

Due: 9AM, Wednesday, **November 28** in class

10 points total

Please show your answers (and all work) on these sheets. Fold your paper the long way and write **your name** and **Homework 9** on the outside.

1. (4 points) For the following code fragment:

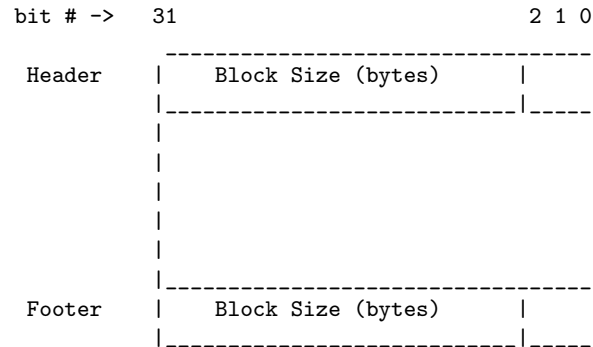
```
int i,n,intarray[512],sum =0;
for (i = 0; i < n; i++)
    sum += intarray[i];
```

- 1a. Rewrite the loop using 3-way loop unrolling. You cannot assume the existence of external functions.

- 1b. Convert the original loop to pointer code. You cannot assume the existence of external functions.

2. (6 points)

Consider an allocator that uses an implicit free list. The layout of each allocated and free memory block is as follows:



Each memory block, either allocated or free, has a size that is a multiple of eight bytes. Thus, only the 29 higher order bits in the header and footer are needed to record block size, which includes the header and footer. The usage of the remaining 3 lower order bits is as follows:

- bit 0 indicates the use of the current block: 1 for allocated, 0 for free.
- bit 1 indicates the use of the previous adjacent block: 1 for allocated, 0 for free.
- bit 2 indicates the use of the next adjacent block: 1 for allocated, 0 for free.

Given the contents of the heap shown on the left on the next page, show the new contents of the heap (in the right table on the next page) after a call to `free(0x400b010)` is executed. If the value on the left is unchanged in the table on the right, then use the old value on the right. Your answers should be given as 4-byte big-endian hex values. Note that memory addresses increase from the top of the page to the bottom. Assume that the allocator uses immediate coalescing, that is, adjacent free blocks are merged immediately when a block is freed. Assume also that for efficiency, the allocator only changes those bytes that must be changed.

0x400affc	+-----+		
0x400b000	+-----+		
0x400b004	+-----+		
0x400b008	+-----+		
0x400b00c	+-----+		
0x400b010	+-----+		
0x400b014	+-----+		
0x400b018	+-----+		
0x400b01c	+-----+		
0x400b020	+-----+		
0x400b024	+-----+		
0x400b028	+-----+		
0x400b02c	+-----+		

0x400affc	+-----+		
0x400b000	+-----+		
0x400b004	+-----+		
0x400b008	+-----+		
0x400b00c	+-----+		
0x400b010	+-----+		
0x400b014	+-----+		
0x400b018	+-----+		
0x400b01c	+-----+		
0x400b020	+-----+		
0x400b024	+-----+		
0x400b028	+-----+		
0x400b02c	+-----+		