



Aufgabe 1. Implementiere doppelt verkettete Listen, die `double`-Variablen speichern.

```
1  /* Definiere hier angemessene Strukturen für einen
2     einzelnen Listeneintrag und die Liste selbst. */
3
4  /* Leere Liste erstellen */
5  LIST *list_create();
6
7  /* Element hinter E einfügen, NULL heißt am Anfang */
8  LISTNODE *list_insert(LIST *L, LISTNODE *E, double p);
9
10 /* Element am Anfang bzw. Ende einfügen */
11 LISTNODE *list_unshift(LIST *L, double p);
12 LISTNODE *list_push(LIST *L, double p);
13
14 /* Element am Anfang bzw. Ende entfernen und
15     die Daten zurück geben */
16 double list_shift(LIST *L);
17 double list_pop(LIST *L);
18
19 /* eine Element aus der Liste entfernen */
20 void list_delete(LIST *L, LISTNODE *E);
21
22 /* zwei Listen zusammenfügen */
23 LIST *list_merge(LIST *L, LIST *M);
24
25 /* Liste inklusive allen Elementen frei geben */
26 void list_free(LIST *L);
```

Aufgabe 2. Schreibe Funktionen, die aus einer verketteten Liste ein `DBLARRAY` (von gestern) und umgekehrt machen.