CS 354 - Machine Organization & Programming
Thursday, February 22, 2018

Project p2 (6%): DUE at 10 pm THIS SUNDAY, February 25th

Note: It is better to submit a working program using indexing
than a non-working program attempting to use address arithmetic.

Project p3 (6%): Assigned Tomorrow

Homework hw3 (1.5%): Assigned Tomorrow

Last Time

C's Heap Allocator (`stdlib.h`)
Posix `brk` (`unistd.h`)
Allocator Design
Simple View of Heap
Free Block Organization
Implicit Free List

Today

Implicit Free List (from last time)
Placement Policies
Free Block - Too Much/Too Little
Coalescing Free Blocks
Footers

Next Time

Heap Caveats
Read: B&O 9.9.11, 9.9.13
Skim: B&O 9.9.12

Placement Policies

Placement Policies

- First Fit (FF): start search from
  stop at
  fail if

  mem util:

  thruput:

- Next Fit (NF): start search from
  stop at
  fail if

  mem util:

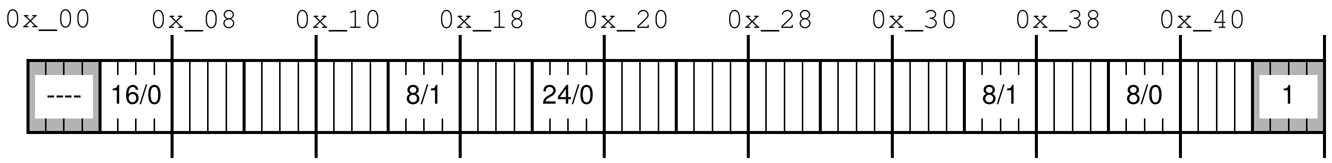
  thruput:

- Best Fit (BF): start search from
  stop at
  fail if

  mem util:

  thruput:

Heap Allocation Run 3 using a Placement Policy:



→ Given the original heap above and the placement policy, what address is `ptr` assigned?

```
ptr = malloc(14 * sizeof(char));    //FF?               BF?

ptr = malloc(3 * sizeof(char));     //FF?               BF?
```

→ Given the original heap above and the address of block most recently allocated, what address is `ptr` assigned using NF?

```
ptr = malloc(2 * sizeof(char));     //0x_14?              0x_34?

ptr = malloc(3 * sizeof(int));      //0x_14?              0x_34?
```

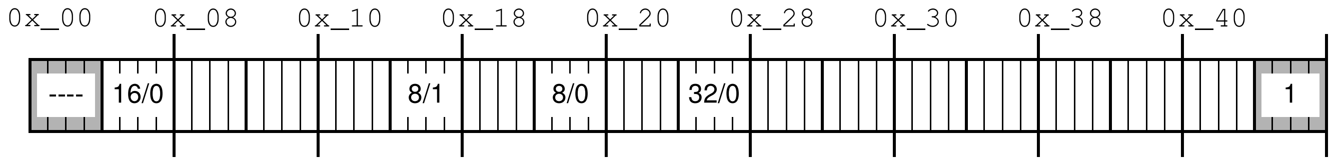
Free Block - Too Much/Too Little

What happens if free block chosen is bigger than the request?

◆

◆

Heap Allocation Run 4 using Splitting and using FF



→ Given the heap to be modified by the 4 mallocs below,
what address is assigned to each pointer?
If there is a new free block, what is its address and size in bytes?

```
1) ptr1 = malloc(sizeof(char));

2) ptr2 = malloc(11 * sizeof(char));

3) ptr3 = malloc(2 * sizeof(int));

4) ptr4 = malloc(5 * sizeof(int));
```

What happens if no free block is large enough to satisfy the request?

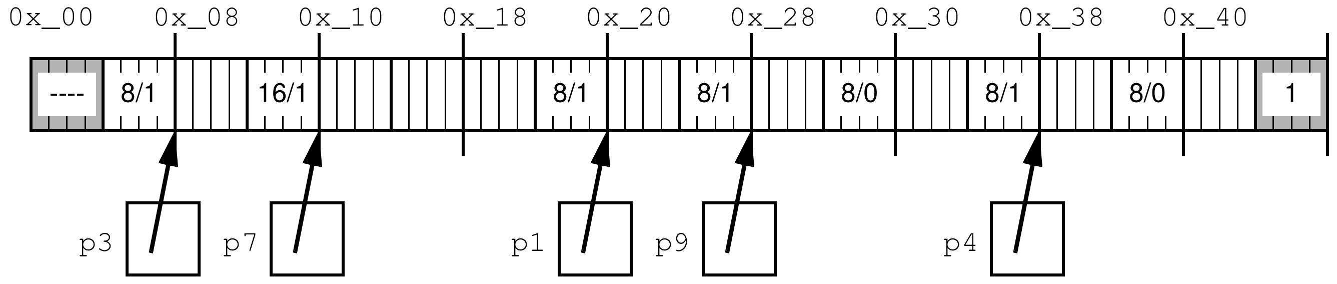
1st.

2nd.

3rd.

Coalescing Free Blocks

Heap Allocation Run 5



→ What's the problem resulting from the following heap operations using FF?

```
1) free(p9); p9 = NULL;
2) free(p1); p1 = NULL;
3) p1 = malloc(4 * sizeof(int));
```

Problem:

Solution:

immediate

delayed

→ Given the original heap above, what is the size in bytes of the freed heap block?

```
1) free(p7); p7 = NULL;
```

→ Given a pointer to a payload, how do you find its block header?

→ Given a pointer to a payload, how do you find the block header of the next block?

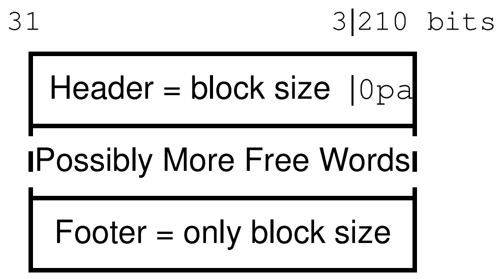
→ Given the modified heap above, what is the size in bytes of the freed heap block when immediate coalescing is used?

```
2) free(p3); p3 = NULL;
3) free(p1); p1 = NULL;
```

➢ Given a pointer to a payload, how do you find the block header of the previous block?

Footers

Heap <u>Free</u> Block Layout with Header and Footer

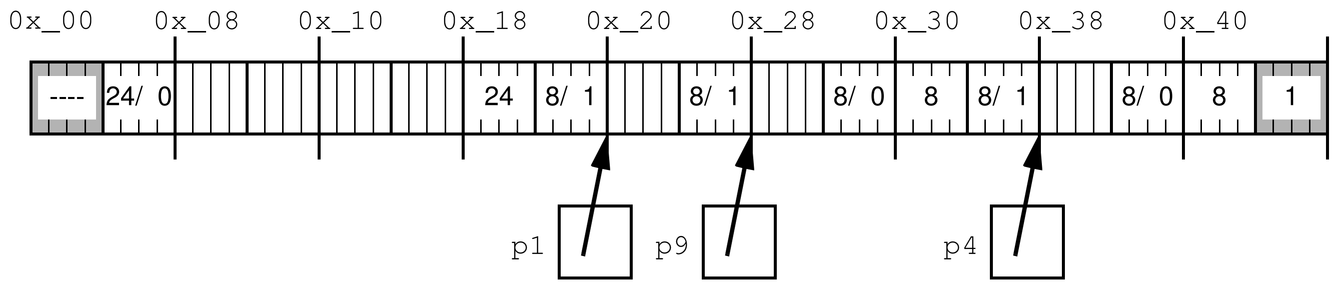


Footer (AKA Boundary Tag)

→ Why don't allocated blocks need footers?

→ Given a pointer to a payload, how do you get to the header of a previous block that's free?

Heap Allocation Run 6 with given Free List using Immediate Coalescing and Free Block Footers



→ Given the heap above, what is the size in bytes of the freed heap block?

```
1) free(p1);
```

→ Given the modified heap above, what is the size in bytes of the freed heap block?

```
2) free(p4);
```

➢ Is coalescing done in a fixed number of steps (constant time)
or is it dependent on the number of blocks (linear time)?