INSTITUTO TECNOLÓGICO DE COSTA RICA **II SEMESTRE 2020**
INGENIERÍA EN COMPUTACIÓN
IC-4700 LENGUAJES DE PROGRAMACIÓN

DOCUMENTACIÓN DE LA TAREA PROGRAMADA PROLOG
PASO POR PUENTE

Nombre: Jose Pablo Muñoz Montero
Carnet: 2019061904

1. Estructuras de datos usadas

Describir brevemente la forma como representa la siguiente información:

- Estado: Un estado se representa mediante una lista como esta: [0,l,[a,b,c,d,e,f],[]]. El ejemplo anterior correspone al estado inicial de un problema con 6 personas (a, b, c, d, e, f). El primer elemento de la lista correspone al tiempo total transcurrido. El segundo elemento corresponde al lado en que se encuentra la antorcha: l (left) o r (right). El tercer elemento corresponde a la lista de nombres de personas que se encuentran en el lado izquierdo, mientras que el cuarto elemento correspone a la lista de nombres de personas que se encuentran al lado derecho, es decir, los que ya cruzaron.

- Movida: Una movida se representa mediante una lista como esta: [a, b]. La lista puede ser de tamaño 1 hasta el maximo de personas que permite cruzar el problema. La movida contiene los nombres de personas que seran transferidas de un lado a otro.

2. Corridas

- Presentar las consultas concretas usadas para correr depth-first en los cuatro casos. Mostrar los resultados obtenidos al resolver los casos (dar dos soluciones).

- Presentar las consultas concretas usadas para correr hill-climb en los dos primeros casos. Mostrar los resultados obtenidos al resolver los casos (dar dos soluciones).

- Presentar las consultas concretas usadas para correr best-first en los dos primeros casos. Mostrar los resultados obtenidos al resolver los casos (dar una solución).

- ***En caso de implementación parcial de la tarea se debe mostrar la ejecución de aquellos predicados del programa que funcionan correctamente.***

Depth First Search

Caso 1:

```
maxTorch(2).
maxTime(28).
crossTime(alberto,1).
crossTime(beatriz,2).
crossTime(carlos,5).
crossTime(dora,10).
crossTime(emilio,15).
```

Las soluciones se interpretan de abajo hacia arriba:

```
21 ?- solveDepthFirst.
[28,r,[],[beatriz,alberto,dora,emilio,carlos]]
[26,l,[beatriz,alberto],[dora,emilio,carlos]]
[24,r,[alberto],[dora,emilio,carlos,beatriz]]
[9,l,[alberto,dora,emilio],[carlos,beatriz]]
[8,r,[dora,emilio],[alberto,carlos,beatriz]]
[3,l,[alberto,carlos,dora,emilio],[beatriz]]
[2,r,[carlos,dora,emilio],[alberto,beatriz]]
[0,l,[alberto,beatriz,carlos,dora,emilio],[]]
Execution took 0 ms.
true ;
[28,r,[],[alberto,beatriz,dora,emilio,carlos]]
[26,l,[alberto,beatriz],[dora,emilio,carlos]]
[25,r,[beatriz],[dora,emilio,alberto,carlos]]
[10,l,[beatriz,dora,emilio],[alberto,carlos]]
[8,r,[dora,emilio],[alberto,carlos,beatriz]]
[3,l,[alberto,carlos,dora,emilio],[beatriz]]
[2,r,[carlos,dora,emilio],[alberto,beatriz]]
[0,l,[alberto,beatriz,carlos,dora,emilio],[]]
Execution took 1052 ms.
true .
```

Caso 2:

```
maxTorch(3).
maxTime(21).
crossTime(alberto,1).
crossTime(beatriz,2).
crossTime(carlos,5).
crossTime(dora,10).
crossTime(emilio,15).
```

Las soluciones se interpretan de abajo hacia arriba:

```
23 ?- solveDepthFirst.
[21,r,[],[alberto,dora,emilio,beatriz,carlos]]
[6,l,[alberto,dora,emilio],[beatriz,carlos]]
[5,r,[dora,emilio],[alberto,beatriz,carlos]]
[0,l,[alberto,beatriz,carlos,dora,emilio],[]]
Execution took 0 ms.
true ;
[21,r,[],[alberto,beatriz,carlos,dora,emilio]]
[16,l,[alberto,beatriz,carlos],[dora,emilio]]
[15,r,[beatriz,carlos],[alberto,dora,emilio]]
[0,l,[alberto,beatriz,carlos,dora,emilio],[]]
Execution took 858 ms.
true .
```

Caso 3:

```
maxTorch(2).
maxTime(42).
crossTime(alberto,1).
crossTime(beatriz,2).
crossTime(carlos,5).
crossTime(dora,10).
crossTime(emilio,15).
crossTime(julio,20).
```

Las soluciones se interpretan de abajo hacia arriba:

```
25 ?- solveDepthFirst.
[42,r,[],[alberto,beatriz,emilio,julio,carlos,dora]]
[40,l,[alberto,beatriz],[emilio,julio,carlos,dora]]
[39,r,[beatriz],[emilio,julio,alberto,carlos,dora]]
[19,l,[beatriz,emilio,julio],[alberto,carlos,dora]]
[17,r,[emilio,julio],[beatriz,alberto,carlos,dora]]
[15,l,[beatriz,alberto,emilio,julio],[carlos,dora]]
[13,r,[alberto,emilio,julio],[carlos,dora,beatriz]]
[3,l,[alberto,carlos,dora,emilio,julio],[beatriz]]
[2,r,[carlos,dora,emilio,julio],[alberto,beatriz]]
[0,l,[alberto,beatriz,carlos,dora,emilio,julio],[]]
Execution took 13 ms.
true ;
[42,r,[],[beatriz,alberto,emilio,julio,carlos,dora]]
[40,l,[beatriz,alberto],[emilio,julio,carlos,dora]]
[38,r,[alberto],[emilio,julio,beatriz,carlos,dora]]
[18,l,[alberto,emilio,julio],[beatriz,carlos,dora]]
[17,r,[emilio,julio],[beatriz,alberto,carlos,dora]]
[15,l,[beatriz,alberto,emilio,julio],[carlos,dora]]
[13,r,[alberto,emilio,julio],[carlos,dora,beatriz]]
[3,l,[alberto,carlos,dora,emilio,julio],[beatriz]]
[2,r,[carlos,dora,emilio,julio],[alberto,beatriz]]
[0,l,[alberto,beatriz,carlos,dora,emilio,julio],[]]
Execution took 824 ms.
true .
```

Caso 4:

```
maxTorch(3).
maxTime(30).
crossTime(alberto,1).
crossTime(beatriz,2).
crossTime(carlos,5).
crossTime(dora,10).
crossTime(emilio,15).
crossTime(julio,20).
```

Las soluciones se interpretan de abajo hacia arriba:

```
27 ?- solveDepthFirst.
[30,r,[],[beatriz,alberto,dora,emilio,julio,carlos]]
[28,l,[beatriz,alberto],[dora,emilio,julio,carlos]]
[26,r,[alberto],[dora,emilio,julio,beatriz,carlos]]
[6,l,[alberto,dora,emilio,julio],[beatriz,carlos]]
[5,r,[dora,emilio,julio],[alberto,beatriz,carlos]]
[0,l,[alberto,beatriz,carlos,dora,emilio,julio],[]]
Execution took 0 ms.
true ;
[30,r,[],[alberto,beatriz,dora,emilio,julio,carlos]]
[28,l,[alberto,beatriz],[dora,emilio,julio,carlos]]
[27,r,[beatriz],[dora,emilio,julio,alberto,carlos]]
[7,l,[beatriz,dora,emilio,julio],[alberto,carlos]]
[5,r,[dora,emilio,julio],[alberto,beatriz,carlos]]
[0,l,[alberto,beatriz,carlos,dora,emilio,julio],[]]
Execution took 774 ms.
true .
```

Hill Climbing

Caso 1:

```
maxTorch(2).
maxTime(28).
crossTime(alberto,1).
crossTime(beatriz,2).
crossTime(carlos,5).
crossTime(dora,10).
crossTime(emilio,15).
```

Las soluciones se interpretan de abajo hacia arriba:

```
29 ?- solveHillClimb.
[28,r,[],[alberto,carlos,beatriz,dora,emilio]]
[23,l,[alberto,carlos],[beatriz,dora,emilio]]
[22,r,[carlos],[beatriz,alberto,dora,emilio]]
[20,l,[beatriz,alberto,carlos],[dora,emilio]]
[18,r,[alberto,carlos],[dora,emilio,beatriz]]
[3,l,[alberto,carlos,dora,emilio],[beatriz]]
[2,r,[carlos,dora,emilio],[alberto,beatriz]]
[0,l,[alberto,beatriz,carlos,dora,emilio],[]]
Execution took 0 ms.
true ;
[28,r,[],[alberto,beatriz,carlos,dora,emilio]]
[26,l,[alberto,beatriz],[carlos,dora,emilio]]
[25,r,[beatriz],[alberto,carlos,dora,emilio]]
[20,l,[beatriz,alberto,carlos],[dora,emilio]]
[18,r,[alberto,carlos],[dora,emilio,beatriz]]
[3,l,[alberto,carlos,dora,emilio],[beatriz]]
[2,r,[carlos,dora,emilio],[alberto,beatriz]]
[0,l,[alberto,beatriz,carlos,dora,emilio],[]]
Execution took 693 ms.
true .
```

Caso 2:

```
maxTorch(3).
maxTime(21).
crossTime(alberto,1).
crossTime(beatriz,2).
crossTime(carlos,5).
crossTime(dora,10).
crossTime(emilio,15).
```

Las soluciones se interpretan de abajo hacia arriba:

```
31 ?- solveHillClimb.
[21,r,[],[alberto,dora,emilio,beatriz,carlos]]
[6,l,[alberto,dora,emilio],[beatriz,carlos]]
[5,r,[dora,emilio],[alberto,beatriz,carlos]]
[0,l,[alberto,beatriz,carlos,dora,emilio],[]]
Execution took 0 ms.
true ;
[21,r,[],[alberto,beatriz,carlos,dora,emilio]]
[16,l,[alberto,beatriz,carlos],[dora,emilio]]
[15,r,[beatriz,carlos],[alberto,dora,emilio]]
[0,l,[alberto,beatriz,carlos,dora,emilio],[]]
Execution took 862 ms.
true .
```

Best First Search

Caso 1:

```
maxTorch(2).
maxTime(28).
crossTime(alberto,1).
crossTime(beatriz,2).
crossTime(carlos,5).
crossTime(dora,10).
crossTime(emilio,15).
```

Las soluciones se interpretan de abajo hacia arriba:

```
33 ?- solveBestFirst.
[28,r,[],[alberto,carlos,beatriz,dora,emilio]]
[23,l,[alberto,carlos],[beatriz,dora,emilio]]
[22,r,[carlos],[beatriz,alberto,dora,emilio]]
[20,l,[beatriz,alberto,carlos],[dora,emilio]]
[18,r,[alberto,carlos],[dora,emilio,beatriz]]
[3,l,[alberto,carlos,dora,emilio],[beatriz]]
[2,r,[carlos,dora,emilio],[alberto,beatriz]]
[0,l,[alberto,beatriz,carlos,dora,emilio],[]]
Execution took 0 ms.
true .
```

Caso 2:

```
maxTorch(3).
maxTime(21).
crossTime(alberto,1).
crossTime(beatriz,2).
crossTime(carlos,5).
crossTime(dora,10).
crossTime(emilio,15).
```

Las soluciones se interpretan de abajo hacia arriba:

```
35 ?- solveBestFirst.
[21,r,[],[alberto,dora,emilio,beatriz,carlos]]
[6,l,[alberto,dora,emilio],[beatriz,carlos]]
[5,r,[dora,emilio],[alberto,beatriz,carlos]]
[0,l,[alberto,beatriz,carlos,dora,emilio],[]]
Execution took 0 ms.
true .
```

3. Estado del programa

   Indicar el estado final en que quedó el programa, problemas encontrados y limitaciones adicionales.

   Llenar la tabla siguiente

	Estado (0%-100%)	**Comentarios**
corrida usando depth-first	100	
corrida usando hill-climb	100	
corrida usando best-first	100	
initial_state	100	
final_state	100	
move	100	
update	100	
legal	100	
value	100	En las corridas adjuntas no se logra apreciar el valor de la heuristica implementada, ya que al ser pocas personas el metodo de fuerza bruta no tarda mucho. Sin embargo esto se puede evidenciar claramente si se compara el caso 3 en los 3 metodos de solucion. Cabe mencionar que los unicos tiempos de ejecucion comparables son los de la primera solucion, ya que de ahí en adelante dependen de cuando uno presione la tecla de siguiente ";".