

Re: int/long unification hides bugs

Source: <http://coding.derkeiler.com/Archive/Python/comp.lang.python/2004-10/4443.html>

From: Alex Martelli (*aleaxit_at_yahoo.com*)

Date: 10/27/04

Date: Wed, 27 Oct 2004 09:02:01 +0200

kartik <kartick_vaddadi@yahoo.com> wrote:

...
> > *Try doing some accounting in Turkish liras, one of these days. Today,*
...
> > *[...]Even just for accounting, unlimited-size integers are simply much*
> > *more practical.*

I see you quote this but can't refute it...

> > > *as another example, using too long a string as an index into a*
> > > *dictionary is not a problem (true, the dictionary may not have a*
> > > *mapping, but i have the same issue with a short string). but too long*
> > > *an index into a list rewards me with an exception.*
> >
> > *But the same index, used as a dictionary key, works just fine. Specious*
> > *argument, therefore.*
>
> *I don't think so. I didn't say that large numbers always cause*
> *trouble, so you can't claim to have refuted by argument by giving a*
> *single counter-example.*

I rewrote your post, using long strings instead of large numbers, to show the arguments are exactly identical, and equally bereft of substance, against getting either unlimited strings or numbers as the default. You tried to show asymmetry by comparing strings used as keys into a dictionary vs ints used as indices into a list, and I refute that silly attempt: if you have dictionaries you can use long strings or large numbers as keys into them just as well.

Your mention of lists, in fact, shows exactly how specious your arguments for a default integer limit of $2^{31}-1$ are. That totally arbitrary limit has nothing to do with the size of any given list; the size of number that would give problems when used as list index varies, but it's more likely to be a few millions, than billions. *Moreover*, as soon as you try to use a too-large index for the specific list, you get an `IndexError`. It's therefore totally useless to try and get an `OverflowError` instead if the index, besides being too big for that specific list, is also over $2^{31}-1$ (or other arbitrary boundary).

> > *As common and everyday a*
> > *computation (in some fields) as the factorial of 1000 (number of*
> > *permutations of 1000 objects) is 2^{8530} — and combinatorial arithmetic*
> > *is anything but an "ivory tower" pursuit these days, and factorial is*
> > *the simplest building block in combinatorial arithmetic.*
>
> *It's nice to get some facts, rather than an attempt to prove your*
> *position by analogy between ints & strings ("Proof by analogy is*
> *fraud" — Bjarne Stroustrup)*

Go use C++, then, and stop wasting our time. If we preferred the way Stroustrup designs programming languages, to that in which van Rossum designs them, we'd be over in comp.lang.c++, not here in comp.lang.python — ever thought of that?

The analogy I posed, and still defend, merely shows your arguments were badly thought out, weak, and totally useless in the first place. It does not need to 'prove' anything, because we're neither in a court of law, nor in mathematics: it just shows up your arguments for the worthless froth they are. The facts (that should be obvious to anybody, of course) that the factorial function easily makes very big numbers, that some countries have very devalued currencies, etc, further show that big numbers (just like big strings) are useful to practical problems, thus your totally wrong-headed request to put default limits on numbers would also cause practical damage — for example, it would make an accounting-arithmetic package designed with strong currencies in mind (Euros, dollars, pounds, ...) unusable for weak currencies, because of the default nature of the limits.

Fortunately there is no chance whatsoever that Python will get into reverse gear and put back the arbitrary default limits you hanker for. If the many analogies, arguments, and practical examples that have been offered to help you see why, help you accept the fact, good. If not, good riddance — you have not offered one sound and useful line of reasoning throughout this big thread, after all, so it's not as if losing your input will sadly impoverish discussions here.

Alex