



# Informe Final

## Proyectos

Lenguajes de Programación

Ana Mora Ocaña

Ingeniería en Computación

ESPOL

Guayaquil - 7 de febrero de 2013



# Índice general

<b>1. GitHub</b>	<b>5</b>
1.1. Introducción . . . . .	6
1.1.1. Requisitos . . . . .	6
1.2. Experiencias . . . . .	7
<b>2. LaTeX</b>	<b>9</b>
2.1. Introducción . . . . .	10
2.1.1. Requisitos . . . . .	10
2.2. Experiencias . . . . .	11
<b>3. Android</b>	<b>13</b>
3.1. Introducción . . . . .	14
3.1.1. Requisitos . . . . .	14
3.2. Como ejecutar el proyecto . . . . .	15
3.3. Aplicación . . . . .	15
3.4. Experiencias . . . . .	18
<b>4. Python</b>	<b>19</b>
4.1. Introducción . . . . .	20
4.1.1. Requisitos . . . . .	20
4.2. Como ejecutar el proyecto . . . . .	21
4.3. Código . . . . .	21
4.4. Experiencias . . . . .	25
<b>5. Haskell</b>	<b>27</b>
5.1. Introducción . . . . .	28
5.2. Descripción del Algoritmo Genético . . . . .	28
5.3. El proyecto . . . . .	29
5.3.1. Requisitos . . . . .	29
5.3.2. Como Ejecutar el Proyecto . . . . .	29
5.4. Programa Fuente . . . . .	30
5.5. Experiencias . . . . .	39



## Capítulo 1

# GitHub

**github**  
SOCIAL CODING

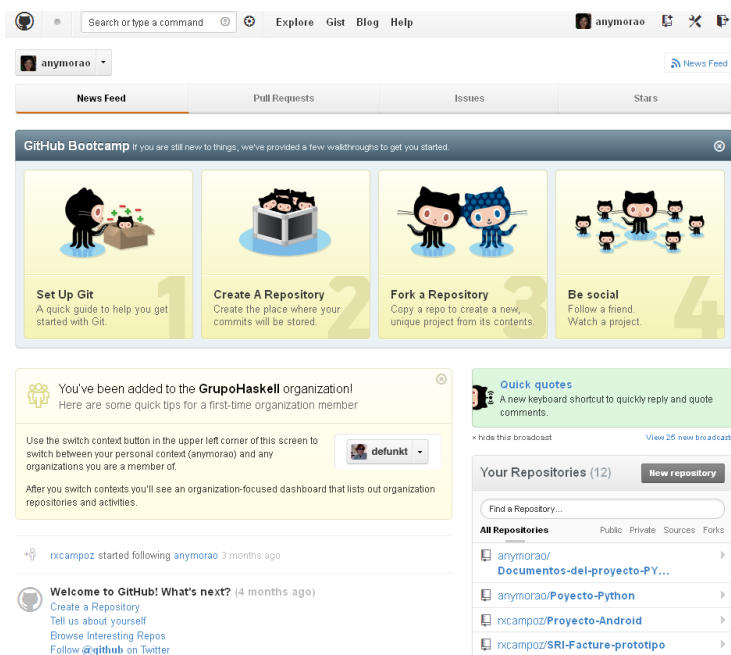


## 1.1. Introducción

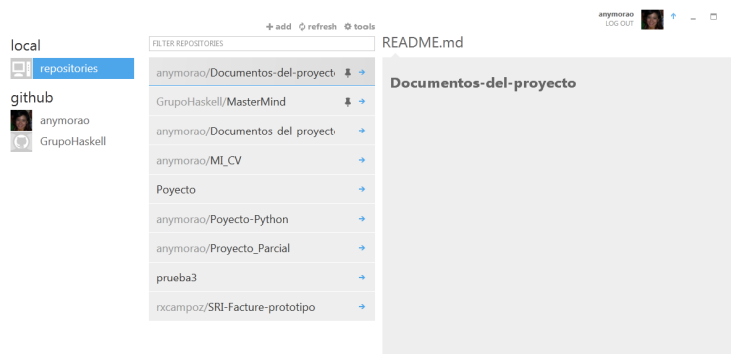
GitHub es un proyecto en linea que sirve para guardar tus proyectos utilizando el sistema de control de versiones Git. El codigo se almacena de forma pública o privada.

### 1.1.1. Requisitos

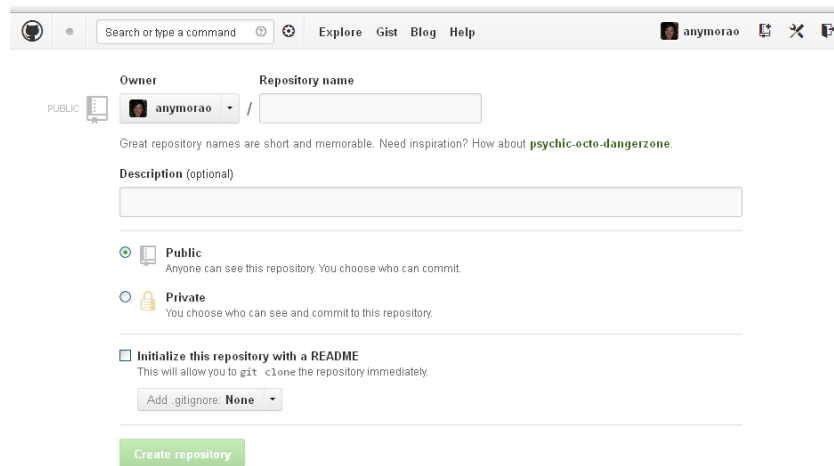
- Abrir una cuenta en la pagina de GitHub <http://www.github.com>



- Instalar GitHub para window.



- Crear el repositorio donde guardaras el proyecto.



The screenshot shows the GitHub 'Create repository' interface. At the top, there's a navigation bar with the GitHub logo, a search bar, and links for 'Explore', 'Gist', 'Blog', and 'Help'. The user 'anymerao' is logged in. The form has two main sections: 'Owner' and 'Repository name'. The 'Owner' is set to 'anymerao' and the 'Repository name' is an empty text box. Below this, there's a hint: 'Great repository names are short and memorable. Need inspiration? How about `psychic-octo-dangerzone`.' The 'Description (optional)' field is empty. There are two radio buttons for visibility: 'Public' (selected) and 'Private'. The 'Public' option is described as 'Anyone can see this repository. You choose who can commit.' The 'Private' option is described as 'You choose who can see and commit to this repository.' There's a checkbox for 'Initialize this repository with a README', which is checked. Below it, a note says 'This will allow you to `git clone` the repository immediately.' There's a dropdown for 'Add .gitignore:' with 'None' selected. At the bottom, there's a green 'Create repository' button.

## 1.2. Experiencias

- Github fue difícil de manejar para mí ya que no estaba acostumbrada a tener un repositorio donde guardar mis proyectos, intenté instalar por consola y se me hizo muy difícil así que opté por instalar el github para windows aunque me tomó un tiempo aprender como usarlo pude lograrlo.
- La falta de costumbre hace que me olvide de subir las versiones al github y cada vez que hago un cambio no lo guardo y subo el proyecto ya terminado.
- Al principio no sabía como realizar un commit esa fue una de las razones por la que no subía nada al GitHub así que tuve que pedirle ayuda a un compañero, gracias a la paciencia que tuvo para explicarme pude realizar un commit.





## Capítulo 2

### LaTeX



## 2.1. Introducción

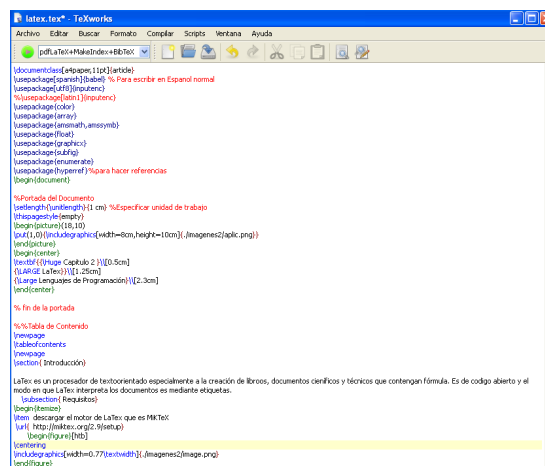
LaTeX es un procesador de texto orientado especialmente a la creación de libros, documentos científicos y técnicos que contengan fórmula. Es de código abierto y el modo en que LaTeX interpreta los documentos es mediante etiquetas.

### 2.1.1. Requisitos

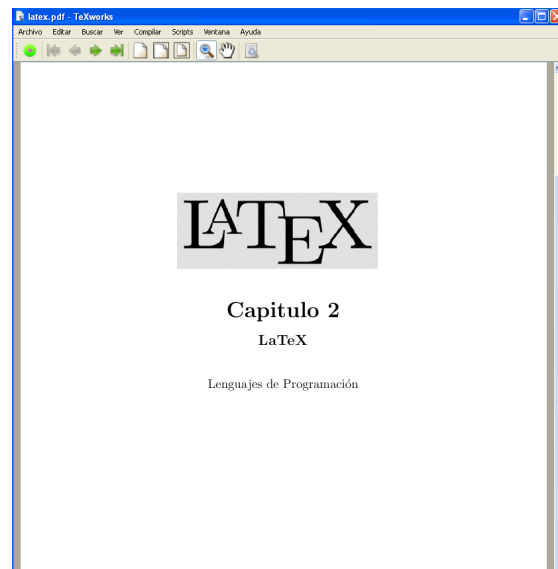
- Descargar el motor de LaTeX que es MiKTeX. <http://miktex.org>



- Abrir el área de trabajo que es el TeXworks.



- Compilar.



## 2.2. Experiencias

- Al principio me costo mucho trabajar con el código de LaTeX para documentar mis proyectos ya que estaba acostumbrada a trabajar con los procesadores de textos habituales como Word.
- Me rehusaba a usarlo porque me parecía innecesario ya que tenía a Word.
- Con la ayuda de plantillas comencé a acostumbrarme y puedo decir que ahora lo domino.
- Con cada proyecto y presentación que se hizo en el semestre poco a poco he aprendido cosas nuevas.
- Ahora me he acostumbrado a trabajar mis documentos usando LaTeX y me parece una buena herramienta para desarrollar mi tesis.



## Capítulo 3

# Android



### 3.1. Introducción

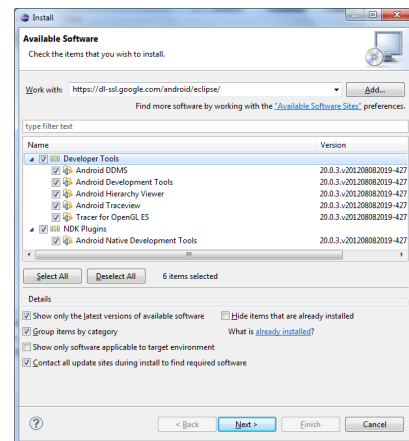
La aplicación realizada con la finalidad de ayudar al usuario a llevar un mejor control de sus facturas. Esta aplicación esta dirigida para personas naturales no obligadas a llevar contabilidad que deben cumplir que tienen problemas al momento de llevar el control de sus facturas para declaraciones de Gastos Personales en el SRI (Servicio de Rentas Internas). La aplicación recordará al usuario las fechas que le toca hacer sus declaraciones al SRI según el informe que se ha generado al llevar la contabilidad de las facturas que se ha ido guardando.

#### 3.1.1. Requisitos

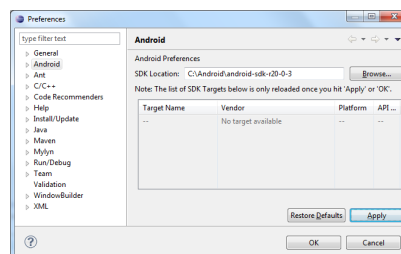
- Eclipse IDE for Java Developers
- Descargar el SDK de Android.
- Plugin Android para Eclipse.



(a)



(b)

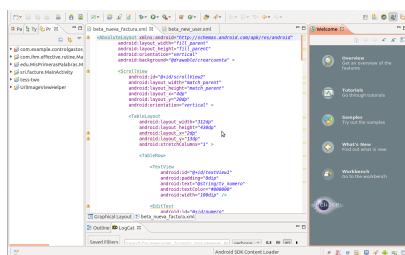


(c)

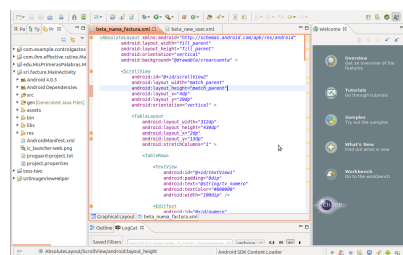
## 3.2. Como ejecutar el proyecto

Primero copiamos el archivo del proyecto en cualquier directorio, luego abrimos Eclipse , seguimos los siguientes pasos:

1. Dar click Archivo -> Abrir Proyecto y buscamos la carpeta del proyecto en el directorio que lo guardamos.
2. Luego en la barra de herramientas dar click en ejecutar.



(a)



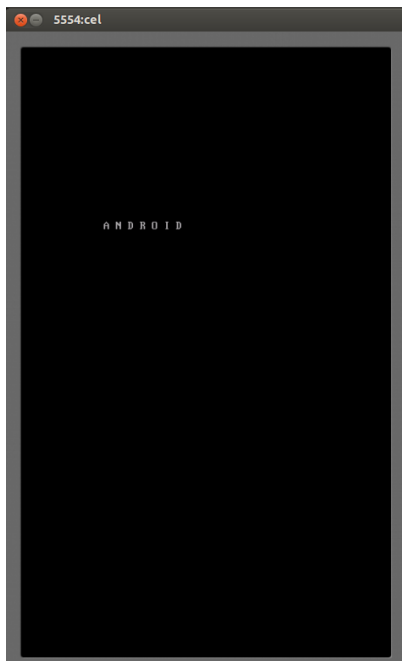
(b)

## 3.3. Aplicación

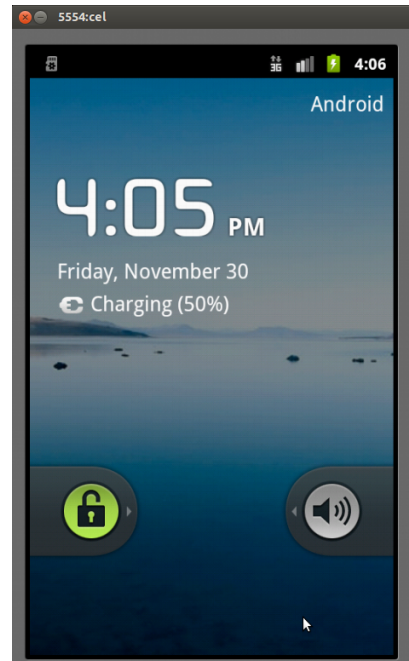
A continuación se muestran algunas imágenes de cada una de las pantallas de la aplicación

- a) El emulador cargandose
- b) Aparece el logo de la aplicación .
- c) La primera ventana donde se ingresa el usuario y la contraseña
- d) Datos de usuario

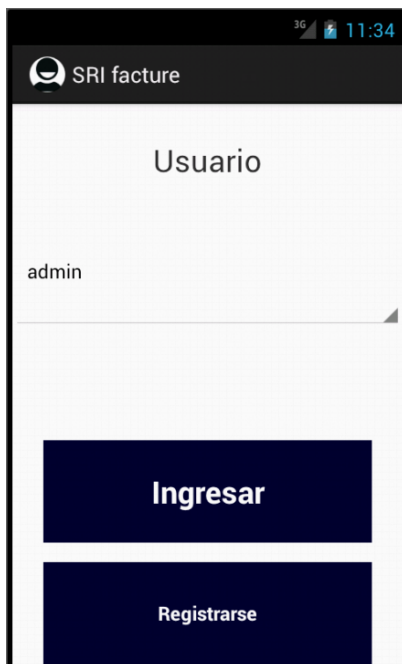
- a) El menu de la aplicación.
- b) Se puede ver la pantalla donde se ingresa a administrar factura.
- c) En esta pantalla se muestran los diferentes valores guardados
- d) Reporte .



(a)



(b)



(c)



(d)





(a)

Screenshot (b) shows the form for adding a new invoice. The fields are: Numero, Fecha (2012/12/3), Total Gasto, Total deducible, and RUC del proveedor. A 'Cambiar' button is next to the date field. A 'Detalle de deducibles' button is at the bottom.

(b)

Factura No	Fecha	
122456789	2012/12/3	12
123456789	2012/12/3	40
5663214587963	2012/12/3	45

At the bottom of the screen are two buttons: 'Añadir' and 'Eliminar'.

(c)

**REPORTE DE ALIMENTOS**

Desde 2012/12/3 hasta 2012/12/3

Factura No	Fecha	Ruc del proveedor	Deducible alimentos
123456789	2012/12/3		1.0
<b>Total:</b>			<b>2.0</b>

Generado con SRI Facture

(d)

### 3.4. Experiencias

Esta experiencia ha mostrado cómo es posible diseñar y aplicar lo que hemos aprendido en anteriores curso complementado con la investigación que se realizó.

- En el entorno que será útil la aplicación será según el lugar de encuentro del usuario en el momento de tener la factura en sus manos, porque de esta manera el usuario el usuario podrá llevar un registro de sus facturas en caso de que pierda una.
- Una observación muy importante es que nuestra aplicacion tiene uso solo para Ecuador, aunque se podria hacer cambios para que se adapte otros paises
- Mediante las pruebas realizadas, los problemas que tuvieron mayor frecuencia son:
  - Problemas para regresar en la aplicación.
  - Problemas para visualizar la opción de Crear Cuenta.
  - Problemas con la información de los reportes.

## Capítulo 4

# Python



**Una Aventura Extraña y Espacial**

## 4.1. Introducción

El proyecto realizado con la finalidad de ayudar a personas con discapacidad visual, Interactuando con el usuario por medio de sonido y de instrucciones de voz. Consiste en narrar una aventura en la cual nosotros mismos le damos el final dependiendo de las opciones que escojamos, todo esto a través de instrucciones de voz.

### 4.1.1. Requisitos

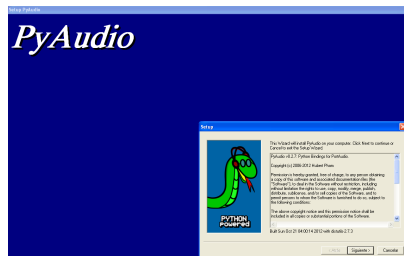
- Instalar Python 2.7.
- Instalar la librería Pygame.
- Instalar la librería Pyaudio.
- Netbeans 6.9 IDE (no es necesario).
- Instalar el plugin de Python para netbeans.



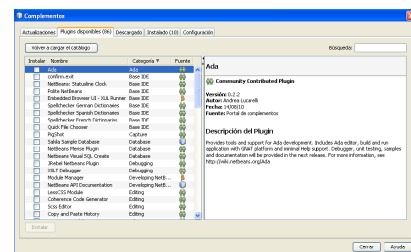
(a)



(b)



(c)

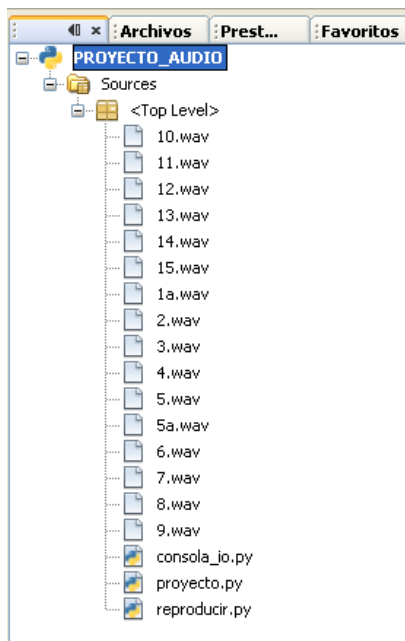


(d)

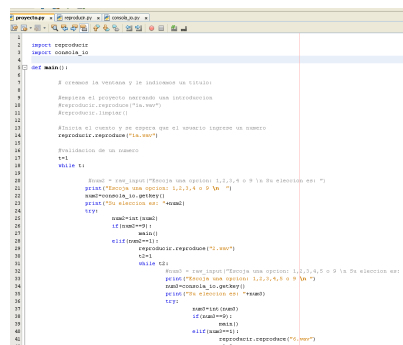
## 4.2. Como ejecutar el proyecto

Primero copiamos el archivo del proyecto en cualquier directorio, luego abrimos NetBeans , seguimos los siguientes pasos:

1. Dar click Archivo -> Abrir Proyecto y buscamos la carpeta del proyecto en el directorio que lo guardamos.
2. Luego en la barra de herramientas dar click en ejecutar.



(a)



(b)



(c)

## 4.3. Codigo

```
import reproducir
import consola_io
```

```
def main():
```

```
# creamos la ventana y le indicamos un titulo:

#empieza el proyecto narrando una introduccion
#reproducir.reproduce("1a.wav")
#reproducir.limpiar()

#Inicia el cuento y se espera que el usuario ingrese un numero
reproducir.reproduce("1a.wav")

#validacion de un numero
t=1
while t:

    #num2 = raw_input("Escoja una opcion: 1,2,3,4 o 9 \n Su eleccion es: ")
    print("Escoja una opcion: 1,2,3,4 o 9 \n ")
    num2=consola_io.getkey()
    print("Su eleccion es: "+num2)
    try:
        num2=int(num2)
        if(num2==9):
            main()
        elif(num2==1):
            reproducir.reproduce("2.wav")
            t2=1
            while t2:
                #num3 = raw_input("Escoja una opcion: 1,2,3,4,5 o 9 \n Su eleccion es: ")
                print("Escoja una opcion: 1,2,3,4,5 o 9 \n ")
                num3=consola_io.getkey()
                print("Su eleccion es: "+num3)
                try:
                    num3=int(num3)
                    if(num3==9):
                        main()
                    elif(num3==1):
                        reproducir.reproduce("6.wav")
                        t2=0
                    elif(num3==2):
                        reproducir.reproduce("7.wav")
                        t2=0
                    elif(num3==3):
                        reproducir.reproduce("8.wav")
                        t2=0
                    elif(num3==4):
```

```
reproducir.reproduce("9.wav")
t2=0
elif(num3==5):
reproducir.reproduce("10.wav")
t2=0

except ValueError:
pass
t=0

elif(num2==2):
reproducir.reproduce("3.wav")
t3=1
while t3:
#num4 = raw_input("Escoja una opcion: 1,2 o 9 \n Su eleccion es: ")
print("Escoja una opcion: 1,2 o 9 \n ")
num4=consola_io.getkey()
print("Su eleccion es: "+num4)
try:
num4=int(num4)
if(num4==9):
main()
elif(num4==1):
reproducir.reproduce("11.wav")
t3=0
t4=1
while t4:
#num5 = raw_input("Escoja una opcion: 1,2 o 9 \n Su eleccion es: ")
print("Escoja una opcion: 1,2 o 9 \n ")
num5=consola_io.getkey()
print("Su eleccion es: "+num5)
try:
num5=int(num5)
if(num5==9):
main()
elif(num5==1):
reproducir.reproduce("12.wav")
t4=0
elif(num5==2):
reproducir.reproduce("13.wav")
t4=0
t5=1
while t5:
#num6 = raw_input("Escoja una opcion: 1,2 o 9 \n Su eleccion es: ")
```

```
print("Escoja una opcion: 1,2 o 9 \n ")
num6=consola_io.getkey()
print("Su eleccion es: "+num6)
try:
    num6=int(num6)
    if(num6==9):
        main()
    elif(num6==1):
        reproducir.reproduce("14.wav")
        t5=0
    elif(num6==2):
        reproducir.reproduce("15.wav")
        t5=0
except ValueError:
    pass

except ValueError:
    pass

elif(num4==2):
    reproducir.reproduce("7.wav")
    t3=0
except ValueError:
    pass
    t=0

elif(num2==3):
    reproducir.reproduce("4.wav")
    t=0
elif(num2==4):
    reproducir.reproduce("5.wav")
    main()

except ValueError:
    pass
print "\nY asi \ncolorin colorado \neste cuento se ha acabado"

#ejemplo limpiar pantalla: reproducir:limpiar
if __name__ == "__main__":
    main()
```



## 4.4. Experiencias

- Python fue un lenguaje nuevo para mi pero debido a toda la información que hay en internet y a los ejemplos que encontré se me hizo fácil.
- Solo se usó la librería de audio para trabajar el proyecto el cual fue divertido realizarlo.
- Buscar el libro y grabar la historia fue interesante, tuve que leer muchas historias para ver cuál era la mejor para el proyecto.
- Hubo una historia a la que me gustó mucho que se llama <sup>EI</sup> "Misterio de las Piedras Sagradas", esta historia es muy interesante pero a mi compañero le pareció muy larga así que tuvimos que buscar otra.
- La historia que grabamos la encontró mi compañero de proyecto en internet, es divertida y no muy larga por eso decidimos hacer esa.



## Capítulo 5

### HasKell



## 5.1. Introducción

Mastermind (Español "Mente maestra") es un juego de mesa, de ingenio y reflexión, para dos jugadores. Se juega en un tablero con fichas blancas y negras pequeñas y de otros colores, de un tamaño algo superior. Uno de los jugadores escoge un número de fichas de colores, 4 en el juego original, y pone un código secreto oculto del otro jugador. Éste, tomando fichas de colores del mismo conjunto, aventura una posibilidad contestada con negras (fichas de color bien colocadas) o blancas (fichas de color con el color correcto, pero mal colocadas). Termina al averiguarse la combinación (es decir, se consigue una combinación con cuatro negras), o bien se agota el tablero (depende del tamaño, aunque generalmente son 15 combinaciones). Mastermind es actualmente una marca comercial propiedad de Pressman Toys; el origen puede derivar de un juego tradicional inglés denominado Toros y vacas, se jugaba sobre papel: los "toros" equivalían a las fichas negras, y las "vacas" a las blancas.

## 5.2. Descripción del Algoritmo Genético

nuestro algoritmo se baso en el paper: <http://www.cs.bris.ac.uk/Publications/Papers/2000067.pdf>

- 1) Pedir el pool y la longitud de la conjetura por teclado.
- 2) Generar la primera conjetura aleatoriamente de acuerdo a la longitud ingresada
- 3) setear el cfg
- 4) Mostrar al usuario la conjetura
- 5) Pedir al usuario que califique la conjetura
- 6) recibir el resultado de blancas y negras por parte del usuario
- 7) Determinar si el cfg debe ser cambiado por la conjetura actual de acuerdo a los siguientes parametros:
  - a) Si las blancas de la conjetura actual es mayor a las blancas del cfg entonces. Cambio cfg
  - b) Si las blancas de la conjetura actual son igual a las blancas del cfg, Si las negras de la conjetura actual son mayor a las negras del cfg entonces Cambio cfg
- 8) Segun la cantidad de negras y blancas del cfg hacer una de las siguientes opciones:

- a) Si las blancas son igual en longitud de la conjetura y las negras son igual a cero entonces ha ganado el juego
  - b) Si las blancas son igual a cero y las negras son igual a cero entonces elimina las letras del pool, añade las letras a la lista de posiciones imposibles (puede ser opcional) y genera una nueva conjetura inicial
  - c) Si las blancas son igual a cero y las negras son diferentes de cero entonces añade las letras a la lista de posiciones imposibles
  - d) Si las blancas más las negras son igual en longitud a la conjetura y las negras son diferentes de cero y las blancas son diferentes de cero entonces el pool solo se queda con dichas letras
- 9) Generar una nueva conjetura siguiendo los siguientes parámetros
- Por cada elemento que no haya estado en correcta posición (longitud del cfg - blancas)
    - Escoger una letra del cfg al azar
    - Escoger una letra del pool al azar
    - Reemplazar letra de la conjetura por la letra del pool (en la misma ubicación que estaba la letra de la conjetura)
- 10) Chequear la conjetura generada en la lista de imposibles de acuerdo al lo siguiente
- Si alguna letra de la nueva conjetura está en la lista de imposibles
  - regreso al paso 10
- 11) Repetir desde el paso 4 hasta que se acabe el juego o hasta que el usuario haya tenido 10 intentos

## 5.3. El proyecto

### 5.3.1. Requisitos

Se necesita tener instalado el compilador para el lenguaje Haskell, este se lo puede encontrar en la página oficial del lenguaje: <http://www.haskell.org>

### 5.3.2. Como Ejecutar el Proyecto

- a) Abrimos el WinGHCi
- b) Cargamos Base2.hs
- c) Cargamos Controlador2.hs



d) Cargamos Principal2.hs

e) Evaluamos

### Inicio del Mastermind

- Después de cargar todos los archivos y de evaluar el programa me pide que ingrese el pool de letras (en este caso se ingreso abcde), las cuales pueden ser maximo 10 letras una vez ingresado el pool doy enter e inmediatamente me pide que ingrese la longitud del codigo a descifrar (en este caso el codigo a desifrar tiene longitud 4).
- A continuación me genera una conjetura la cual tengo que evaluar poniendo :
  - El número de letras en posición correcta
  - El número de letras en posición incorrecta

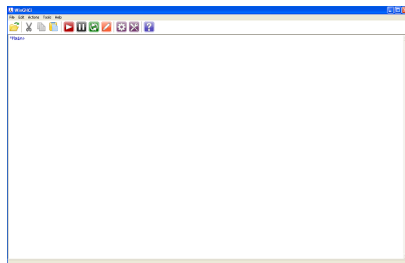
En este caso no hay ninguna letra en posición correcta pero si estan las 4 letras que pertenecen al codigo que debe adivinar.

- cuando el número de letras en la posición correcta es igual a la longitud a descifrar y el número de letras en posición incorrecta es 0 entonces sale un mensaje de felicitaciones ganaste.

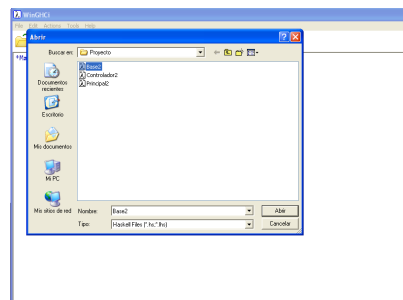
## 5.4. Programa Fuente

- BASE

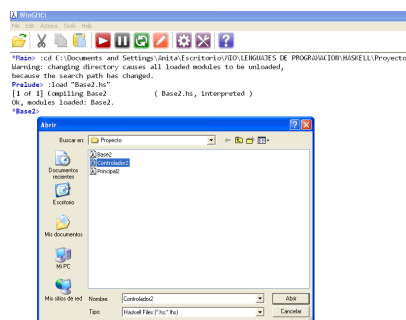
```
module Base2
( Conjetura (..)
, Imposibles (..)
, conjetura_codigo
, conjetura_aciertos
```



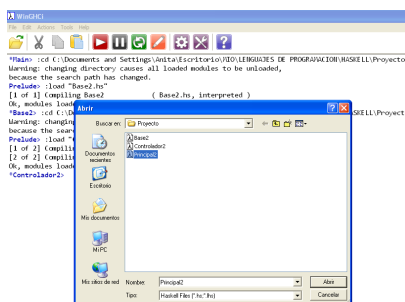
(a)



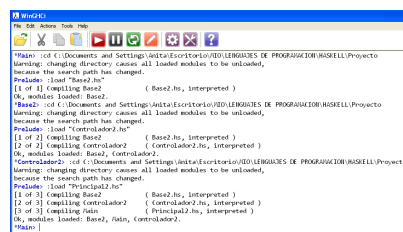
(b)



(c)



(d)



(e)

```

, conjetura_errores
, imposible_letra
, imposible_pos
) where
{-----}
data Conjetura = Conjetura String Int Int
  deriving(Show)
conjetura_codigo(Conjetura codigo _ _) = codigo
conjetura_aciertos(Conjetura _ aciertos _) = aciertos

```

```

WinGHCi
File Edit Actions Tools Help

Ingreso el pool de letras: abcde
Ingreso la longitud del codigo a descifrar: 4

```

```

WinGHCi
File Edit Actions Tools Help

Ingreso el pool de letras: abcde
Ingreso la longitud del codigo a descifrar: 4
Conjetura actual: dcba
Ingreso el numero de letras en posicion correcta: 0
Ingreso el numero de letras en posicion incorrecta: 4

```

```
conjetura_errores(Conjetura _ _ errores) = errores
```

```
data Imposibles = Imposibles Char Int
    deriving(Show)
```

```
imposible_letra(Imposibles letra _) = letra
imposible_pos(Imposibles _ posicion) = posicion
```

```
{-----}
```

#### ■ CONTROLADOR

```
import Controlador2
import Base2
```

```
{-----}
```

```
nameLambda (contador , cfg, codigo , tam , pool , imposibles , historial ,
```



```
*Main> :main
Ingrese el pool de letras: abcde
Ingrese la longitud del codigo a descifrar: 4
Conjetura actual: dcba
Ingrese el numero de letras en posicion correcta: 4
Ingrese el numero de letras en posicion incorrecta: 0
Felicitaciones , ganaste

Tu historial ha sido: dcba-----
*Main>
```

```

if (codigo == "trampa-") then
    putStr "Tramposo, has dado una mala retroalimentacion, el pool esta va
    putStr historial >>
    putStr "\n"
else
    if (contador == 10) then
        putStr "Perdiste, agotaste tus 10 intentos\n\n Tu historial ha
        putStr historial >>
        putStr "\n"
    else
        if (bln == tam) then
            putStr "Felicitaciones , ganaste\n\n Tu historial ha sido:
            putStr historial >>
            putStr "\n"
        else
            if (contador == 1 ) then
                putStr "Ingrese el pool de letras: " >>
                getLine >>=
                \first -> putStr "Ingrese la longitud del codigo a descifrar: " >>
                getLine >>=
                \last -> let pool_inicio = first
                    longitud = stringtoInt (last)
                    codigo_inicio = conjetura_inicial (pool_inicio , longit
                    cont_aux = contador + 1
                    cfg_inicial = Conjetura codigo_inicio 0 0
                    in nameLambda (cont_aux , cfg_inicial , codigo_inicio

```

```

else
    putStr "Conjetura actual: " >>
    putStr codigo >>
    putStr "\nIngrese el numero de letras en posicion correcta: "
getLine >>=
\blancas -> putStr "Ingrese el numero de letras en posicion incorrec
getLine >>=
\negras -> let historial_new = cambia_historial (codigo , historial
    aciertos = stringtoInt (blancas)
    errores = stringtoInt (negras)
    conjetura = Conjetura codigo aciertos errores
    cfg_new = cambiar_cfg (cfg , conjetura)
    imposibles_new = cambiar_imposibles (imposibles , cfg_new)
    ( pool_new , bandera ) = cambiar_pool (cfg_new , imposibles_new)
    conjetura_gen = crear_conjetura (cfg_new , conjetura , pool_new ,
    cont_new = contador + 1
in  nameLambda (cont_new , cfg_new , conjetura_gen , tam , pool_new ,

{-----

main = nameLambda ( 1 , Conjetura "ggg" 0 0 , "ggggg" , 4 , "abcdef" , [] )

```

## ■ PRINCIPAL

```

module Controlador2

( cambiar_cfg
, nuevo_imposible
, cambiar_imposibles
, elimina_letras_pool
, cambiar_pool
, valida_posicion
, ubica_letra
, crear_conjetura
, stringtoInt
, cambia_historial
, imprimir_conjetura
, imprimir_historial
, conjetura_inicial
, inttoString
) where

```

```

import Base2

{-----
cambiar_cfg (conjetura , cfg) =
    let
        aciertos_cfg = conjetura_aciertos (cfg)
        aciertos_conj = conjetura_aciertos (conjetura)
        errores_cfg = conjetura_errores (cfg)
        errores_conj = conjetura_errores (conjetura)
    in
        if ( (aciertos_conj > aciertos_cfg) || (aciertos_cfg == aciertos_conj &
            conjetura
        else
            cfg

{-----

nuevo_imposible ( [] , imposible , posicion ) = imposible {---ojo con el []--}
nuevo_imposible ( cadena , imposible , posicion ) =
    let
        letra = head (cadena)
        imp = Imposibles letra posicion
        conc = imp : imposible
        cola = tail (cadena)
        pos_new = posicion + 1
    in
        nuevo_imposible (cola , conc , pos_new)

{-----

cambiar_imposibles (imposibles , cfg) =
    let
        aciertos = conjetura_aciertos (cfg)
        errores = conjetura_errores (cfg)
        aux = conjetura_codigo (cfg)
        cont = 1
    in
        if (aciertos == 0 && errores /= 0 ) then
            nuevo_imposible (aux , imposibles , cont)
        else
            imposibles

```

```

{-----

elimina_letras_pool (codigo , [] , arreglo) = arreglo
elimina_letras_pool (codigo , pool , arreglo) =
    let
        ele = head (pool)
        col = tail (pool)
        band = ele 'elem' codigo
        conc = ele : arreglo
    in
        if (band == False ) then
            elimina_letras_pool (codigo , col , conc)
        else
            elimina_letras_pool (codigo , col , arreglo)

{-----

cambiar_pool (cfg , pool , longitud) =
    let
        aciertos = conjetura_aciertos (cfg)
        errores = conjetura_errores (cfg)
        codigo = conjetura_codigo (cfg)
    in
        if (aciertos == 0 && errores == 0) then
            ( elimina_letras_pool (codigo , pool , [] ) , False )
        else
            if (aciertos + errores == longitud) then
                ( codigo , True )
            else
                ( pool , True )

{-----

valida_posicion (letra_pool , num_cod , [] ) = True
valida_posicion (letra_pool , num_cod , imposibles) =
    let
        imp = head (imposibles)
        letra = imposible_letra (imp)
        pos = imposible_pos (imp)
    in
        if (letra == letra_pool && pos == num_cod) then
            False

```

```

        else
            valida_posicion (letra_pool , num_cod , tail (imposibles) )

{-----

ubica_letra (codigo , letra_pool , num_cod) =
    let
        ( x , y ) = splitAt num_cod codigo
        aux = letra_pool : y
    in
        init (x) ++ aux

{-----

crear_conjetura (cfg , pool , imposibles , cont , bandera) =
    let
        aciertos = conjetura_aciertos (cfg)
        errores = conjetura_errores (cfg)
        codigo = conjetura_codigo (cfg)
        long_cod = length (codigo)
        long_pool = length (pool)
        num_cod = 3 {-numero_aleatorio (1 , long_cod)-}
        num_pool = 3 {-numero_aleatorio (1 , long_pool)-}
        letra_cod = codigo !! (num_cod - 1)
        letra_pool = pool !! (num_pool - 1)
        aux = long_cod - aciertos
        letra_ok = ubica_letra (codigo , letra_pool , num_cod)
        cfg_new = Conjetura letra_ok aciertos errores
        cont_new = cont + 1
    in
        if ( null (pool) == True ) then
            "trampa-"
        else
            if ( bandera == False ) then
                conjetura_inicial (pool , long_cod , [] , 1 )
            else
                if ( valida_posicion ( letra_pool , num_cod , imposibles) == False )
                    crear_conjetura (cfg , pool , imposibles , cont , bandera)
                else
                    if (cont >= aux) then
                        letra_ok
                    else
                        crear_conjetura (cfg_new , pool , imposibles , cont_new , band

```

```

{-----

stringtoInt (cad) =
    if (cad == "0") then 0 else if (cad == "1") then 1 else if (cad == "2"

{-----

{-pedir_retroalimentacion (conjetura) =
    do
        putStrLn "Escriba el numero de letras en correcta posicion: \n"
        s <- getLine
        putStrLn "Escriba el numero de letras en incorrecta posicion: \n"
        x <- getLine
        let
            aciertos = stringtoInt (s)
            errores = stringtoInt (x)
            conj_new = Conjetura conjetura aciertos errores
        return conj_new-}

{-----

cambia_historial (conjetura , historial) = conjetura ++ "-----" ++ historial

{-----

imprimir_conjetura (conjetura) = putStrLn $ "-----Conjetura a

{-----

imprimir_historial (historial) = putStrLn $ "----- "++historial

{-----

conjetura_inicial ([] , longitud , arreglo , contador) = conjetura_inicial
conjetura_inicial (pool , longitud , arreglo , contador) =
    let
        aux = head (pool)
        conc = aux : arreglo
        cont = contador + 1
        col = tail (pool)
    in
        if (longitud == contador) then

```

```

        conc
    else
        conjetura_inicial ( col , longitud , conc , cont)

{-----

inttoString (cad) =
    if (cad == 0) then "0" else if (cad == 1) then "1" else if (cad == 2) then

```

## 5.5. Experiencias

- Haskell fue un lenguaje difícil de entender para mí debido a lo complicado de la programación funcional.
- Haskell no tiene muchas funcionalidades como las tienen los demás lenguajes de programación secuenciales.
- Tuve muchos problemas en realizar la parte del proyecto que me tocaba ya que no encontraba cómo generar de forma aleatoria la conjetura.
- También tuvimos problemas con uno de mis compañeros de grupo, que no trabajó su parte del proyecto.
- La parte de aleatoriedad en la que teníamos problemas, pudimos resolverla a último minuto pero aún tiene fallas al generar la conjetura.