Labb 6: SharedPtr

Ni ska implementera er egen variant av std::shared_ptr och std::weak_ptr, som lämpligen namnges SharedPtr och WeakPtr. Den ska vara exception safe (vilket inte gör stor skillnad).

Observera att ni skall använda er av kontrollen för minnesläckor.

G nedan står för vad som krävs för G och VG för VG. WeakPtr behövs bara för VG.

Operationerna nedan stämmer med STLs förutom att STL har fler member functions och static functions.

	SharedPtr	**WeakPtr**	**Kommentar**
- Konstruktor som tar:			
o void dvs. ()	G	VG	
o En pekare	G		
o En SharedPtr&	G	VG	
o En SharedPtr&&	G		Move-constructor
o En WeakPtr	VG	VG	Throw exception!
- Destruktor	G	VG	
- Tilldelning från en			
o SharedPtr&	G	VG	
o SharedPtr&&	VG		Move-assignmnet
o WeakPtr		VG	
- Jämförelse med (== och <)			
o nullptr	G		
o SharedPtr	G		Jämförelse av den underliggande pekaren
- operator*	G		
- operator->	G		
- operator bool	G		True om det finns ett objekt
- funktioner:			
o reset(T*=nullptr)	G		Byter objekt
o get()	G		Ger tillgång till pekaren
o unique()	G		
o lock()		VG	
o expired()		VG	
void Check() Skall göra assert på Invarianten så det blir exception om något är fel.	G	VG	Denna funktion är bara till för testprogrammet.
- non member funktion:			
template<class T> void swap(SharedPtr<T>& lhs, SharedPtr<T>& rhs) noexcept;	G		Är lämpligen en friend funktion
template<class T> void swap(WeakPtr<T>& lhs, WeakPtr <T>& rhs) noexcept;		VG	Är lämpligen en friend funktion

1 G

Gör det som är märkt med G ovan, WeakPtr skall inte alls implementeras.

2 VG

För VG så krävs att även WeakPtr implementeras vilket även förändrar hur SharedPtr implementeras. Gör det som är märkt med VG ovan. Lägg märke till att WeakPtr har få metoder, för att använda en WeakPtr så får man först gör an SharedPtr från den med Lock eller med SharedPtr(const WeakPtr&) constructorn.

Det krävs även att det hela tiden allokeras så lite minne som möjligt, det ska inte allokeras minne förrän det är tvunget och avallokeras så fort som möjligt. Detta innebär att:

- så fort en weakPtr används så ska den om den är expired räkna ner referensräknaren så att referensräknarobjektet kan deletas så fort som möjligt.
- Fixa konstruktorerna så att de kan ta alla sorters pekare av kompatibel typ.

Testprogrammet i Main.cpp
Observera att det testprogram som finns i Main.cpp bara är en hjälp och varken fullständigt eller garanterat helt korrekt. Det är möjligt att testprogrammet kör felfritt fast er lösning är felaktig. Det är även möjligt – men inte troligt – att er lösning är korrekt fast testprogrammet inte kör/kompilerar felfritt.
Observera att ni i filen VG.h styr om testprogrammet är för VG eller G.