Implementieren Sie wieder ein *Dictionary A* unter Verwendung von *Hashing* (siehe Praktikum 4). Verwenden Sie für die Implementierung nun aber das *Hashing* mit verketteten Listen (*resolution by chaining*). Dazu definieren Sie ein Array aus m Listenköpfen (diese sollten vom Typ der Listenelemente sein). Für eine Eingabe a berechnen Sie dann mit der Hashfunktion $h(a) = a \bmod m$ eine Behälternummer (den Arrayindex). Für die Operation `insert` nehmen Sie dann die Zahl in die lineare Liste der entsprechenden Arrayposition auf. Sie müssen keine Dubletten verhindern (d.h. eine Zahl darf mehrfach eingetragen werden). Wenn nicht genügend Speicherplatz zur Verfügung steht, dann liefert die Funktion `insert` den Wert 0, ansonsten den Wert 1. Die Funktion `insert` soll in konstanter Laufzeit arbeiten. Für die Operation `member` durchsuchen Sie die Liste an der Arrayposition $h(a)$. Geben Sie nur dann den Wert 1 zurück, wenn sich a in der Liste befindet. Die Funktion `delete` liefert den Wert 1, falls der Schlüssel (das Listenelement) gelöscht werden konnte. Ansonsten wird der Wert 0 geliefert.
Als Test lassen Sie die folgende Sequenz ablaufen (`SIZE` entspricht der Anzahl der Arrayelemente).

```
int i;
for(i = 1; i <= 2 * SIZE; i++){
  printf("%d", insert(i));
}
for(i = 1; i <= SIZE; i++){
  printf("%d", member(i));
}
for(i = SIZE+1; i <= 2*SIZE; i++){
  printf("%d", delete(i));
}
for(i = 1; i <= 2*SIZE; i++){
  printf("%d", member(i));
}
printf("\n");
```

Es muss zunächst eine Folge von `5 * SIZE` Einsen ausgegeben werden. Direkt danach erfogt die Ausgabe von `SIZE` Nullen.
Schreiben Sie dann ein weiteres Hauptprogramm. Nehmen Sie zunächst alle Zahlen in das Dictionary auf, die über die Kommandozeile übergeben werden. Lesen Sie danach Zahlen über die Tastatur ein. Für jede eingegebene Zahl wird eine Meldung ausgeben, ob sich die Zahl im Dictionary befindet. Nach Eingabe der Zahl -1 wird das Programm beendet.