

# Paradigmas de la Programación

---

## Práctica 3

### Introducción

En este informe de práctica se detallará el proceso de creación de una aplicación de consola destinada a la gestión de listas de tareas (TODO list), desarrollada con el lenguaje de programación funcional **Haskell**. El propósito central fue adquirir conocimientos y familiarizarse con los principios básicos de Haskell, su sintaxis, el manejo de operaciones de entrada y salida, la manipulación de listas y el uso de herramientas como **Stack**. Esta actividad se llevó a cabo mediante la adaptación y el análisis de tutoriales y ejemplos existentes, con el fin de construir la aplicación de manera progresiva.

### Haskell y Stack: Entorno de desarrollo

**Haskell** es un lenguaje de programación puramente funcional. En el desarrollo de este proyecto se utilizó el compilador GHC (Glasgow Haskell Compiler), y se recurrió a **Stack** como herramientas principal para la gestión del proyecto, Stack permite crear proyectos de manera sencilla, gestionar sus dependencias, compilar y ejecutar el código, todo dentro de un entorno de desarrollo controlado. Además, garantiza la consistencia del entorno a encargarse de la instalación del compilador GHC y las bibliotecas necesarias de forma aislada para cada proyecto.

```
[3 of 3] Linking .stack-work\dist\f24b2e15\build\todo-exe\todo-exe.exe
todo > copy/register
Installing library in C:\Users\Asus ROG\Documents\Paradigmas\Portafolio\Practica3\todo\stack-work\install\6e3385ff\lib\x86_64-windows-ghc-9
.8.4\todo-0.1.0.0-GcdqhfVnuS0HCwRwhJChP4
Installing executable todo-exe in C:\Users\Asus ROG\Documents\Paradigmas\Portafolio\Practica3\todo\stack-work\install\6e3385ff\bin
Registering library for todo-0.1.0.0.
todo > test (suite: todo-test)

Test suite not yet implemented

todo > Test suite todo-test passed

Asus ROG\Desktop-7AQ214R MINGW64 ~/Documents/Paradigmas/Portafolio/Practica3/todo (master)
$ stack run
someFunc

Asus ROG\Desktop-7AQ214R MINGW64 ~/Documents/Paradigmas/Portafolio/Practica3/todo (master)
$ ls
app/  CHANGELOG.md  LICENSE  package.yaml  README.md  Setup.hs  src/  stack.yaml  stack.yaml.lock  test/  todo.cabal

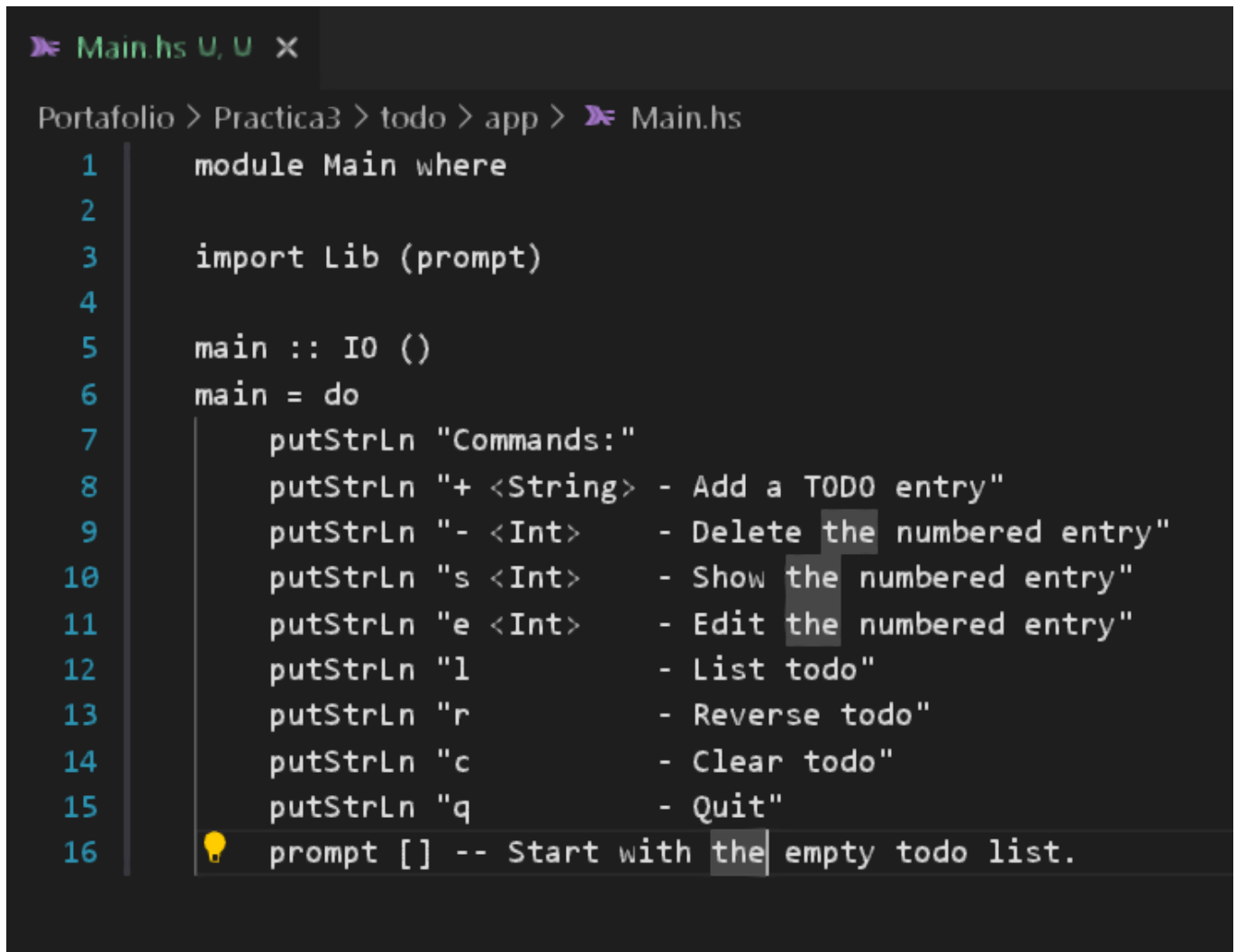
Asus ROG\Desktop-7AQ214R MINGW64 ~/Documents/Paradigmas/Portafolio/Practica3/todo (master)
```

Durante esta sesión se generó un todo, con el cual utilizamos algunos comandos para verificar que fue creado correctamente.

Al aplicar stack test se ejecuta pruebas asociadas al proyecto de Haskell.

Después utilizamos stack run para ejecutar el proyecto Haskell que ha creado mostrando el resultado en la consola.

En main.hs se configuró una interfaz con la cual se mostrarán las operaciones que se podrán realizar utilizando el proyecto.



```
Portafolio > Practica3 > todo > app > Main.hs
1  module Main where
2
3  import Lib (prompt)
4
5  main :: IO ()
6  main = do
7      putStrLn "Commands:"
8      putStrLn "+ <String> - Add a TODO entry"
9      putStrLn "- <Int>    - Delete the numbered entry"
10     putStrLn "s <Int>    - Show the numbered entry"
11     putStrLn "e <Int>    - Edit the numbered entry"
12     putStrLn "l          - List todo"
13     putStrLn "r          - Reverse todo"
14     putStrLn "c          - Clear todo"
15     putStrLn "q          - Quit"
16     prompt [] -- Start with the empty todo list.
```

Después se modificará lib.hs para que se llame la función prompt con la cual volveremos a utilizar stack run y ahora sí podremos usar las operaciones

```
[3 of 3] Linking .stack-work\dist\f24b2e15\build\todo-exe\todo-exe.exe [Objects changed]
todo> copy/register
Installing library in C:\Users\Asus ROG\Documents\Paradigmas\Portafolio\Practica3\todo\.stack-work\install\6e3385ff\lib\x86_64-windows-ghc-9.
8.4\todo-0.1.0.0-GcdqhfVnuS0HCwRwhJCHP4
Installing executable todo-exe in C:\Users\Asus ROG\Documents\Paradigmas\Portafolio\Practica3\todo\.stack-work\install\6e3385ff\bin
Registering library for todo-0.1.0.0..
Commands:
+ <String> - Add a TODO entry
- <Int>    - Delete the numbered entry
s <Int>    - Show the numbered entry
e <Int>    - Edit the numbered entry
l          - List todo
r          - Reverse todo
c          - Clear todo
q          - Quit

Test todo with Haskell. You can use +(create), -(delete), s(show), e(edit), l(list), r(reverse), c(lear), q(uit) commands.
```

```
* $ stack run
Commands:
+ <String> - Add a TODO entry
- <Int>    - Delete the numbered entry
s <Int>    - Show the numbered entry
e <Int>    - Edit the numbered entry
l          - List todo
r          - Reverse todo
c          - Clear todo
q          - Quit

Test todo with Haskell. You can use +(create), -(delete), s(show), e(edit), l
(list), r(reverse), c(lear), q(uit) commands.
l

"0 in total"

Test todo with Haskell. You can use +(create), -(delete), s(show), e(edit), l(list), r(reverse), c(lear), q(uit) commands.
+ "Jugar Pokemon"

Test todo with Haskell. You can use +(create), -(delete), s(show), e(edit), l(list), r(reverse), c(lear), q(uit) commands.
+ "Ver beisbol"

Test todo with Haskell. You can use +(create), -(delete), s(show), e(edit), l(list), r(reverse), c(lear), q(uit) commands.
+ "Comer"

Test todo with Haskell. You can use +(create), -(delete), s(show), e(edit), l(list), r(reverse), c(lear), q(uit) commands.
l

"3 in total"
0: "Comer"
```

Posteriormente se agregó información al archivo spec.hs y se utilizó stack test para verificar que la lógica de la función editIndex es correcta.

Portafolio > Practica3 > todo > test > Spec.hs

```

1  import Control.Exception
2
3  import Lib (editIndex)
4
5  main :: IO ()
6  main = do
7      putStrLn "Test:"
8      let index = 1
9      let new_todo = "two"
10     let todos = ["Write", "a", "blog", "post"]
11     let new_todos = ["Write", "two", "blog", "post"]
12
13     let result = editIndex index new_todo todos == new_todos
14
15     -- assert :: Bool -> a -> a
16     putStrLn $ assert result "editIndex worked."

```

Asus ROG@DESKTOP-7AQ214R MINGW64 ~/Documents/Paradigmas/Portafolio/Practica3/todo (master)

```

$ stack test
todo-0.1.0.0: unregistering (components added: test:todo-test)
todo> configure (lib + exe + test)
Configuring todo-0.1.0.0...
todo> build (lib + exe + test) with ghc-9.8.4
Preprocessing library for todo-0.1.0.0..
Building library for todo-0.1.0.0..
Preprocessing test suite 'todo-test' for todo-0.1.0.0..
Building test suite 'todo-test' for todo-0.1.0.0..
[1 of 2] Compiling Main [Source file changed]
[3 of 3] Linking .stack-work\dist\f24b2e15\build\todo-test\todo-test.exe [Objects changed]
Preprocessing executable 'todo-exe' for todo-0.1.0.0..
Building executable 'todo-exe' for todo-0.1.0.0..
todo> copy/register
Installing library in C:\Users\Asus ROG\Documents\Paradigmas\Portafolio\Practica3\todo\.stack-work\install\6e3385ff\lib\x86_64-windows-ghc-9.8.4\todo-0.1.0.0-GcdqhfVnuS0HCwRhJChP4
Installing executable todo-exe in C:\Users\Asus ROG\Documents\Paradigmas\Portafolio\Practica3\todo\.stack-work\install\6e3385ff\bin
Registering library for todo-0.1.0.0..
todo> test (suite: todo-test)

Test:
editIndex worked.

todo> Test suite todo-test passed
Completed 2 action(s).

```