construire des listes de N termes

- définir le prédicat `mklist(T,N,L)` qui est vrai si `L` est une liste de `N` termes `T`

exécution

```
?- mklist(a,5,L).
L = [a, a, a, a, a] .

?- mklist([a,b],3,L).
L = [[a, b], [a, b], [a, b]] .
```

dupliquer les termes d'une liste

- définir le prédicat `duplicate(L1,N,L2)` qui est vrai si `L2` est une liste contenant `N` fois les termes de `L1`
- utiliser `mklist/3`

exécution

```
?- duplicate([1,2,3],2,X).
X = [1, 1, 2, 2, 3, 3] .

?- duplicate([1,2,3],3,X).
X = [1, 1, 1, 2, 2, 2, 3, 3, 3] .
```

rotation à gauche d'une liste

- définir le prédicat `rotate_left(L1,N,L2)` qui est vrai si `L2` correspond à `N` rotations vers la gauche de `L1`

exécution

```
?- rotate_left([1,2,3,4],1,X).
X = [2, 3, 4, 1] .

?- rotate_left([1,2,3,4],2,X).
X = [3, 4, 1, 2] .

?- rotate_left([1,2,3,4],5,X).
X = [2, 3, 4, 1] .
```

rotation à droite d'une liste

- définir le prédicat `rotate_right(L1,N,L2)` qui est vrai si `L2` correspond à `N` rotations vers la droite de `L1`
  - en utilisant un prédicat `list_and_last(L1,L2,A)` qui est vrai si `L1` est égale à `L2` en ajoutant `A` à la fin de `L2` (i.e. pour `L1 = {1,2,3}`, on a `L2 = {1,2}` et `A = 3`)
  - en utilisant `last/2` et `reverse/2`

exécution

```
?- rotate_right([1,2,3,4],1,X).
X = [4, 1, 2, 3] .

?- rotate_right([1,2,3,4],2,X).
X = [3, 4, 1, 2] .

?- rotate_right([1,2,3,4],5,X).
X = [4, 1, 2, 3] .
```

être un palindrome

- définir le prédicat `palin(L)` qui est vrai si `L` est un palindrome
  - en utilisant `length/2`, `list_and_last/3`
  - en utilisant `length/2`, `last/2` et `reverse/2`
  - en utilisant `last/2` et `reverse/2` (i.e. sans `length/2`)

exécution

```
?- palin([1]).
true .

?- palin([1,1]).
true .

?- palin([1,2,1]).
true .

?- palin([1,2,3,2,1]).
true .
```

un élément aléatoire d'une liste

- définir le prédicat `random_elt(L,R)` qui est vrai si `R` est un élément aléatoire de `L`
  - en utilisant `length/2`, `random/3` et `nth0/3`

exécution

```
?- random_elt([],X).
false.

?- random_elt([1,2,3,4,5],X).
X = 1.

?- random_elt([1,2,3,4,5],X).
X = 4.
```

couple aléatoire de 2 éléments d'une liste

- définir le prédicat `random_couple(L,R)` qui est vrai si `R` est un couple aléaoire de 2 éléments de `L`
  - en utilisant `random_elt/2` et `select/3`
  - en utilisant `random_permutation/2`

exécution

```
?- random_couple([1],X).
false.

?- random_couple([1,2],X).
X = [2, 1] .

?- random_couple([1,2,3,4],X).
X = [4, 3] .
```