How to sort a hash in Perl?

hash keys values sort cmp <=> $a $b

Prev Next

This question comes up often, and every time it might hide something interesting behind it.

One of the important features of a hash, or hashmap, or dictionary, or associative array as some other languages like to call it, is that it is a set of **unsorted** key-value pairs.

So when someone asks **how to sort a hash?**, the reaction usually is that you can't.

But what do people mean when they want to **sort a hash?**

For example let's say we have a hash with the names of the planets in our Solar system and with their average distance from the Sun as measured in Astronomical units.

Reading those pages actually revealed that Pluto is not considered a planet on its own any more, but as a dwarf planet which is part of the Kuiper belt. There is also Charon which is sometimes considered to be a moon of Pluto, sometimes as part of a binary system. It will be useful for our purposes, so I added it to the hash.

Anyway, we create the hash, fill it with values and the print it out using a `foreach` loop on the planet names returned by the `keys` function.

```perl
use strict;
use warnings;
use 5.010;

my %planets = (
    Mercury => 0.4,
    Venus   => 0.7,
    Earth   => 1,
    Mars    => 1.5,
    Ceres   => 2.77,
    Jupiter => 5.2,
    Saturn  => 9.5,
    Uranus  => 19.6,
    Neptune => 30,
    Pluto   => 39,
    Charon  => 39,
);

foreach my $name (keys %planets) {
    printf "%-8s %s\n", $name, $planets{$name};
}
```

The output looks like this:

```
Jupiter  5.2
Uranus   19.6
Ceres    2.77
Saturn   9.5
Earth    1
Neptune  30
Charon   39
Mars     1.5
Venus    0.7
Mercury  0.4
Pluto    39
```

Not only is it in a seemingly random order, depending on our version of Perl it can be in different order as well. Even if in older versions of Perl this order seemed to be stable between runs, starting from 5.18.0, even that is very unlikely.

Sort the hash in alphabetical order of its keys

When someone wants to **sort a hash**, one possibility is that he wants to sort the planets in alphabetical order. That's quite easy.

```
foreach my $name (sort keys %planets) {
    printf "%-8s %s\n", $name, $planets{$name};
}
```

The output is always going to be:

```
Ceres    2.77
Charon   39
Earth    1
Jupiter  5.2
Mars     1.5
Mercury  0.4
Neptune  30
```

```
Pluto     39
Saturn    9.5
Uranus    19.6
Venus     0.7
```

But that's not exactly alphabetical sorting. The default behavior of `sort` is to sort based on the ASCII table. (Except when `use locale` is in effect, but we don't want to go there now.) This means that the default sorting would put all the upper-case letters in front of all the lower-case letters. If we really want to have alphabetical sorting we can do the following:

```perl
foreach my $name (sort {lc $a cmp lc $b} keys %planets) {
    printf "%-8s %s\n", $name, $planets{$name};
}
```

That's OK, but what if what we want is to **sort the values of the hash**?

Sort the values of the hash

That's another thing that can be easily misunderstood. If we take that request literally we will write the following code:

```perl
foreach my $distance (sort values %planets) {
    say $distance;
}
```

gaining the following output:

```
0.4
0.7
1
1.5
19.6
2.77
30
39
39
5.2
9.5
```

That is, we fetch the `values` of the hash, and sort them based on the ASCII table.

Maybe we are kind and notice the values are numbers and 2.77 should not fall between 19.6 and 30, and sort them according to their numerical value like this:

```perl
foreach my $distance (sort {$a <=> $b} values %planets) {
    say $distance
```