Subject: string[] causes many overload complaints
Posted by aftershock on Sun, 01 Mar 2009 17:17:35 GMT
View Forum Message <> Reply to Message

I had a code:

```
unsigned int i=0;
String s;
if (s[i]==' ')
{

}
```

and Microsoft Compiler complaint a lot (cannot decide which
overload to use....

because of s[i]

My solution was to add:

```
int operator[](unsigned int i) const { ASSERT(i >= 0 && i <= B::GetCount()); return B::Begin()[i]; }
```

line in String.h

Subject: Re: string[] causes many overload complaints
Posted by mirek on Mon, 02 Mar 2009 07:44:36 GMT
View Forum Message <> Reply to Message

Why do not you just use 'int'?!

Mirek

Subject: Re: string[] causes many overload complaints
Posted by mr_ped on Mon, 02 Mar 2009 07:47:36 GMT
View Forum Message <> Reply to Message

I prefer personally unsigned as index variable, because it makes it easy to check for negative indexes.

Subject: Re: string[] causes many overload complaints
Posted by Mindtraveller on Mon, 02 Mar 2009 08:06:22 GMT
View Forum Message <> Reply to Message

Strange. Could you please explain how checking negative for unsigned int is better than using signed? As far as I understand, you check one bit? So what makes you think that (i == -1) is quicker than (i & 0x80000000)?

Subject: Re: string[] causes many overload complaints
Posted by mr_ped on Mon, 02 Mar 2009 09:15:43 GMT
View Forum Message <> Reply to Message

Uhm, I was sort of joking in a cryptic way, so to make myself clear.
If you need a range from 0 to MaxN, and you have index I, with "signed" type you need to validate I by:
is_valid = (0 <= I && I <= MaxN);

With "unsigned" type for I the very same condition turns into:
is_valid = (I <= MaxN);

This is why I like to use 0..N ranges indexed by unsigned variables, the safety checks then cost me just single compare, not two of them.

Addendum:
And AFAIK there's no special use for negative index values in NTL containers, ranges are always from 0, so if I would design the NTL containers, the basic [] operator would work with unsigned type, not signed. Probably breaking the convention of many programmers writing "for (int i=0; i<max; i++)" ... I never care too much about conventions anyway, unless they make sense, and this one does not.

Subject: Re: string[] causes many overload complaints
Posted by mirek on Mon, 02 Mar 2009 15:40:42 GMT
View Forum Message <> Reply to Message

mr_ped wrote on Mon, 02 March 2009 04:15
And AFAIK there's no special use for negative index values in NTL containers, ranges are always from 0, so if I would design the NTL containers, the basic [] operator would work with unsigned type, not signed. Probably breaking the convention of many programmers writing "for (int i=0; i<max; i++)" ... I never care too much about conventions anyway, unless they make sense, and this one does not.

Relative offsets can be negative. Obviously, in the end it always ends as positive number, but values involved in computing the index value can be negative. Using 'int' may avoid some casting (or equivalent warnings).

Mirek

Subject: Re: string[] causes many overload complaints
Posted by aftershock on Mon, 02 Mar 2009 22:37:55 GMT
View Forum Message <> Reply to Message

funny
I got a lot of error because of that.

unsigned int could be about type safety.

Subject: Re: string[] causes many overload complaints
Posted by mirek on Tue, 03 Mar 2009 07:20:11 GMT
View Forum Message <> Reply to Message

aftershock wrote on Mon, 02 March 2009 17:37funny
I got a lot of error because of that.

unsigned int could be about type safety.

unsigned int index1, index2;
int offset = index1 - index2;

Mirek

Subject: Re: string[] causes many overload complaints
Posted by aftershock on Wed, 04 Mar 2009 11:07:31 GMT
View Forum Message <> Reply to Message

Could you explain what you wrote in a more detailed way?

Why did you write that?
What are you trying to show?

Subject: Re: string[] causes many overload complaints
Posted by Infausto on Wed, 04 Mar 2009 15:48:47 GMT
View Forum Message <> Reply to Message

i suppose mirek wrote that to demonstrate you than an index can be negative when you calculate an offset.

salute.

Subject: Re: string[] causes many overload complaints
Posted by aftershock on Thu, 05 Mar 2009 11:09:37 GMT
View Forum Message <> Reply to Message

Ok,

I do not know why would anyone need an offset.
Personally, I do not subtract indecies.

Even if you do, I guess you do not put them into an a[].
If you do, how often does one need an operation like that?
I almost never.
For that reason, to make the type of indices int seems to me an overkill.

Anyway, there could be a definition for both type of indices....
with int and unsigned type....
STL uses unsigned types.
Where can you see a problem with that?

Subject: Re: string[] causes many overload complaints
Posted by mr_ped on Thu, 05 Mar 2009 11:29:11 GMT
View Forum Message <> Reply to Message

I can imagine offsets needed, but:
1) I rarely need them
2) they work well even when they are declared as unsigned, and it doesn't hurt my mind to do things like index += unsigned(-1);, actually it hurts much less then having to do NTL[int(unsigned_index)] every time I have to work with container value.

Quote:Where can you see a problem with that?

It's purely matter of taste?
Except saving 1 compare when boundary checking with unsigned, and having 2*max_range (both

of them I like much more then signed offsets), I don't see any major difference between those two, so I never really bothered and I can see why Mirek insists on his way.
I'm just adding the int(..) casting when the compiler yells, for me personally it's not even worth to add unsigned variants into NTL, because I don't run into thing that often.