Subject: Re: Sort a HASH
Posted by Gray on Wed, 22 Jun 2011 15:30:03 GMT
View Forum Message <> Reply to Message

On Jun 22, 8:06 am, JDS <jdtsmith.nos...@yahoo.com> wrote:
> How can you sort a HASH() in IDL?
>
> IDL> h=hash(['a','c','d','b'],[7,1,5,8])
> IDL> print,h
> c: 1
> a: 7
> b: 8
> d: 5
>
> OK, they are randomly sorted. That's cool, it's a hash. I'll just sort on the hash key...
>
> IDL> s=sort(h->keys())
> IDL> print,s
> 0
>
> Hmmm, it seems this isn't sorting correctly...
>
> IDL> help, h->keys()
> <Expression> LIST <ID=3175 NELEMENTS=4>
>
> Ahah, the return of Keys() is a list! (Why does SORT pretend it can sort a LIST?). We need an array. Continuing onwards...
>
> IDL> s=sort((h->keys())->toarray())
> IDL> print,(h.keys())[s]
> a
> b
> c
> d
>
> Now we're getting there. Let's just index...
>
> IDL> print,h[(h.keys())[s]]
> c: 1
> a: 7
> b: 8
> d: 5
>
> Uhhh... what? Maybe it has something to do with indexing via a LIST. Let's make it an array first:
>
> IDL> print,((h.keys()).toArray())[s]
> a b c d

>
> Should be perfect...
>
> IDL> print,h[((h.keys()).toArray())[s]]
> c: 1
> a: 7
> b: 8
> d: 5
>
> But it's not!
>
> Is there any way to index a hash and have the order of the keys be maintained, for example for this sorting problem? Loss of order is a well known feature of hash storage, but when you explicitly specify the order, it should be respected. Am I missing something?

http://groups.google.com/group/comp.lang.idl-pvwave/browse_thread/thread/fa55eae4c4be5880/012750823bc9dce2

Subject: Re: Sort a HASH
Posted by Gray on Wed, 22 Jun 2011 15:33:13 GMT
View Forum Message <> Reply to Message

On Jun 22, 8:06 am, JDS <jdtsmith.nos...@yahoo.com> wrote:
> How can you sort a HASH() in IDL?
>
> IDL> h=hash(['a','c','d','b'],[7,1,5,8])
> IDL> print,h
> c: 1
> a: 7
> b: 8
> d: 5
>
> OK, they are randomly sorted. That's cool, it's a hash. I'll just sort on the hash key...
>
> IDL> s=sort(h->keys())
> IDL> print,s
>   0
>
> Hmmm, it seems this isn't sorting correctly...
>
> IDL> help, h->keys()
> <Expression>   LIST <ID=3175 NELEMENTS=4>
>
> Ahah, the return of Keys() is a list! (Why does SORT pretend it can sort a LIST?). We need an array. Continuing onwards...
>
> IDL> s=sort((h->keys())->toarray())

> IDL> print,(h.keys())[s]
> a
> b
> c
> d
>
> Now we're getting there. Let's just index...
>
> IDL> print,h[(h.keys())[s]]
> c: 1
> a: 7
> b: 8
> d: 5
>
> Uhhh... what? Maybe it has something to do with indexing via a LIST. Let's make it an array first:
>
> IDL> print,((h.keys()).toArray())[s]
> a b c d
>
> Should be perfect...
>
> IDL> print,h[((h.keys()).toArray())[s]]
> c: 1
> a: 7
> b: 8
> d: 5
>
> But it's not!
>
> Is there any way to index a hash and have the order of the keys be maintained, for example for this sorting problem? Loss of order is a well known feature of hash storage, but when you explicitly specify the order, it should be respected. Am I missing something?

I don't see my first post of this... so apologies if it appears twice. I had the same question a little while ago:

http://groups.google.com/group/comp.lang.idl-pvwave/browse_thread/thread/fa55eae4c4be5880/012750823bc9dce2

Subject: Aw: Re: Sort a HASH
Posted by JDS on Fri, 24 Jun 2011 08:37:04 GMT
View Forum Message <> Reply to Message

You appear to be asking why a HASH is not ordered. Having been familiar with HASH's from other languages, I did not expect them to be ordered. What is peculiar here, is that, without

converting hashes into a more primitive data type, there is no way to order them yourselves (or rearrange them in any way, other than a loop)! That is in stark contrast to other languages. For example, a common idiom in Perl is:

foreach key (sort keys %hash) {
 % do something
}

Is there really no equivalent in IDL?

Subject: Re: Sort a HASH
Posted by Gray on Fri, 24 Jun 2011 10:52:25 GMT
View Forum Message <> Reply to Message

On Jun 24, 4:37 am, JDS <jdtsmith.nos...@yahoo.com> wrote:
> You appear to be asking why a HASH is not ordered. Having been familiar with HASH's from other languages, I did not expect them to be ordered. What is peculiar here, is that, without converting hashes into a more primitive data type, there is no way to order them yourselves (or rearrange them in any way, other than a loop)! That is in stark contrast to other languages. For example, a common idiom in Perl is:
>
> foreach key (sort keys %hash) {
> % do something
>
> }
>
> Is there really no equivalent in IDL?

Well, I was asking why when I indexed a hash were the elements not returned in the order I indexed them.

Subject: Re: Sort a HASH
Posted by chris_torrence@NOSPAM on Fri, 24 Jun 2011 22:28:04 GMT
View Forum Message <> Reply to Message

Hi all,

The reason this isn't working is because when you index a hash, you get back another hash. It doesn't matter the order of the keys when you do the indexing, IDL will create a new hash with the desired key/value pairs, and they will be in whatever order the hash determines.

A way around this is to just get both the keys & values out, then sort on those:

```
h=hash(['a','c','d','b'],[7,1,5,8])
keys = (h.KEYS()).toArray()
val = (h.VALUES()).toArray()
s = SORT(keys)
print, keys[s], val[s]

a b c d
      7       8       1       5
```

If you were worried about making an extra copy of the values, you could use a foreach loop instead:

```
foreach key, (h.KEYS())[s] do print, key, h[key]
a       7
b       8
c       1
d       5
```

This will work in IDL 8.1.

Perhaps a better approach would be for us to add a SORT keyword to the KEYS() method. So you could do something like:

```
  foreach key, h.KEYS(/SORT) do print, key, h[key]
```

What do you all think about that?

-Chris
ITTVIS
