Name: Bibek Wagle
Studdent Id: 1001986216
Executive Summary:

The report provides an overview of the implementation and testing of memory allocation algorithms within the malloc.c file. Four different allocation algorithms, including First Fit, Best Fit, Worst Fit, and Next Fit, were implemented and tested alongside the traditional malloc function. The purpose of the report is to analyze and compare the performance of these algorithms based on various test cases.

Description of Algorithms:

1. First Fit: The first fit algorithm is the algorithm that searches for the first available free block in the heap that is large enough to accommodate the requested size.

2. Best Fit: The best Fit algorithm that algorithm that iterates through the free blocks and selects the block that is closest in size to the requested size without being smaller.

3. Worst Fit: The Worst Fit algorithm is the algorithm that selects the largest free block available in the heap that can accommodate the requested size.

4. Next Fit: The Next Fit algorithm is the algorithm that starts searching for free blocks from the last allocated block in the heap, wrapping around to the beginning if necessary.

Test Implementation:

The test implementation consists of various test cases (i.e Test1.c, Test2.c, Test3.c, Test4.c, bfwf.c,calloc.c, relloc.c, ffnf.c) to evaluate the behavior and performance of each allocation algorithm. Each test case involves requesting memory allocations of different sizes and patterns.

Test Results:

The test results for allcandidates:

For First fit:
env LD_PRELOAD=lib/libmalloc-ff.so tests/ffnf

First fit should pick this one: 0x5630dee5f018
Next fit should pick this one: 0x5630dee60c58

Chosen address: 0x5630dee5f018

heap management statistics
mallocs: 12
frees: 3
reuses: 12
grows: 10
splits: 0
coalesces: 0
blocks: 10
requested: 16064
max heap: 9064

For Next Fit:

env LD_PRELOAD=lib/libmalloc-nf.so tests/ffnf

First fit should pick this one: 0x55971325e018
Next fit should pick this one: 0x55971325fc58
Chosen address: 0x55971325fc58

heap management statistics
mallocs: 12
frees: 3
reuses: 12
grows: 10
splits: 0
coalesces: 0
blocks: 10
requested: 16064
max heap: 9064

For Worst Fit:

env LD_PRELOAD=lib/libmalloc-wf.so tests/bfwf
Worst fit should pick this one: 0x55a9687c7018
Best fit should pick this one: 0x55a9687d70c4
Chosen address: 0x55a9687c7018

heap management statistics
mallocs: 7
frees: 2
reuses: 7
grows: 6

splits: 1
coalesces: 0
blocks: 7
requested: 73636
max heap: 72636

For Best Fit:

env LD_PRELOAD=lib/libmalloc-bf.so tests/bfwf
Worst fit should pick this one: 0x556b6e2fe018
Best fit should pick this one: 0x556b6e30e0c4
Chosen address: 0x556b6e30e0c4

heap management statistics
mallocs: 7
frees: 2
reuses: 7
grows: 6
splits: 1
coalesces: 0
blocks: 7
requested: 73636
max heap: 72636

For realloc:
env LD_PRELOAD=lib/libmalloc-ff.so tests/realloc

realloc test PASSED

heap management statistics
mallocs: 3
frees: 1
reuses: 3
grows: 3
splits: 0
coalesces: 0
blocks: 3
requested: 1044
max heap: 1044

For calloc:

env LD_PRELOAD=lib/libmalloc-ff.so tests/calloc

calloc test PASSED

heap management statistics
mallocs: 2
frees: 1
reuses: 2
grows: 2
splits: 0
coalesces: 0
blocks: 2
requested: 1044
max heap: 1044

For simple malloc and free:

env LD_PRELOAD=lib/libmalloc-ff.so tests/test1
Running test 1 to test a simple malloc and free

heap management statistics
mallocs: 2
frees: 1
reuses: 2
grows: 2
splits: 0
coalesces: 0
blocks: 2
requested: 66560
max heap: 66560

To test coalesce:
env LD_PRELOAD=lib/libmalloc-ff.so tests/test3
Running test 3 to test coalesce

heap management statistics
mallocs: 4
frees: 3
reuses: 4
grows: 4
splits: 0
coalesces: 0
blocks: 4
requested: 5472
max heap: 5472

To test block, split and reuse:

env LD_PRELOAD=lib/libmalloc-ff.so tests/test4
Running test 4 to test a block split and reuse

heap management statistics
mallocs: 3
frees: 2
reuses: 3
grows: 2
splits: 1
coalesces: 1
blocks: 2
requested: 4096
max heap: 3072

Explanation and Interpretation of Results:

First Fit and Next Fit:
Both First Fit and Next Fit algorithms exhibit similar behavior in terms of memory allocation and heap management statistics.
They show a relatively high number of mallocs, indicating frequent allocation requests.
The number of frees is lower compared to mallocs, suggesting that a significant portion of allocated memory is still in use.
Reuses and grows indicate that the allocator efficiently manages memory by reusing freed blocks and growing the heap when needed.
The max heap size is sufficient to accommodate the requested memory.

Worst Fit and Best Fit:
Both Worst Fit and Best Fit algorithms also demonstrate similar performance in terms of memory allocation and heap management statistics.
They exhibit a lower number of mallocs compared to First Fit and Next Fit, indicating potentially better memory utilization.
The number of frees is proportionally lower, suggesting effective memory reuse and management.
The splits count is non-zero, indicating that the allocator performs block splitting when necessary to fulfill allocation requests.
Coalesces count remains zero, suggesting that there is no coalescing of adjacent free blocks.
The max heap size is sufficient to accommodate the requested memory, similar to First Fit and Next Fit.

realloc Test:
The realloc test passes, indicating that the realloc function behaves as expected without any errors.
The number of mallocs, frees, reuses, and grows are within expected ranges.
The max heap size is sufficient to accommodate the requested memory.

calloc Test:
The calloc test passes, indicating that the calloc function behaves as expected without any errors.
The number of mallocs, frees, reuses, and grows are within expected ranges.
The max heap size is sufficient to accommodate the requested memory.

Simple malloc and free Test:
The test for a simple malloc and free operation passes successfully.
The heap management statistics, including mallocs, frees, reuses, and grows, are consistent with the expected behavior.
The max heap size is sufficient to accommodate the requested memory.

Coalesce Test:
The test to verify coalescing functionality passes successfully.
The number of mallocs, frees, reuses, and grows are within expected ranges.
The coalesces count remains zero, indicating that there is no coalescing of adjacent free blocks.
The max heap size is sufficient to accommodate the requested memory.

Block Split and Reuse Test:
The test for block splitting and reuse passes successfully.
The allocator successfully splits blocks when necessary to fulfill allocation requests.
The coalesces count is non-zero, indicating that the allocator coalesces adjacent free blocks when possible.
The max heap size is sufficient to accommodate the requested memory.

Conclusion:
Overall, the implemented memory allocation algorithms, including First Fit, Next Fit, Worst Fit, and Best Fit, demonstrate effective memory management capabilities. Each algorithm has its strengths and weaknesses, and the choice of algorithm depends on the specific requirements of the application. The tests validate the functionality of memory allocation operations, including malloc, realloc, calloc, and free, without any critical issues or errors. Further optimization and fine-tuning may be required based on specific use cases and performance requirements.