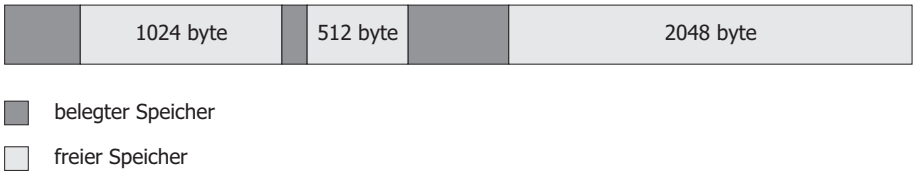
Übungsblatt 6

Abgabe: 3. Juni 2019

Aufgabe 6.1: Speicherbelegung bei Segmentierung (2 + 3 = 5 Punkte)

Gegeben sei folgende Belegung eines Speichers:



Weiterhin wird Segmentierung zur Belegung des Speichers verwendet und es seien vier Belegungsstrategien für Speicherplatzanforderungen aus der Vorlesung gegeben:

First-Fit	Belege von vorne beginnend den ersten freien Speicherbereich, der groß genug ist, die Anforderung zu erfüllen.
Rotating-First-Fit	Wie First-Fit, jedoch wird vom Ende des zuletzt ausgewählten Speicherbereichs aus ein passender Bereich gesucht. D.h., wurde bei obigem Speicherzustand eine Anfrage nach z.B. 400 Byte in der ersten Lücke von 1024 Byte Größe platziert, entsteht zwar eine neue Lücke von 624 Byte Länge, diese wird allerdings bei der nächsten Anfrage übersprungen und die Suche am Beginn der Lücke von 512 Byte Größe fortgesetzt. Wird das Ende des Speichers erreicht, so wird die Suche am Anfang des Speichers fortgesetzt. Bei der ersten Anforderung beginnt die Suche am Anfang des Speichers.
Best-Fit	Belege den kleinsten freien Speicherbereich, in den die Anforderung passt.
Worst-Fit	Belege den größten freien Speicherbereich, in den die Anforderung passt.

a) Wie wird der oben abgebildete Speicher belegt, wenn nacheinander vier Anforderungen der Größe 384 Byte, 640 Byte, 512 Byte und 2048 Byte gestellt werden? Notieren Sie für jede der vier Strategien die freien Speicherbereiche nach jeder Anforderung (z.B. in der Form (1024, 512, 2048) für die obige Ausgangsbelegung) und geben Sie an, für welche Strategien alle Anforderungen erfüllt werden können.

   Beachten Sie, dass die Daten jeweils linksbündig in einer Lücke abgelegt und einmal belegte Speicherbereiche nicht wieder freigegeben werden.

b) Sie dürfen nun die Reihenfolge der freien Speicherbereiche und die der Anforderungen vertauschen (aber keine freien Speicherbereiche zusammenfassen). Geben Sie für jede der vier Strategien eine Speicherbelegung und eine Anforderungsreihenfolge an, bei der jeweils nur diese Strategie erfolgreich alle Anforderungen erfüllen kann. Geben Sie auch jeweils für die erfolgreiche Strategie die Speicherbelegung nach Abarbeitung der einzelnen Anforderungen an.

Aufgabe 6.2: Adressen (0,5 + 3 = 3,5 Punkte)

Bei der Speicherverwaltung nach dem Segmentierungsverfahren sei für einen Prozess die folgende Segmenttabelle gegeben:

Segment	Basisadresse	Länge
0	3300	305
1	1900	198
2	1865	25
3	1710	145
4	2700	182
5	4300	55
6	4205	65

Sowohl Basisadresse als auch Länge seien in Byte angegeben.

a) Wieviele Byte stehen dem Prozess im physikalischen Speicher zur Verfügung?

b) Eine logische Adresse wird als Paar (Segment-Nummer, Offset) dargestellt. Berechnen Sie zu den folgenden physikalischen Adressen jeweils die logischen Adressen. Ist dies für alle physikalischen Adressen möglich oder versucht man auf Speicherbereiche zuzugreifen, die dem Prozess nicht zugeordnet sind? Was würde in diesem Fall geschehen?

1. 4270
2. 1810
3. 2888
4. 1935
5. 2777
6. 4222

Aufgabe 6.3: Speicherverwaltung mit Buddy-Systemen (3 + 1 + 3 + 1 = 8 Punkte)

Gegeben sei ein Rechner, der mit 32 GByte Hauptspeicher ausgerüstet ist. Dieser Speicher wird mit einem Buddy-System verwaltet und ist im Moment leer.

Die Speicherbelegung kann mit Hilfe eines Binärbaumes dargestellt werden. Dabei stellt die Wurzel den gesamten Speicher von 32 GByte dar, die nächste Ebene zwei Speicherbereiche der Größe 16 GByte, usw. Ist der Speicher leer, existiert nur die Wurzel. Bei einer Speicheranforderung wird der Baum zunächst von links nach rechts durchsucht, ob ein passender Speicherbereich existiert. Ein Speicherbereich der Größe 2^x ist dann passend, wenn für die Speicheranforderung der Größe s gilt: $2^{x-1} < s \leq 2^x$. Existiert kein solcher Speicherbereich, wird nach einem nächstgrößeren Speicherbereich (der Größe 2^{x+1}) gesucht. Falls kein solcher existiert, wird wieder nach einem nächstgrößeren gesucht (Größe 2^{x+2}) usw., bis ein Speicherbereich gefunden oder die Wurzel erreicht wird. Existieren zwei oder mehr gleich große Speicherbereiche, wird immer der am weitesten links im Baum liegende Bereich ausgewählt. Wurde ein Speicherbereich der Größe 2^{x+1} oder größer ausgewählt, wird er in zwei Bereiche (Buddies) geteilt und der linke Bereich ausgewählt. Ist er immer noch größer als benötigt, wird er weiter geteilt, so lange bis durch die letzte Teilung zwei Buddies der Größe 2^x entstehen; der linke Buddy wird der Anfrage zugeteilt. Wird kein Speicherbereich gefunden, der groß genug ist, wird die Suche erfolglos abgebrochen. Bei Freigabe von Speicherbereichen werden benachbarte freie Buddies so weit möglich zu einem einzigen freien Speicherbereich zusammengefasst.

Der Reihe nach werden nun Speicherbereiche folgender Größe angefordert bzw. freigegeben:

Nr.	Operation	Name	Größe in GByte
1	Anforderung	A	12
2	Anforderung	B	5
3	Anforderung	C	1
4	Anforderung	D	2
5	Freigabe	C	
6	Freigabe	B	
7	Anforderung	E	4
8	Anforderung	F	3

a) Stellen Sie die Speicherbelegung nach jeder Anforderung bzw. Freigabe als Binärbaum dar. Vor der ersten Anforderung sei der Speicher leer, d.h. es existiert nur die Wurzel, die den gesamten Speicher von 32 GByte darstellt.

b) Was ist die größtmögliche Speicheranforderung, die nach Ablauf aller acht Schritte noch gewährt werden kann? Welcher Anteil des gesamten Speichers geht durch interne Fragmentierung verloren?

c) Nun werden gewichtete Buddies betrachtet. Hierbei wird ein Speicherbereich nicht in zwei gleich große Unterbereiche aufgeteilt. Stattdessen werden Speicherbereiche, deren Größe ohne Rest durch 3 teilbar ist, in zwei Buddies im Verhältnis 1:2 aufgeteilt. Ist die Größe eines Bereichs durch 4, aber nicht durch 3 teilbar, werden zwei Buddies im Größenverhältnis 1:3 erzeugt. Ist die Größe weder durch 3 noch durch 4 teilbar, so hat ein Bereich seine minimale Größe erreicht. Bei der Aufteilung soll der kleinere Buddy stets links im Baum angeordnet werden.

   Wiederholen Sie die obigen Operationen, beginnend mit einem leeren Speicher von 32 GByte. Bei einer Speicheranforderung teilt der Algorithmus auch hier den kleinsten noch passenden Buddy zu. Wie oben wird zunächst nach einem Bereich passender Größe gesucht; wird keiner gefunden, wählt man auch hier einen größeren Speicherbereich und teilt ihn in zwei Buddies. Bitte beachten Sie, dass die Suche nach einem passenden Buddy komplexer ist als bei einem nicht gewichteten Buddy-System, da unter Umständen nach der Teilung eines Speicherbereiches in zwei Buddies der größere der Buddies ausgewählt werden muss, um zum kleinsten möglichen Speicherbereich zu kommen. (Nehmen Sie Sich z.B. Folie VII-27 zur Orientierung her, um herauszufinden, welcher Buddy als nächster geteilt werden muss.) Sollten mehrere Buddies gleicher Größe existieren, wird wiederum derjenige gewählt, der sich am weitesten links im Baum befindet. Es werde angenommen, dass die kleinste mögliche Größe eines Speicherbereichs 1 GByte betrage und ein Buddy nicht mehr weiter unterteilt werden kann, wenn dadurch einer der beiden entstehenden Bereiche kleiner als 1 GByte würde.

d) Was ist die größtmögliche Speicheranforderung, die in diesem System nach Ablauf aller acht Schritte noch gewährt werden kann? Welcher Anteil des gesamten Speichers geht nun durch interne Fragmentierung verloren?

Aufgabe 6.4: Page Table (1 + 0,5 + 2 = 3,5 Punkte)

Gegeben sei ein Betriebssystem, das Paging mit einer Seitengröße von 32 KiB einsetzt. Die logischen Adressen der Prozesse sind 48 Bit lang.

a) Es wird eine einstufige Page Table verwendet; wie viele Einträge hat die Page Table? Geben Sie an, wie Sie auf Ihr Ergebnis kommen.

b) Nun wird das System verändert: es wird dreistufiges Paging eingesetzt, wobei die äußere, die mittleren und die inneren Page Tables gleich groß sind. Welchen Umfang haben die Tabellen nun?

c) Welchen Vorteil hat mehrstufiges Paging? Welchen Nachteil hat es? Begründen Sie Ihre Aussagen knapp.