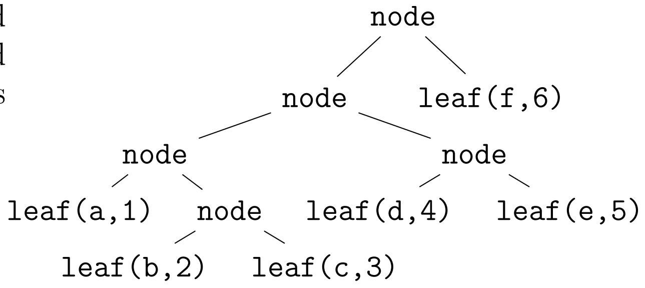
4 Prolog (ijl20)

In your answers ensure each relation has a comment giving a declarative reading of its behaviour. Avoid unnecessary use of *cut* and do not use extra-logical relations such as `findall`, `assertz` and `not` (`\+`). Built-in library relations should not be assumed. The Prolog operator in `A\=B`, meaning `A` will not unify with `B`, may be used.

(*a*) Explain Prolog's process of *unification.* What situation would an *occurs check* guard against? [3 marks]

(*b*) Assume `node(Left,Right)` and `leaf(Name,Value)` compound terms are used to represent trees such as:



Define a relation `lookup(+Tree,?Name,?Value)` which finds the value(s) associated with a given name in trees of the above form. [3 marks]

(*c*) Given a list of atoms, `L1`, define a relation `rle(+L1,?L2)` which run-length *encodes* `L1` into `L2`. For example, `rle([a,a,b,c,a,a,a],L)` should succeed with `L=[2*a, 1*b, 1*c, 3*a]`. Giving reasons, indicate for your answer whether a query `rle(L,[2*a, 1*b, 1*c, 3*a])` would succeed with `L=[a,a,b,c,a,a,a]`. [5 marks]

(*d*) Complementary to `rle/2`, define a relation `rld(+L1,?L2)` which *decodes* a run-length-encoded list `L1` as defined in part (c) into `L2`. [4 marks]

(*e*) The Prolog relations below, given a query `alter_list([2,4,6],L)`, will succeed with `L=[a,a,b]`. Use an additional difference-list argument to accumulate the execution path through the Prolog clauses. Number the clauses 1 to 4 such that `alter_list([2,4,6],L,Path-[])` will succeed with the sequence of clauses as a list of integers in Path, i.e. with `L=[a,a,b]` and `Path=[4,1,4,1,4,2,3]`.

```
change(N,a) :- N < 5.
change(N,b) :- N >= 5.

alter_list([],[]).
alter_list([H1|T1],[H2|T2]) :- change(H1,H2), alter_list(T1,T2).
```

[5 marks]