Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación

Lenguajes de Programación Gr 40
IC-4700

Tarea 1 N-Reinas

Profesor José Castro Mora

Bryan Jiménez Chacón 2014114175

Fecha de Entrega 7 de octubre

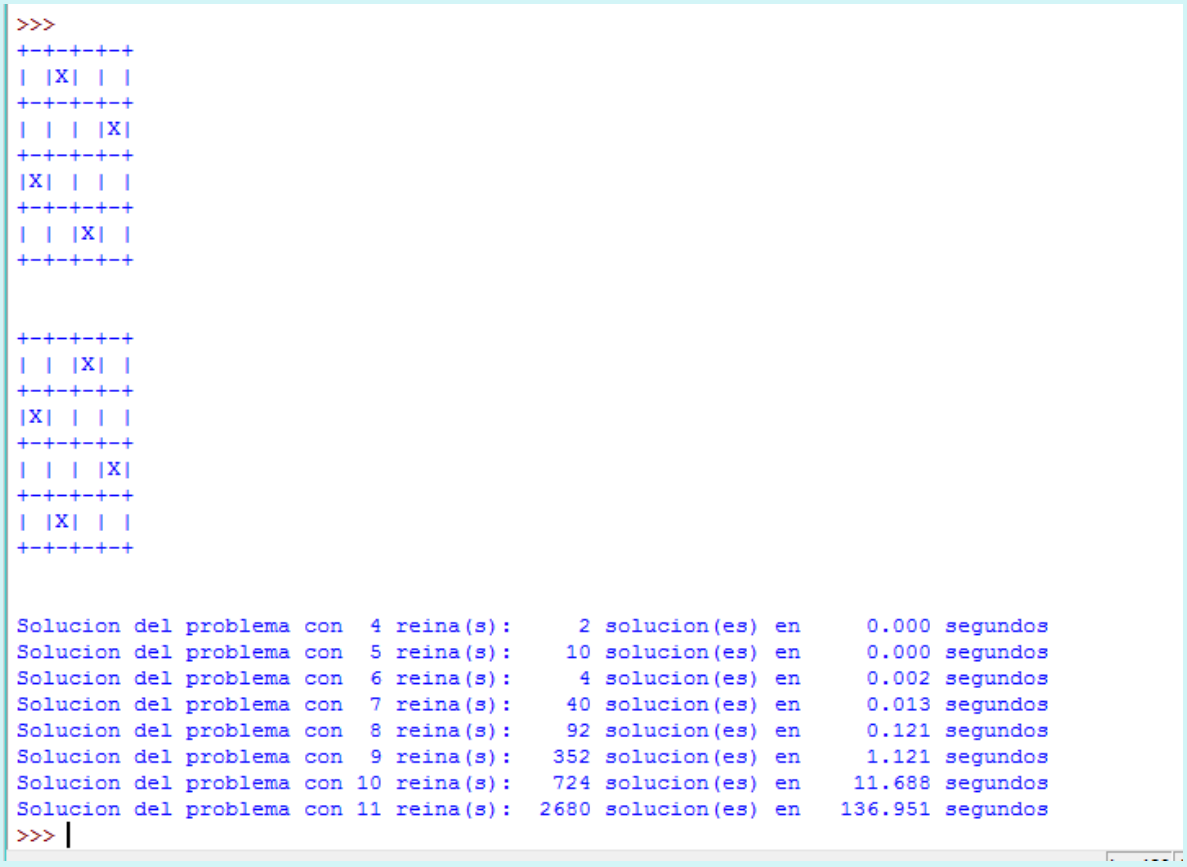
II Semestre 2016

Contenido

Resultados en cada Lenguaje

A continuación, se muestra el resultado con 4 reinas en cada lenguaje y la duración para resolver el problema desde 4 hasta 11 reinas

Python

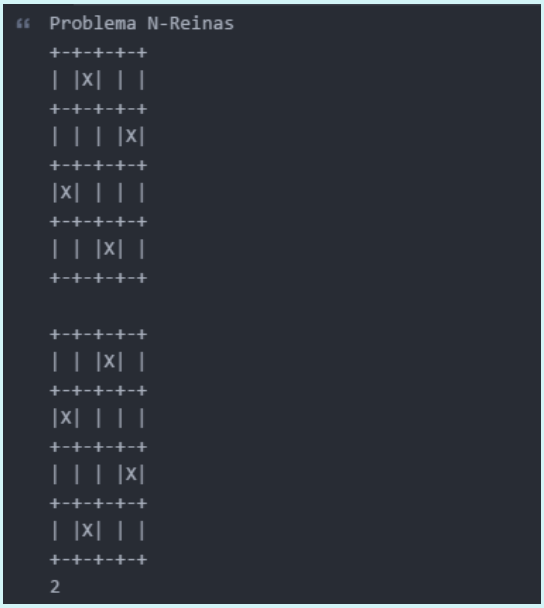


```
>>>
+-+-+-+-+
| |X| | |
+-+-+-+-+
| | | |X|
+-+-+-+-+
|X| | | |
+-+-+-+-+
| | |X| |
+-+-+-+-+


+-+-+-+-+
| | |X| |
+-+-+-+-+
|X| | | |
+-+-+-+-+
| | | |X|
+-+-+-+-+
| |X| | |
+-+-+-+-+


Solucion del problema con  4 reina(s):     2 solucion(es) en     0.000 segundos
Solucion del problema con  5 reina(s):    10 solucion(es) en     0.000 segundos
Solucion del problema con  6 reina(s):     4 solucion(es) en     0.002 segundos
Solucion del problema con  7 reina(s):    40 solucion(es) en     0.013 segundos
Solucion del problema con  8 reina(s):    92 solucion(es) en     0.121 segundos
Solucion del problema con  9 reina(s):   352 solucion(es) en     1.121 segundos
Solucion del problema con 10 reina(s):   724 solucion(es) en    11.688 segundos
Solucion del problema con 11 reina(s):  2680 solucion(es) en   136.951 segundos
>>>
```

Julia




```
Problema con 4 Reinas
Cantidad de soluciones: 2 elapsed time: 8.6784e-5 seconds

Problema con 5 Reinas
Cantidad de soluciones: 10 elapsed time: 5.2129e-5 seconds

Problema con 6 Reinas
Cantidad de soluciones: 4 elapsed time: 9.785e-5 seconds

Problema con 7 Reinas
Cantidad de soluciones: 40 elapsed time: 0.000120274 seconds

Problema con 8 Reinas
Cantidad de soluciones: 92 elapsed time: 0.000323255 seconds

Problema con 9 Reinas
Cantidad de soluciones: 352 elapsed time: 0.001296227 seconds

Problema con 10 Reinas
Cantidad de soluciones: 724 elapsed time: 0.005988096 seconds

Problema con 11 Reinas
Cantidad de soluciones: 2680 elapsed time: 0.032325872 seconds
```

Haskell

```
Prelude> :load "NReinas.hs"
[1 of 1] Compiling Main             ( NReinas.hs, interpreted )
Ok, modules loaded: Main.
*Main> :main
Incio
+-+-+-+-+
| | |X| |
+-+-+-+-+
|X| | | |
+-+-+-+-+
| | | |X|
+-+-+-+-+
| |X| | |
+-+-+-+-+

+-+-+-+-+
| |X| | |
+-+-+-+-+
| | | |X|
+-+-+-+-+
|X| | | |
+-+-+-+-+
| | |X| |
+-+-+-+-+

No hay mas soluciones
FIN
```

```
Prelude> :load "NReinas.hs"
[1 of 1] Compiling Main             ( NReinas.hs, interpreted )
Ok, modules loaded: Main.
(0.02 secs,)
*Main> length (reinas 4)
2
(0.02 secs, 115,680 bytes)
*Main> length (reinas 5)
10
(0.02 secs, 205,120 bytes)
*Main> length (reinas 6)
4
(0.00 secs, 583,864 bytes)
*Main> length (reinas 7)
40
(0.02 secs, 2,284,296 bytes)
*Main> length (reinas 8)
92
(0.03 secs, 10,314,824 bytes)
*Main> length (reinas 9)
352
(0.16 secs, 49,921,904 bytes)
*Main> length (reinas 10)
724
(0.80 secs, 251,495,208 bytes)
*Main> length (reinas 11)
2680
(4.00 secs, 1,368,630,208 bytes)
```

Prolog

```
20 ?- queen(_,_,4).
Problema con 4 reinas


+-+-+-+-+
| | |X| |
+-+-+-+-+
|X| | | |
+-+-+-+-+
| | | |X|
+-+-+-+-+
| |X| | |
+-+-+-+-+


+-+-+-+-+
| |X| | |
+-+-+-+-+
| | | |X|
+-+-+-+-+
|X| | | |
+-+-+-+-+
| | |X| |
+-+-+-+-+
Total de soluciones 2.
true.
```

```
82 ?- time(reinas(_,_,4)).
Problema con 4 reinas
Total de soluciones 2.
% 364 inferences, 0.000 CPU in 0.002 seconds (0% CPU, Infinite Lips)
true.

83 ?- time(reinas(_,_,5)).
Problema con 5 reinas
Total de soluciones 10.
% 1,517 inferences, 0.000 CPU in 0.003 seconds (0% CPU, Infinite Lips)
true.

84 ?- time(reinas(_,_,6)).
Problema con 6 reinas
Total de soluciones 4.
% 7,061 inferences, 0.000 CPU in 0.002 seconds (0% CPU, Infinite Lips)
true.

85 ?- time(reinas(_,_,7)).
Problema con 7 reinas
Total de soluciones 40.
% 32,870 inferences, 0.000 CPU in 0.006 seconds (0% CPU, Infinite Lips)
true.

86 ?- time(reinas(_,_,8)).
Problema con 8 reinas
Total de soluciones 92.
% 162,915 inferences, 0.016 CPU in 0.026 seconds (60% CPU, 10426560 Lips)
true.

87 ?- time(reinas(_,_,9)).
Problema con 9 reinas
Total de soluciones 352.
% 831,697 inferences, 0.125 CPU in 0.116 seconds (108% CPU, 6653576 Lips)
true.

88 ?- time(reinas(_,_,10)).
Problema con 10 reinas
Total de soluciones 724.
% 4,384,846 inferences, 0.625 CPU in 0.615 seconds (102% CPU, 7015754 Lips)
true.

89 ?- time(reinas(_,_,11)).
Problema con 11 reinas
Total de soluciones 2680.
% 24,775,618 inferences, 3.578 CPU in 3.581 seconds (100% CPU, 6924190 Lips)
true.
```

Scala

```
Problema con 4 Reinas

+-+-+-+-+
| | |X| |
+-+-+-+-+
|X| | | |
+-+-+-+-+
| | | |X|
+-+-+-+-+
| |X| | |
+-+-+-+-+

+-+-+-+-+
| |X| | |
+-+-+-+-+
| | | |X|
+-+-+-+-+
|X| | | |
+-+-+-+-+
| | |X| |
+-+-+-+-+
Total de soluciones 2
```

```
Problema con 4 Reinas
Total de soluciones 2
Tiempo transcurrido: 0.043885617 segundos

Problema con 5 Reinas
Total de soluciones 10
Tiempo transcurrido: 0.008621902 segundos

Problema con 6 Reinas
Total de soluciones 4
Tiempo transcurrido: 0.014782109 segundos

Problema con 7 Reinas
Total de soluciones 40
Tiempo transcurrido: 0.022698673 segundos

Problema con 8 Reinas
Total de soluciones 92
Tiempo transcurrido: 0.031713142 segundos

Problema con 9 Reinas
Total de soluciones 352
Tiempo transcurrido: 0.080855014 segundos

Problema con 10 Reinas
Total de soluciones 724
Tiempo transcurrido: 0.115088969 segundos

Problema con 11 Reinas
Total de soluciones 2680
Tiempo transcurrido: 0.671515018 segundos
```

Tabla Comparativa

Cada categoría tendrá una calificación de 1 a 5 (cantidad de lenguajes) y se sumaran los puntajes para cada lenguaje.

Categorías/Lenguajes	Python	Julia	Haskell	Prolog	Scala
Eficiencia (5 el más eficiente)	1	5	2	3	4
Claridad (5 el más claro)	5	4	1	2	3
Duración de programación (5 el más rápido)	5	3	1	2	4
Curva de aprendizaje (5 el más rápido)	5	4	2	1	3
Facilidad de Abstracción del Problema (5 el más fácil)	5	3	2	1	4
Total (Mayor puntaje = mejor)	21	19	8	9	18

Esta comparación determina que el lenguaje que resultó más sencillo de utilizar fue Python sin embargo, también demostró ser el más ineficiente, por otro lado el más eficiente fue Julia seguido por Scala, ambos obtuvieron una calificación muy cercana a Python así que representa una muy buena alternativa que no implique tanto aprendizaje extra por parte de un programador acostumbrado al estilo de programación de Python o la orientación a objetos, por otra parte como era de esperar Haskell y Prolog obtuvieron los últimos lugares, seguramente por el hecho de tener paradigmas muy diferentes a los que se está acostumbrado y esto representó un esfuerzo mucho mayor para poder resolver el mismo problema al no tener experiencia en su sintaxis ni su forma de operar, probablemente si se realizara el mismo problema nuevamente desde 0, las calificaciones de estos lenguajes mejorarían al tener un poco de experiencia en sus respectivos paradigmas.

Comentarios:

Python:

Por el tiempo que llevo programando en python y ser el primer lenguaje que aprendí me resultó bastante sencillo resolver el problema, sin necesidad de buscar documentación adicional, sin embargo, por las características de python sabía que no era la mejor opción y que probablemente se vería reflejado en los resultados. Sabía que lo iba a resolver en el menor tiempo de los 5 lenguajes, aunque sentía que tenía que haber otra opción más eficiente.

Julia:

Julia resultó ser más amigable de lo que pensaba en primer lugar, la sintaxis no se sintió como algo nuevo, así como su forma de trabajar, fue sencillo aprender a utilizarlo solo tuve que buscar algunos detalles sobre su funcionamiento para poder manejarlo correctamente. Me sorprendió su eficiencia y definitivamente lo tendré en cuenta para futuros proyectos.

Haskell:

Haskell fue un lenguaje diferente a los que conocía por lo que tuve que leer bastante al respecto antes de poder sentir que entendía lo que hacía, sin embargo, cuando comprendí bien su funcionamiento, me pareció mucho más fácil de manejar de lo que pensaba que iba a ser y por lo tanto mucho más útil, además de que facilita muchísimo utilizar la recursión que se requería en el problema de las n-reinas. Aunque tuve ciertos problemas en los primeros intentos para poder hacer trabajar bien las funciones en conjunto el resto fue relativamente sencillo y rápido para ser la primera vez programando en un lenguaje con ese paradigma.

Prolog:

Prolog fue sin lugar a dudas el lenguaje que más me costó interiorizar, debido a su naturaleza lógica, y su sintaxis tan diferente, requirió de mucha investigación extra para poder resolver el problema planteado, si siento que resulta la opción

indiscutible cuando se trata de problemas que tengan que ver con toma real de decisiones con muchas posibilidades y reglas.

Scala:

Scala fue una sorpresa, ya que su orientación a objetos y su uso de la máquina virtual de java me hacían pensar que sería prácticamente lo mismo, pero al usarlo me di cuenta que tiene diferencias importantes que lo hacen resaltar, como poder declarar todo como variable, permitiendo pasar funciones como parámetros de otra función, o no tener de declarar específicamente el tipo de dato de las variables, también note que los programas eran más pequeños en cantidad de líneas de código lo cual me gusto, por mi experiencia en Java no resultó complicado de entender, sin duda scala es un lenguaje que me llamó mucho la atención como alternativa a Java, con más versatilidad a primera vista pero sin perder facilidad y orden en el código.