Programmierung.

16 Programmiere einen überlappenden Schwarz-Vorkonditionierer ohne Grobgitter. Z.B.:

```
class OSPreconditioner
{
public:
  void init (const SMatrix& K, const vector<int>& blockstart,
             const vector<int>& blocklength);

  void solve (const Vector& r, Vector& w) const;

  ...
};
```

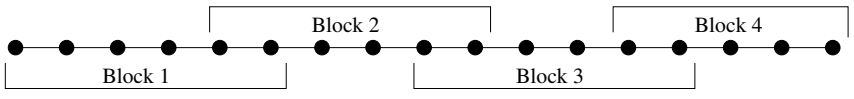
Hierbei ist `K` die Steifigkeitsmatrix, und `blockstart[i]`, `blocklength[i]` definieren den i-ten Block, d.h. jene Knoten, deren Basisfunktionen den i-ten Unterraum aufspannen.
Hinweis: Speichere die entsprechenten Teilmatrizen ab und verwende den Thomas-Algorithmus (Tridiagonallöser).

17 Teste den Vorkonditionierer (ohne Grobgitter) für das Problem aus Übung 5. Untersuche die Anzahl der Iterationen sowie die geschätzte Konditionszahl für die folgenden Parameter:

1024 Elemente, 64 ca. gleich große Blöcke

Knoten im Overlap	Knoten pro Block	PCG-Iterationen	Konditionszahl
0	ca. 16		
2	ca. 17		
4	ca. 19		
8	ca. 23		
16	ca. 31		

Zur Veranschaulichung: Das untenstehende Beispiel zeigt ein Gitter mit 16 Elementen, 4 Blöcken mit ca. 6 Knoten pro Block und 2 Knoten im Overlap.



18 Programmiere eine Routine

```
void prolongate (const Mesh& coarse_mesh, const Mesh& fine_mesh,
                 const Vector& coarse_vec, Vector& fine_vec);
```

welche eine Grobgitter-Funktion am Feingitter darstellt. Es darf angenommen werden, dass Grobgitterknoten auch Feingitterknoten sind.

Vorschau: Wir werden den obigen Vorkonditionierer um einen Grobgitterlöser erweitern.