Subject: Re: random integers between 0 and 1,000,000
Posted by steven.leopardi on Sun, 23 Oct 2005 02:44:35 GMT
View Forum Message <> Reply to Message

Not sure why you're getting negative values here, randomu gives numbers
between 0 and 1..., but try this

```
weights  = long(1000000*RANDOMU( seed, 30 ))
help,weights
print,weights

WEIGHTS        LONG      = Array[30]

849159      15356     438991     564245     872939     652027
127233     367133     577606     326948     279024     421151
564968     521128     605257     133419
    512581     827491     231006     287929     870252
26568     630852     541590     872538     240737     513706
184865     955750     911741
```

RPF

Subject: Re: random integers between 0 and 1,000,000
Posted by peter.albert@gmx.de on Mon, 24 Oct 2005 06:12:02 GMT
View Forum Message <> Reply to Message

Hi Mike,

I'd guess you got the negative numbers because you defined the array
weights beforehand to be of integer type. Assigning it with values of
type long will lead to negative numbers if the values are larger than
2^15-1.

Cheers,

    Peter

Subject: Re: random integers between 0 and 1,000,000
Posted by Norbert Hahn on Mon, 24 Oct 2005 08:59:50 GMT
View Forum Message <> Reply to Message

"Peter Albert" <peter.albert@gmx.de> wrote:

> Hi Mike,
>

> I'd guess you got the negative numbers because you defined the array

Unfortunately your guess does not correspond to the docu:

> weights beforehand to be of integer type. Assigning it with values of
> type long will lead to negative numbers if the values are larger than
> 2^15-1.

fix   uses 15+1 bits for integer numbers (one bit is used for sign)
long   uses 31+1 bits for integer numbers ...
ulong   uses 32 bits for unsigned numbers. There is no sign.
long64...

So the original problem comes from interpreting the internal bits of
unsigned numbers. The preferable function for transforming 0...1 float
to integer without sign would be long.

Norbert

Subject: Re: random integers between 0 and 1,000,000
Posted by peter.albert@gmx.de on Mon, 24 Oct 2005 13:23:09 GMT
View Forum Message <> Reply to Message

I did not suggest using fix(), and using long() would also not help, if
the array itself is defined to be of type integer. Anyway your point
about interpreting internal bits is perfectly correct. However, the
problem is not about transforming float numbers, as he deals with long
numbers all the time. My point is that although IDL allows you to be
quite sloppy with data types most of the time, you can get problems
when assigning values to individual elements of an array. In that case,
the whole array does not automatically adjust its data type.

Cheers,

    Peter

Subject: Re: random integers between 0 and 1,000,000
Posted by James Kuyper on Mon, 24 Oct 2005 14:04:42 GMT
View Forum Message <> Reply to Message

Norbert Hahn wrote:
> "Peter Albert" <peter.albert@gmx.de> wrote:
>
>> Hi Mike,
>>

>> I'd guess you got the negative numbers because you defined the array
>
> Unfortunately your guess does not correspond to the docu:

I don't follow you. How does his guess fail to correspond to the
docu[mentation?]?

>> weights beforehand to be of integer type. Assigning it with values of
>> type long will lead to negative numbers if the values are larger than
>> 2^15-1.
>
> fix uses 15+1 bits for integer numbers (one bit is used for sign)
> long uses 31+1 bits for integer numbers ...
> ulong uses 32 bits for unsigned numbers. There is no sign.
> long64...

Agreed. Therefore, if the code the original poster showed us were
preceeded by the following statement:

weights = intarr(29)

then the expression:

weights[i] = ULONG(1000000 * RANDOMU(seed, 1))

takes a 32 bit unsigned long with a value somewhere in the range from 0
to 1000000, and converts it into a 16 bit signed int, with a range from
-32768 to 32767. This is the only way that weights[i] could ever gain a
negative value from that expression. For any other data types, all of
the values would be positive.

> So the original problem comes from interpreting the internal bits of
> unsigned numbers. The preferable function for transforming 0...1 float
> to integer without sign would be long.

Using 'long' rather than 'ulong' wouldn't do any good if the type of
"weights" itself is INT.

Note to original poster: loops are very inefficient in IDL. You'd be
better off writing:

weights[*] = LONG(1000000*RANDOMU(seed, 29))

If the length of "weights" happens to be 29, and not something longer,
you're even better off using:

weights = LONG(1000000*RANDOMU(seed,29))

Subject: Re: random integers between 0 and 1,000,000
Posted by Norbert Hahn on Mon, 24 Oct 2005 16:36:55 GMT
View Forum Message <> Reply to Message

"James Kuyper" <kuyper@wizard.net> wrote:

> takes a 32 bit unsigned long with a value somewhere in the range from 0
> to 1000000, and converts it into a 16 bit signed int, with a range from
> -32768 to 32767.

I took a closer look on what might have gone on. I ran the following
program:

```
z = randomu(seed,30)
a =  long (z*1000000) & print, a
b = ulong (z*1000000) & print, b
i = fix(a)
print, a
```

I found that a(1) was negative (-20848). So I printed

```
print, a[1], format="(z8)"
print, b[1], format="(z8)"
print, i[1], format="(z4)"
```

and got 4AE90 for both a[1] and b[1] and I got AE90 for i[1]

Thus the fix function simply takes the 16 low order bits from either long
or ulong variable and stores it into the result.

Norbert

Subject: Re: random integers between 0 and 1,000,000
Posted by Norbert Hahn on Mon, 24 Oct 2005 16:38:56 GMT
View Forum Message <> Reply to Message

I'm sorry for my harsh post.

"Peter Albert" <peter.albert@gmx.de> wrote:

> My point is that although IDL allows you to be
> quite sloppy with data types most of the time, you can get problems
> when assigning values to individual elements of an array. In that case,
> the whole array does not automatically adjust its data type.

This is often overlooked. Thank you very much for pointing this out!

Norbert

Subject: Re: random integers between 0 and 1,000,000
Posted by James Kuyper on Mon, 24 Oct 2005 19:06:16 GMT
View Forum Message <> Reply to Message

Norbert Hahn wrote:
> "James Kuyper" <kuyper@wizard.net> wrote:
>
>> takes a 32 bit unsigned long with a value somewhere in the range from 0
>> to 1000000, and converts it into a 16 bit signed int, with a range from
>> -32768 to 32767.
>
> I took a closer look on what might have gone on. I ran the following
> program:
>
> z = randomu(seed,30)
> a = long (z*1000000) & print, a
> b = ulong (z*1000000) & print, b
> i = fix(a)
> print, a
>
> I found that a(1) was negative (-20848). So I printed

I was unable to reproduce that result. I tested with several million
random numbers, and never once got a negative value from
long(1000000*randomu(seed,N)). Could you identify the value of z[1]
that gave you that value for a[1]?