This is an introduction to MAPLE which is a program for doing mathematics with computers
(a computer algebra system). We will focus on permutations, groups, and discrete mathematics

We work all the time with permutations S_n for some n.
A PERMUTATION inside MAPLE can be represented in two ways:

1) As permutation list (1 ->3, 2->1,3 ->2) which is encoded by a list [3, 1, 2].

2) As the product of disjoint cycles: (1,2) (3,2,4) is encoded in a list of lists [[1,2],[3,2,4]].

We can construct many many groups using permutations. In fact we can construct ALL
finite groups this way! Here is an example:

The cheap_rubik box is a toy like the famous Rubik's cube.
Instead of a 3x3 cube is a 3x2 box.
The point is that once we label the faces of cheap_rubik box we can express the mechanical
moves by a series of permutations of 18 symbols of letters!! See picture for a description of the
toy

The subgroup of S_18 generated by these permutations is precisely the group of operations
for the cheap_rubik box. Now here is a practical problem. I give you a configuration that
could in principle be a valid cheap_rubik configuration. How can we decided whether there
are moves that really achieve it? Take for example [10, 6, 9, 11, 5, 8, 12, 4, 7, 3, 2, 1, 13, 14, 15, 16, 17, 18]:

Let’s begin our computer exploration:

Since we are beginners we can just start by asking for help:

```
> help(group);
```

You can use this as a reference for all kinds of commands inside MAPLE. Now we call the library
of commands that deals with groups and permutations.

```
> with(group);
```

Once we have all the commands available we can define inside maple the group of moves: These are
certain ‘‘basic’’ permutations from which any other configuration

can be obtained. They are the generators of the group!!

```
> moves_rubik:={a=[[1,3,9,7],[2,6,8,4]],
  b=[[10,12,18,16],[11,15,17,13]],
> c=[[1,12],[3,10],[2,11]],d=[[3,18],[9,12],[
  6,15]],
  e=[[9,16],[7,18],[8,17]],f=[[7,10],[1,16],[
  4,13]]};
> cheaprubik:=permgroup(18,moves_rubik);
```

Then we can ask questions about the group: How many elements? Is the group Abelian?, etc:

```
> grouporder(cheaprubik);
```

Another possibility is to ask whether an specific permutation belongs to the group CHEAPRUBIK.

```
> groupmember([[1,10,3,9,7,12],[2,6,8,4,11]],
  cheaprubik);
```

Here is the same question for a different permutation!

```
> groupmember([[1,10,11]],cheaprubik);
```

Here is a way of multiplying to permutation as long as they are in cycle notation

```
> mulperms([[2,3,4], [1,6]], [[4,6]]);
```

Of course, it then becomes necessary to transfer between the cycle notation and the list notation. Here is how to do it:

```
> convert([[2, 8, 13, 11, 4, 6, 15, 17], [3,
  16], [7, 9], [10, 18]], 'permlist', 18);
> convert([10, 6, 9, 11, 5, 8, 12, 4, 7, 3,
  2, 1, 13, 14, 15, 16, 17, 18],'disjcyc');
```

Now suppose you wish to decide what are the cosets of a certain subgroup (like in the proof of Lagrange theorem!). First you input the

group and the candidate subgroup:

```
> pg1:=permgroup(4,{[[1,2,3]],[[1,3,2]],[[1,3
  ,4]],[[1,4,3]],[[1,2,4]],[[1,4,2]],[[2,3,4]
  ],[[2,4,3]]});
> grouporder(pg1);
pg2:=permgroup(4,{[[1,2],[3,4]]});
```

But we wish first to be sure pg2 is a subgroup! Else the calculation makes no sense.

```
> issubgroup(pg2,pg1);
```

Finally we ask for the list of cosets to be computed. Did we expect that answer?

```
> cosets(pg1,pg2);
>
```