Smarika Pathak
1001860795

Malloc Assignment Report

Executive Summary

This code allocates dynamic memory and uses linked list to track the memory use. We find out the best fit, worst fit, next fit, and the first fit (i.e. we've used four algorithms). Also, we join/combine two free blocks when they are adjacent. For example: we freed the block before and have another free block right next to it, we can combine them. However, if their size is larger than the requested size, we split them. We allocated memory and freed them.

Algorithms Implemented

We've implemented four algorithms in this code.

a) First Fit: In this algorithm, we search for the memory block that is at first and is free. If it is not large enough, then it enters the while loop, iterates over the linked list, and finds the first space that can fulfill the needed size. However, if there is not enough size, then the pointer returns NULL and terminates.
b) Best Fit: To find the best fit, we check the difference between the size of the current block that the pointer is pointing with the needed size. If the needed size is less than the difference and is greater or equal to zero (just fits the space), we use the memory block. If not, we go to the next block and do the same calculations.
c) Worst Fit: This algorithm is opposite to the best fit. To find the worst fit, we check the difference between the size of current block and the needed size. After that we compare it with the largest free block found. If the calculated difference is greater than the free block found, then it is the worst fit.
d) Next Fit: In this algorithm, we check if the last pointer is not NULL to know that current block is not the first block. If last is not NULL, we set the curr to next block. Then we find a free memory block with size greater or equal to needed size and set the last pointer to current block and move on to the next block. We scan through the block once.

Test implementation

We tested to find the first fit, next fit, worst fir, best fit, coalesce, and block split and reuse. The *ffnf* showed the first fit and next fit address. The first fit and next fit did not have any splits in them. It is because for these two algorithms, we find the first block and next block that is large enough to fit the needed size which does not require splitting the available other blocks. Similarly, we used *bfwf* to test the best fit and worst fit. There are splits during best fit and worst fit because we search for the block that is closest to the needed size and split it to fit the allocation request.

Test Results

I have attached screenshots of my test results. I created 4 test cases mytest1, mytest2,mytest3, and mytest4.

```
@Smarika12 → /workspaces/Malloc-Assignment (master) $ make
make: Nothing to be done for 'all'.
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-bf.so tests/bfwf
Worst fit should pick this one: 0x55bbce45e018
Best fit should pick this one: 0x55bbce46e0c4
Chosen address: 0x55bbce46e0c4

heap management statistics
mallocs:        7
frees:          2
reuses:         1
grows:          7
splits:         2
coalesces:      2
blocks:         7
requested:      8
max heap:       72636
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-wf.so tests/bfwf
Worst fit should pick this one: 0x557ef4a84018
Best fit should pick this one: 0x557ef4a940c4
Chosen address: 0x557ef4a84018

heap management statistics
mallocs:        7
frees:          2
reuses:         1
grows:          7
splits:         2
coalesces:      2
blocks:         7
requested:      8
max heap:       72636
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-nf.so tests/ffnf
First fit should pick this one: 0x563bec77d018
Next fit should pick this one: 0x563bec77ec58
Chosen address: 0x563bec77ec58

heap management statistics
mallocs:        12
frees:          3
reuses:         2
grows:          10
splits:         0
coalesces:      3
blocks:         10
requested:      12
max heap:       9064
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-ff.so tests/ffnf
First fit should pick this one: 0x55ff9ce07018
Next fit should pick this one: 0x55ff9ce08c58
Chosen address: 0x55ff9ce07018

heap management statistics
mallocs:        12
frees:          3
reuses:         2
grows:          10
splits:         0
coalesces:      3
blocks:         10
requested:      12
max heap:       9064
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-bf.so tests/bfwf
Worst fit should pick this one: 0x56017f2b5018
Best fit should pick this one: 0x56017f2c50c4
Chosen address: 0x56017f2c50c4

heap management statistics
mallocs:        7
frees:          2
reuses:         1
grows:          7
splits:         2
coalesces:      2
blocks:         7
requested:      8
max heap:       72636
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-wf.so tests/bfwf
Worst fit should pick this one: 0x55fc566a2018
Best fit should pick this one: 0x55fc566b20c4
Chosen address: 0x55fc566a2018

heap management statistics
mallocs:        7
frees:          2
reuses:         1
grows:          7
splits:         2
coalesces:      2
blocks:         7
requested:      8
max heap:       72636
@Smarika12 → /workspaces/Malloc-Assignment (master) $
```

```
grows:          7
splits:         2
coalesces:      2
blocks:         7
requested:      8
max heap:       21001032
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-wf.so tests/mytest2
Worst fit should pick this one: 0x56514c3e4018
Best fit should pick this one: 0x56514c3e4220
Chosen address: 0x56514c3e4018

heap management statistics
mallocs:        7
frees:          2
reuses:         1
grows:          6
splits:         0
coalesces:      2
blocks:         6
requested:      7
max heap:       1648
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-wf.so tests/mytest3

checking to see different splits, blocks, and max heaps when smaller space is malloced first

 No splits

heap management statistics
mallocs:        5
frees:          4
reuses:         1
grows:          4
splits:         0
coalesces:      4
blocks:         3
requested:      5
max heap:       1401224
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-wf.so tests/mytest4

checking to see different splits, blocks, and max heaps when larger space is malloced first

 Split occurs

heap management statistics
mallocs:        4
frees:          3
reuses:         2
grows:          4
splits:         4
coalesces:      3
blocks:         2
requested:      6
max heap:       601024
@Smarika12 → /workspaces/Malloc-Assignment (master) $
```


```
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-wf.so tests/mytest1
Worst fit should pick this one: 0x56079a0b7018
Best fit should pick this one: 0x56079ab34940
Chosen address: 0x56079a0b7018

heap management statistics
mallocs:        7
frees:          2
reuses:         1
grows:          7
splits:         2
coalesces:      2
blocks:         7
requested:      8
max heap:       21001032
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-wf.so tests/mytest2
Worst fit should pick this one: 0x56514c3e4018
Best fit should pick this one: 0x56514c3e4220
Chosen address: 0x56514c3e4018

heap management statistics
mallocs:        7
frees:          2
reuses:         1
grows:          6
splits:         0
coalesces:      2
blocks:         6
requested:      7
max heap:       1648
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-wf.so tests/mytest3

checking to see different splits, blocks, and max heaps when smaller space is malloced first

 No splits

heap management statistics
mallocs:        5
frees:          4
reuses:         1
grows:          4
splits:         0
coalesces:      4
blocks:         3
requested:      5
max heap:       1401224
@Smarika12 → /workspaces/Malloc-Assignment (master) $ env LD_PRELOAD=lib/libmalloc-wf.so tests/mytest4

checking to see different splits, blocks, and max heaps when larger space is malloced first

 Split occurs

heap management statistics
```

Explanation of Results

We can see that first fit and next fit are simpler. First fit and next fit is easy to understand has no splits. Also, the max heap is low for first fit and next fit. However, the best and worst fit takes more time and splits the blocks as well. As all of these have their own use, our requirement influences on which fit we use in our programs/life.

Conclusion

To conclude, I learnt ways to dynamically allocate memory from this assignment. We can create memory space according to our requirement using malloc and free it.