Es sind die vorgegebenen Datentypen zu beachten (int Werte werden in Variable vom Typ int eingelesen und double Werte in Variable vom Typ double).

Keine Zeichenketten oder Strings verwenden, wenn Zahlen verlangt sind.

Wenn ein Ergebnis berechnet werden soll, dann reicht die Ausgabe alleine (außer wenn nur diese explizit verlangt ist) nicht aus. Der Wert muss in einer Variable zur Verfügung stehen, sodas er prinzipiell für weitere Berechnungen verwendbar wäre.

Verwendung von globalen Variablen und goto ist (soweit sie nicht in der Angabe explizit empfohlen wird) untersagt.

Achten Sie darauf, dass die Objekte Ihrer Klasse in jedem Fall in einem konsistenten Zustand sein müssen.

Erweitern Sie die Klasse Lagerplatz (Evaluierungsaufgabe der Vorwoche) und schreiben Sie einen Kopierkonstruktor und einen Kopierzuweisungsoperator, die jeweils den dynamisch allozierten Speicher ordnungsgemäß kopieren.

Schreiben Sie weiters einen Ausgabeoperator << für die Klasse Lagerplatz, der die gespeicherten Werte in folgendem Format ausgibt:

ein/aus x Stück, … z.B.:

aus 3 Stück, ein 4 Stück, ein 2 Stück

Die Angabe der Vorwoche:

Implementieren Sie eine Klasse Lagerplatz. An einem Lagerplatz können beliebig viele Aus- und Einlagerungen durchgeführt werden. Für jeden Vorgang ist zu speichern, wie viel Stück (int, für Auslagerungen negativ) gelagert bzw. entnommen wurden. (Zum Speichern ist ein dynamisches Array von int Werten notwendig.)

Schreiben Sie einen Konstruktor mit einem Parameter, der die Startgröße des dynamischen Arrays festlegt.

Schreiben Sie eine Methode void bewegen(int), die eine Aus- bzw. Einlagerung durchführt. Beachten Sie, dass die Maximallänge des Arrays nicht überschritten werden darf. Ein dynamisches Wachstum des Arrays ist nicht notwendig.

Schreiben Sie einen Destruktor, der das dynamisch allozierte Array wieder ordnungsgemäß freigibt.

Erweiterung (30 Minuten zusätzlich):

Schreiben Sie eine operator[] Methode, die in einem dynamisch allozierten Array die Nummern (Indizes) aller Lageroperationen retourniert, nach denen mindestens soviele Stück auf Lager waren, als der gegebene Indexwert angibt. Die Länge des retournierten Arrays soll dabei im Array selbst beim Index 0 gespeichert werden.

```
Lagerplatz l(10);
l.bewegen(7); l.bewegen(-5); l.bewegen(2); l.bewegen(4);
//l[5] soll nun ein int-Array mit dem Inhalt {2,0,3} liefern
```

Testen Sie Ihre Klasse ausgiebig.

Eine mögliche Lösung:

```
#include <iostream>
 using namespace std;

 class Lagerplatz {
  int len;
  int pos;
  int *trans;
 public:
  explicit Lagerplatz(int len): len(len), pos(0) {
    if (len<=0) throw "Laenge muss groesser 0 sein";
    trans = new int[len];
  }
  ~Lagerplatz() {
    delete[] trans;
  }
  void bewegen(int anz) {
    if (pos<len)
      trans[pos++]=anz;
  }
  void print() {
    cout<<'{';
    for (int i=0; i<pos; ++i) {
      cout<<trans[i];
      if (i<pos-1) cout<<", ";
    }
    cout<<'}';
  }
```

```
 int *operator[](int stand) {
   int *res = new int[len+1];
   int p = 1;

   for (int i=0; i<pos; ++i) {
     sum+=trans[i];
     if (sum>=stand)
       res[p++]=i;
   }
   res[0]=p-1;
   return res;
 }
};
```