CS3011 Symbolic Programming
Second of Two Assessed Labs[1]

1. Write a DCG that accepts strings of the form $u2v$ where u and v are strings over the alphabet $\{0, 1\}$ such that the number of 0's in u is twice the number of 1's in v. For example,

```
| ?- s([0,1,0,1,2,0,0,1,0],L).
L = [];
L = [0];
no
```

2. Exercise 6.6 in Learn Prolog Now describes a street with

   (∗) three neighbouring houses that all have a different colour, namely red, blue, and green. People of different nationalities live in the different houses and they all have a different pet.

   Leaving out all the other constraints mentioned in that exercise, write a DCG that outputs strings

   `[Col1,Nat1,Pet1, Col2,Nat2,Pet2, Col3,Nat3,Pet3]`

   satisyfing (∗), where the nationalities are `english, spanish, japanese` and the pets are `jaguar, snail, zebra`. For example,

```
| ?- s([red,english,snail, blue,japanese,jaguar, green,spanish,Z],
       []).
Z = zebra;
no
```

3. Write a DCG that given a non-negative integer `Sum`, accepts lists of integers ≥ 1 that add up to `Sum`. For example,

```
| ?- s(3,L,[]).
L = [3];
L = [2,1];
L = [1,2];
L = [1,1,1];
no
```

   It may be useful to write a predicate `mkList(+Num,?List)` that returns a list `List` of integers from `Num` down to 1. For example,

```
| ?- mkList(3,L).
L = [3,2,1];
no
```

Be sure you understand how the DCG clauses translate to ordinary Prolog clauses with difference lists.

[1] Due Nov 20 (Tuesday): submit to Blackboard and, if possible, demonstrate during lab (Mon 4-5, Tues 2-3). For any extensions beyond Nov 20, email your demonstrator, David Woods (dwoods@tcd.ie).