Observaciones:

1. *En la evaluación se tendrá en cuenta la corrección, simplicidad y eficiencia de las respuestas.*
2. *Hay que describir las definiciones auxiliares (menos las del sistema).*

Los ejercicios del examen tienen como objetivo determinar una sucesión óptima de acciones de edición para corregir una palabra. Las acciones de edición son

- `copiar` que copia el carácter en la posición actual,
- `borrar` que borra el carácter en la posición actual,
- `insertar(X)` que inserta el carácter `X` en la posición actual,
- `cambiar_por(X)` que cambia el carácter en la posición actual por el carácter `X` y
- `borrar_hasta_final` que borra los caracteres desde la posición actual hasta el final de la palabra.

Ejercicio 1 (2 puntos) *Definir la relación* `transformación(+L1,+L2,-L3)` *que se verifique si* `L3` *es una lista de acciones que aplicada a la palabra representada por la lista de caracteres* `L1` *la transforma en la palabra representada por* `L2`*. Por ejemplo,*

```
?- transformación([a],[l,a],L).
L = [borrar, insertar(l), insertar(a)] ;
L = [insertar(l), copiar] ;
L = [cambiar_por(l), insertar(a)] ;
No
?- transformación([u,n,o],[u,n,a],L).
L = [copiar, copiar, borrar, insertar(a)] ;
L = [copiar, copiar, insertar(a), borrar_hasta_final] ;
L = [copiar, copiar, cambiar_por(a)] ;
No
```

Solución:
La definición de `transformación` es

```
transformación([],[],[]).
transformación([_|_],[],[borrar_hasta_final]).
transformación([],[X|L],L3) :-
   findall(insertar(Y),member(Y,[X|L]),L3).
transformación([X|L1],[X|L2],[copiar|L]) :-
   transformación(L1,L2,L).
transformación([X|L1],[Y|L2],[borrar|L]) :-
   X \= Y,
   transformación(L1,[Y|L2],L).
transformación([X|L1],[Y|L2],[insertar(Y)|L]) :-
   X \= Y,
   transformación([X|L1],L2,L).
transformación([X|L1],[Y|L2],[cambiar_por(Y)|L]) :-
   X \= Y,
   transformación(L1,L2,L).
```

Ejercicio 2 (1.5 puntos) *Se considera el siguiente programa lógico*

```
p(L,N) :-
   q(L,N,0).

q([],A,A).
q([c|L],N,A) :-
   !,
   q(L,N,A).
q([X|L],N,A) :-
   B is A+1,
   q(L,N,B).
```

Escribir el árbol de resolución y las repuestas correspondientes al programa y a la pregunta

```
?- p([c,b,c],N).
```

Solución:
El árbol de resolución se muestra en la figura 1 (página 3).

Ejercicio 3 (1.5 puntos) *El coste de una lista de acciones de edición es el número de acciones distintas de* `copiar`. *Definir la relación* `coste(+L,-N)` *que se verifique si* `N` *es el coste de la lista de acciones de edición* `L`. *Por ejemplo,*

```
?- coste([copiar,cambiar_por(s),borrar,insertar(d),copiar],N).
N = 3
```

Solución:
La definición de `coste` es

```
coste(L,N) :-
   coste_aux(L,N,0).

coste_aux([],A,A).
coste_aux([copiar|L],N,A) :-
   !, coste_aux(L,N,A).
coste_aux([_X|L],N,A) :-
   B is A+1,
   coste_aux(L,N,B).
```

Ejercicio 4 (2 puntos) *Definir la relación* `transformación_óptima(+L1,+L2,-L3)` *que se verifique si* `L3` *es una lista de acciones que aplicada a la palabra representada por la lista de caracteres* `L1` *la transforma en la palabra representada por* `L2` *y no hay listas de menor coste que* `L3` *que transforme* `L1` *en* `L2`. *Por ejemplo,*

```
?- transformación_óptima([a],[l,a],L).
L = [insertar(l), copiar] ;
No
?- transformación_óptima([u,n,o],[u,n,a],L).
L = [copiar, copiar, cambiar_por(a)] ;
No
```

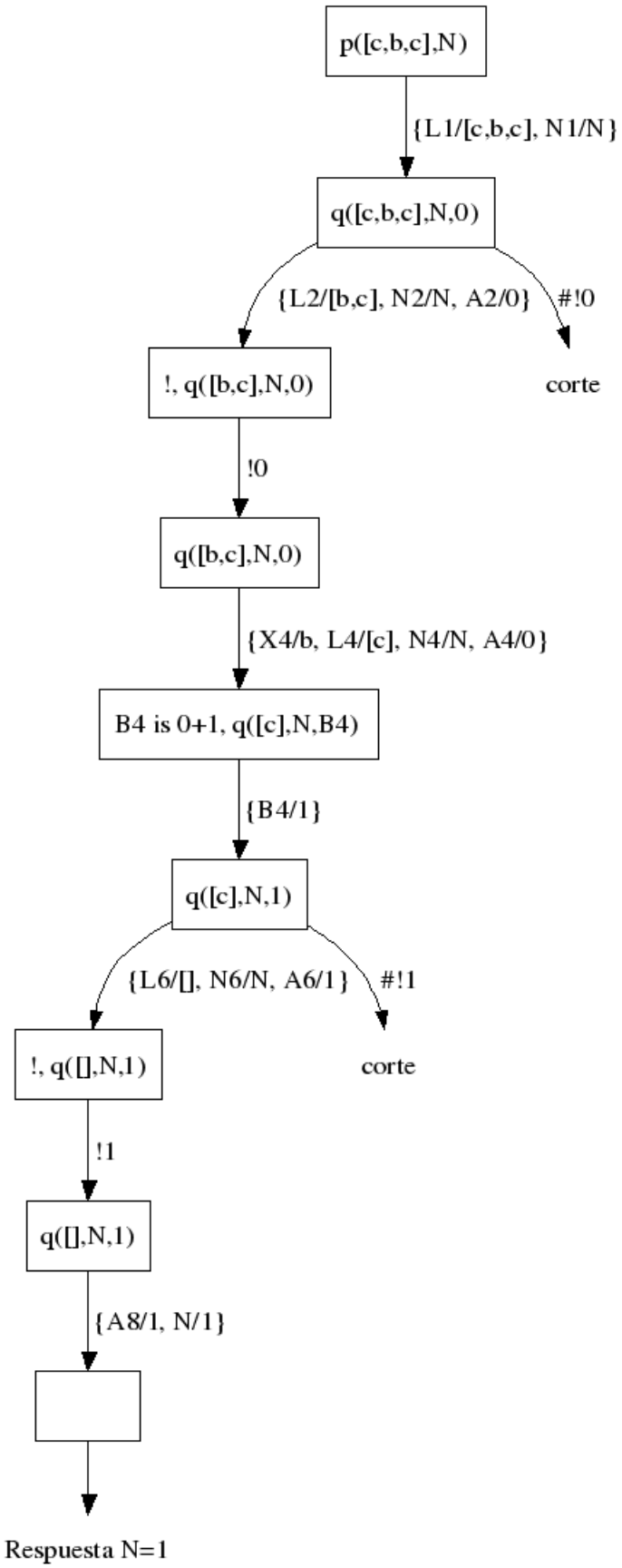

Figura 1: Árbol de resolución

```
?- transformación_óptima([o,s,o],[o,g,r,o],L).
L = [copiar, insertar(g), cambiar_por(r), copiar] ;
L = [copiar, cambiar_por(g), insertar(r), copiar] ;
No
```

Solución:
La definición de `transformación_óptima` es

```
transformación_óptima(L1,L2,L3) :-
   transformación(L1,L2,L3),
   coste(L3,N),
   not((transformación(L1,L2,L4),
        coste(L4,M),
        M < N)).
```

Ejercicio 5 (2 puntos) *Definir la relación* `palabra_a_lista(+P,-L)` *que se verifique si* `L` *es la lista de caracteres de la palabra* `P`. *Por ejemplo,*

```
?- palabra_a_lista(ogro,L).
L = [o, g, r, o]
```

Solución:
La definición de `palabra_a_lista` es

```
palabra_a_lista(A,L) :-
   name(A,L1),
   findall(C,(member(X,L1), name(C,[X])),L).
```

Ejercicio 6 (1 punto) *Definir la relación* `transformación_óptima_palabras(+P1,+P2,-L)` *que se verifique si* `L` *es una lista de acciones que aplicada a la palabra* `P1` *la transforma en la palabra* `P2` *y no hay listas de menor coste que* `L` *que transforme* `P1` *en* `P2`. *Por ejemplo,*

```
?- transformación_óptima_palabras(oso,ogro,L).
L = [copiar, insertar(g), cambiar_por(r), copiar] ;
L = [copiar, cambiar_por(g), insertar(r), copiar] ;
No
```

Solución:
La definición de `transformación_óptima_palabras` es

```
transformación_óptima_palabras(P1,P2,L) :-
   palabra_a_lista(P1,L1),
   palabra_a_lista(P2,L2),
   transformación_óptima(L1,L2,L).
```