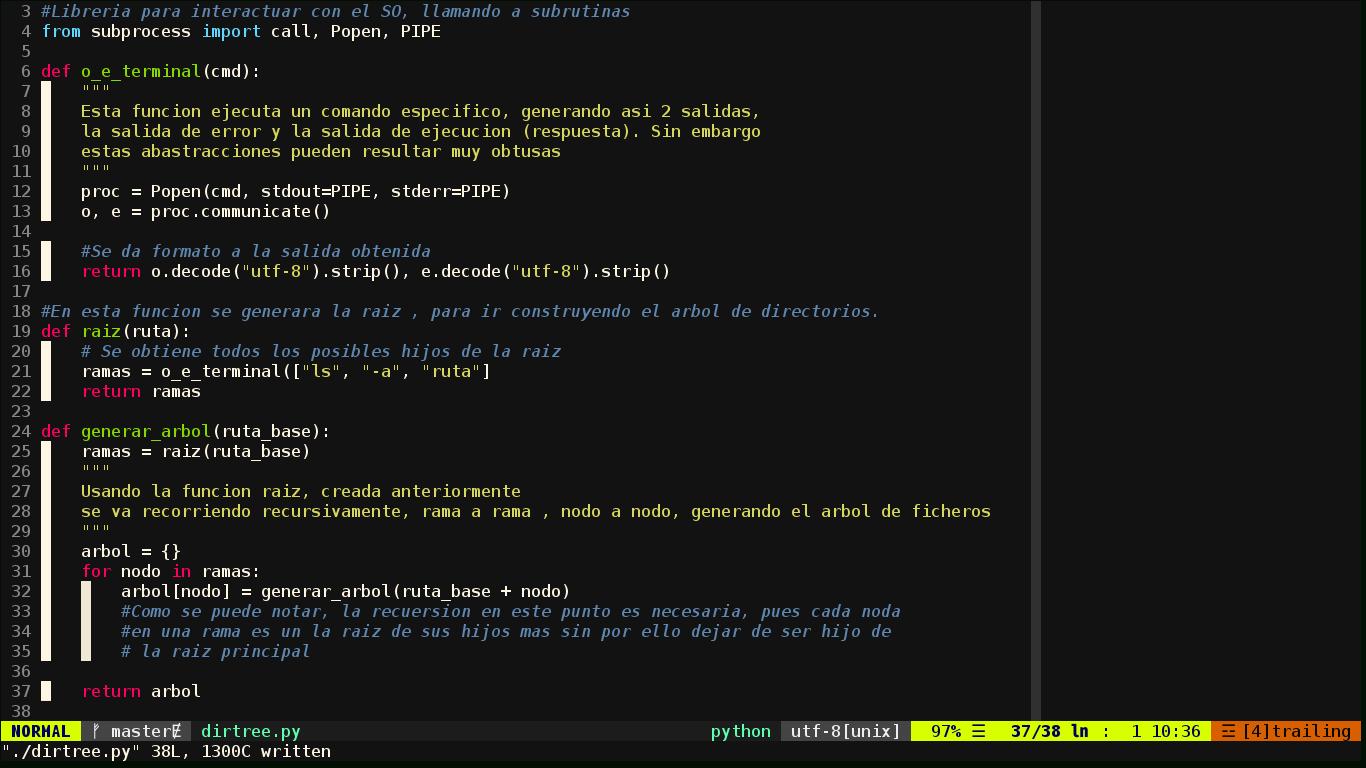
Debemos garantizar que se cumplan las funciones minimas necesarias para el redimiento sustentable

del sistema operativo, por esto es que resulta importante mantener presente los requerimiento: Un sistema operativo amigable para con los ni;os con discapacidad mental

Una funcion minima necesaria, en este contexto, es la navegacion eficiente entre ficheros, evitando

problemas posteriores como la perdida de archivos o la desorientacion del usuario.

**Estructura de Datos: Arbol**



Codigo:

**#! /usr/bin/python3 -s**

**#Libreria para interactuar con el SO, llamando a subrutinas**

**from subprocess import call, Popen, PIPE**

**def o\_e\_terminal(cmd):**

**"""**

**Esta funcion ejecuta un comando especifico, generando asi 2 salidas,**

**la salida de error y la salida de ejecucion (respuesta). Sin embargo**

**estas abastracciones pueden resultar muy obtusas**

**"""**

**proc = Popen(cmd, stdout=PIPE, stderr=PIPE)**

**o, e = proc.communicate()**

**#Se da formato a la salida obtenida**

**return o.decode("utf-8").strip(), e.decode("utf-8").strip()**

**#En esta funcion se generara la raiz , para ir construyendo el arbol de directorios.**

**def raiz(ruta):**

**# Se obtiene todos los posibles hijos de la raiz**

**ramas = o\_e\_terminal(["ls", "-a", "ruta"]**

**return ramas**

**def generar\_arbol(ruta\_base):**

**ramas = raiz(ruta\_base)**

**"""**

**Usando la funcion raiz, creada anteriormente**

**se va recorriendo recursivamente, rama a rama , nodo a nodo, generando el arbol de ficheros**

**"""**

**arbol = {}**

**for nodo in ramas:**

**arbol[nodo] = generar\_arbol(ruta\_base + nodo)**

**#Como se puede notar, la recuersion en este punto es necesaria, pues cada noda**

**#en una rama es un la raiz de sus hijos mas sin por ello dejar de ser hijo de**

**# la raiz principal**

**return arbol**