

22bps1059

one

```
File Edit View Terminal Tabs Help
1 #include <stdio.h>
2 #include <stdlib.h>
3 #define MAX 5
4 void display(int** arr, int* mems) {
5     printf("current memory\n");
6     for(int i = 0; i<MAX; i++) {
7         printf("block %d:", i);
8         for(int j = 0; j<mems[i]; j++) {
9             printf("%d", *arr[i]);
10            arr[i]++;
11        }
12        printf("\n");
13    }
14    for(int j = 0; j<mems[i]; j++) {
15        *arr[i]--; // resetting the pointer
16    }
17 }
18
19 int worstfit(int** arr, int* mems, int request) {
20     printf("\nallocating %d, using worst fit\n", request);
21     int minindex = 0;
22     for(int i = 0; i<MAX; i++) {
23         printf("minindex is %d\n", minindex);
24         if(mems[i]<request) continue; // cause not possible
25         printf("%d, %d", mems[i]-request, mems[minindex]-request);
26         if(mems[i]> mems[minindex]) {
27             minindex = i;
28             printf("minindex is %d", minindex);
29         }
30     }
31 }
32
33 if(*arr[minindex] == 0) { // checking if it is unallocated
34     for(int j = 0; j<request; j++) {
35         *arr[minindex] = 1;
36         *arr[minindex]++;
37     }
38     for(int j = 0; j<request; j++) {
39         *arr[minindex]--; // resetting the pointer
40     }
41     return 0;
42 }
43
44
45 int bestfit(int** arr, int* mems, int request) {
46     printf("\nallocating %d, using best fit\n", request);
47     int perindex = 0;
48     for(int i = 0; i<MAX; i++) {
49         if(mems[i]<request){
50             perindex = i;
51             break;
52         }
53     }
54 }
55
56 }
57
58 }
59
60 }
61
62 }
63     return 0;
64 }
65 }
66
67
68 int firstfit(int** arr, int* mems, int request) {
69     printf("\nallocating %d, using first fit\n", request);
70     int perindex = 0;
71     for(int i = 0; i<MAX; i++) {
72         if(mems[i]>request){
73             perindex = i;
74             break;
75         }
76     }
77
78 if(*arr[perindex] == 0) { // checking if it is unallocated
79     for(int j = 0; j<request; j++) {
80         *arr[perindex] = 1;
81         *arr[perindex]++;
82     }
83     for(int j = 0; j<request; j++) {
84         *arr[perindex]--; // resetting the pointer
85     }
86     return 0;
87 }
88 }
89
90 int main() {
91     int *arr[MAX] = {};
92     int mems[MAX] = {};
93
94     // allocating memory
95     for(int i = 0; i<MAX; i++) {
96         int mem = rand() % 10 + 1;
97         mems[i] = mem;
98         printf("allocating %d mem\n", mem);
99         arr[i] = calloc(mem, sizeof(int));
100    }
101
102    // displaying memory
103    display(arr, mems);
104    worstfit(arr, mems, 5);
105    display(arr, mems);
106
107    display(arr, mems);
108    bestfit(arr, mems, 6);
109    display(arr, mems);
110
111    display(arr, mems);
112    firstfit(arr, mems, 4);
113    display(arr, mems);
114 }
115 }
```

current memory  
block 0:0000  
block 1:00000000  
block 2:00000000  
block 3:000000  
block 4:0000

allocating 5, using worst fit  
minindex is 0  
minindex is 0  
2, -1minindex is 1minindex is 1  
3, 2minindex is 2minindex is 2  
1, 3minindex is 2

current memory  
block 0:0000  
block 1:00000000  
block 2:11111000  
block 3:000000  
block 4:0000

current memory  
block 0:0000  
block 1:00000000  
block 2:11111000  
block 3:000000  
block 4:0000

allocating 6, using best fit

current memory  
block 0:0000  
block 1:00000000  
block 2:11111000  
block 3:111111  
block 4:0000

current memory  
block 0:0000  
block 1:00000000  
block 2:11111000  
block 3:111111  
block 4:0000

allocating 4, using first fit

current memory  
block 0:1111  
block 1:00000000  
block 2:11111000  
block 3:111111  
block 4:0000

one.c 1,1 Top one.c [R0] 62,1 Bot

0 0: bash\$ "moon" 13:27 01-Nov-23

two

```
File Edit View Terminal Tabs Help
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <stdbool.h>
4 #include <limits.h>
5
6 #define NUM_FRAMES 3 // Number of frames in memory
7
8 int frames[NUM_FRAMES]; // Memory frames
9 int referenceString[] = {7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2}; // Page reference string
10 int n = sizeof(referenceString) / sizeof(referenceString[0]);
11 int pageFaults = 0;
12 // Initialize memory frames to -1 (empty)
13 void initializeFrames() {
14     for (int i = 0; i < NUM_FRAMES; i++) {
15         frames[i] = -1;
16     }
17 }
18 // Display the memory frames
19 void displayFrames() {
20     printf("Frames: ");
21     for (int i = 0; i < NUM_FRAMES; i++) {
22         if (frames[i] == -1) {
23             printf("Empty ");
24         } else {
25             printf("%d ", frames[i]);
26         }
27     }
28     printf("\n");
29 }
30 // Check if a page is in memory frames
31 bool isPageInFrames(int page) {
32     for (int i = 0; i < NUM_FRAMES; i++) {
33         if (frames[i] == page) {
34             return true;
35         }
36     }
37     return false;
38 }
39
40 // FIFO Page Replacement Algorithm
41 void FIFO() {
42     initializeFrames();
43     int front = 0;
44
45     printf("FIFO Page Replacement:\n");
46     for (int i = 0; i < n; i++) {
47         int page = referenceString[i];
48         if (!isPageInFrames(page)) {
49             pageFaults++;
50             frames[front] = page;
51             front = (front + 1) % NUM_FRAMES;
52         }
53     }
54
55     if (frames[j] != -1) {
56         frames[j] = page;
57         break;
58     }
59     displayFrames();
60     printf("Page Faults: %d\n", pageFaults);
61 }
62
63 // OPTIMAL Page Replacement Algorithm
64 void OPTIMAL() {
65     initializeFrames();
66
67     printf("OPTIMAL Page Replacement:\n");
68     for (int i = 0; i < n; i++) {
69         int page = referenceString[i];
70         if (!isPageInFrames(page)) {
71             pageFaults++;
72             int farthest = -1;
73             int index = -1;
74             for (int j = 0; j < NUM_FRAMES; j++) {
75                 int k;
76                 for (k = i + 1; k < n; k++) {
77                     if (frames[j] == referenceString[k]) {
78                         if (k > farthest) {
79                             farthest = k;
80                             index = j;
81                         }
82                     }
83                     break;
84                 }
85                 if (k == n) {
86                     index = j;
87                     break;
88                 }
89             }
90             frames[index] = page;
91             displayFrames();
92             printf("Page Faults: %d\n", pageFaults);
93         }
94     }
95 }
96
97 int main() {
98     FIFO();
99     MRU();
100     pageFaults = 0;
101     OPTIMAL();
102 }
103
104 cardi~/vit/os/Lab10 ./two
FIFO Page Replacement:
Frames: 7 Empty Empty
Frames: 7 0 Empty
Frames: 7 0 1
Frames: 2 0 1
Frames: 2 0 1
Frames: 2 3 1
Frames: 2 3 0
Frames: 4 3 0
Frames: 4 2 0
Frames: 4 2 3
Frames: 0 2 3
Frames: 0 2 3
Page Faults: 10
MRU Page Replacement:
Frames: Empty Empty Empty
Frames: Empty Empty Empty
Frames: Empty Empty Empty
Frames: Empty Empty Empty
Frames: Empty Empty Empty
Frames: Empty Empty Empty
Frames: Empty Empty Empty
Frames: Empty Empty Empty
Frames: Empty Empty Empty
Frames: Empty Empty Empty
Page Faults: 11
OPTIMAL Page Replacement:
Frames: 7 Empty Empty
Frames: 0 Empty Empty
Frames: 0 1 Empty
Frames: 0 2 Empty
Frames: 0 2 Empty
Frames: 0 2 3
Frames: 0 2 3
Frames: 4 2 3
Frames: 4 2 3
Frames: 0 2 3
Frames: 0 2 3
Page Faults: 7
cardi~/vit/os/Lab10 scrot --focused fig2.png
two.c 30,1 Top two.c [R0] 71,1 95%
two.c" 122L, 3027B written
[0] 0:00bash "moon" 09:35 26-Oct-23
```