CoCoALib - Design #1678

Unsigned long for indexes (& sizes?)

20 Apr 2022 11:30 - John Abbott

Status:	In Progress	**Start date:**	20 Apr 2022
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	10%
Category:	Safety	**Estimated time:**	0.00 hour
Target version:	CoCoALib-0.99880	**Spent time:**	1.25 hour

Description

So far we have avoided using unsigned integer types, and recommend using just long.

I now have doubts about this: *e.g.* because std::vector expects an unsigned type for index values.

Consider whether to change to unsigned long (or ULL?) for indexes, and sizes/lengths.
This will require reviewing all loops which count down to 0!

PS this was partly stimulated by notes I have written for my course (& that signed integer overflow is "undefined behaviour", and not "implementation defined" as I had believed).

History

#1 - 20 Apr 2022 11:37 - John Abbott

I spoke to Anna yesterday about this. Naturally, she was uneasy about reversing a decision we made a long time ago.

Undoubtedly, it will take time to chase through all changes [if we do decide to change].

I wonder whether should should introduce a "typedef" (type alias)? Maybe **ulong** instead of the longer/clumsier unsigned long.
Perhaps even use a name which indicates the purpose (rather than the probably definition)?
[while unsigned long is likely to be the definition, it could be something else: *e.g.* size_t]

Qn: *Are polynomial degrees the same as indexes?*

HINT grep for **for** and double minus; should find most cases
Also grep for [^-]--[^->]

#2 - 05 Aug 2022 17:27 - John Abbott

- Status changed from New to In Progress

- % Done changed from 0 to 10

The students in my course, who already had some experience in C++ and other similar languages, definitely expected size_t to be used for indexes, and container sizes (perhaps because that it what they were taught).

Anna suggested that len could return a long rather than a size_t, and noted that the different name could be used a signal of this difference.

A real problem is Microsoft's C++ compiler which (by default) has 32-bit long values, and reserved 64 bits for long long. This means that if we do choose to stay with a signed type, we must probably use long long for portability (or else something like int64_t). At this point size_t seems to be a simpler option so long as users do not use unsigned values carelessly in arithmetical expressions...

#3 - 07 Mar 2024 20:25 - John Abbott

- Target version changed from CoCoALib-0.99850 to CoCoALib-0.99880