ulm university universität
uulm

Institut für Angewandte Informationsverarbeitung
Dr. Andreas F. Borchert
Fabian Berstecher

9. Dezember 2016
Blatt 8

Systemnahe Software I (WS 2016/2017)

Abgabe bis zum 23. Dezember 2016, 14:00 Uhr

Lernziele:

- Eine Speicherverwaltung selbst entwerfen und programmieren

Aufgabe 10: Eigene Speicherverwaltung (20 Punkte)

Im Rahmen dieses zweiwöchigen Blattes ist eine eigene Speicherverwaltung zu entwickeln. Es steht Ihnen dabei nur die folgende Funktion zur Belegung weiteren Speichers vom System zur Verfügung:

```
1   void* get_block_from_system();
```

Diese Funktion liefert einen Zeiger auf einen freien Speicherblock der Größe *BLOCKSIZE* (8192 Bytes) zurück. Die anderen Funktionen der Standardbibliothek wie *malloc()* oder *calloc()* dürfen nicht verwendet werden. Wenn Sie Speicherplatz für dynamische Datenstrukturen zur Verwaltung benötigen, ist dieser mit in dem über *get_block_from_system()* bezogenen Speicher unterzubringen.

Im Rahmen der Aufgabe sind die folgenden Funktionen zu implementieren:

void *init_my_alloc()* Diese Funktion wird zu Beginn einmal aufgerufen, um die eigene Speicherverwaltung zu initialisieren. Wenn das nicht nötig ist, muss die Funktion trotzdem (mit einem leerem Rumpf) implementiert werden.

void* *my_alloc*(*size_t size*) Liefert einen Zeiger auf eine vom Aufrufer anschließend frei verwendbaren Speicherbereich der Größe *size* zurück. Dieser Speicherplatz muss in einem Bereich sein, der vorher vom System mit *get_block_from_system* geholt worden ist.

void *my_free*(void* *ptr*) Gibt einen zuvor mit *my_alloc()* reservierten Speicherbereich wieder frei, d.h. der Speicherbereich darf später von *my_alloc* wieder zur Verfügung gestellt werden.

Dabei ist folgendes zu beachten:

- Sie können davon ausgehen, dass die angeforderte Größe des Speicherbereichs bei einem Aufruf von *my_alloc* ein Vielfaches von 8 ist und dass kein Speicherblock mit einer Größe von mehr als als 256 Byte angefordert wird. Alle zurückgegebenen Speicherbereiche müssen an einer Adresse, die ein Vielfaches von 8 ist, beginnen.

  Die Funktion *get_block_from_system* hält sich daran.

- Ihre Implementierung darf nicht mehr als 256 Bytes für globale Variablen verwenden (unter der Annahme, dass auf einer 32-Bit-Architektur gearbeitet wird, bei der die Zeiger in vier Bytes passen). Hierbei zählen auch **static**-Variablen innerhalb der Funktionen.

- Zu einer vollständigen Lösung gehört auch eine kurze Beschreibung der verwendeten Datenstrukturen und Ihres Verfahrens zur Speicherverwaltung.

- Eine vollständige Lösung sollte mindestens 50.000 Operationen in erträglicher Zeit durchführen können.

- Es sollte nicht mehr als 30 % überschüssiger Speicher vom System geholt werden.

Sie können davon ausgehen, dass *my_alloc* und *my_free* korrekt verwendet werden. Das bedeutet, dass nur Speicher freigegeben wird, der vorher auch belegt wurde, dass die angeforderte Größe ein Vielfaches von 8 und nicht größer als 256 ist, etc. Es muss natürlich damit gerechnet werden, dass der mit *my_alloc* geholte Speicher auch tatsächlich verwendet wird.

Laden Sie die Testumgebung von der Vorlesungshomepage herunter, packen Sie diese mit *tar* aus und bearbeiten Sie die Datei *my_alloc.c* entsprechend den obigen Anforderungen.

Sie können Ihre Lösung wieder mit *submit* einreichen:

```
submit ss1 10 team [notes] my_alloc.c
```

Viel Erfolg!