Ex-no: 8 M.Rohith
Date: 09-05-2023 3122 21 5001 085

SSN COLLEGE OF ENGINEERING, KALAVAKKAM
(An Autonomous Institution, Affiliated to Anna University, Chennai)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
II Year CSE – A & B Sections (IV Semester)

Academic Year 2022-23

UCS2412 - OPERATING SYSTEMS LAB

Lab Exercise 8 Implementation of Memory Management Algorithms

Scenario:

This problem can be implemented using linked list or arrays.

Free space is maintained as a linked list of nodes with each node having the starting byte address and the ending byte address of a free block. Each memory request consists of the process-id and the amount of storage space required in bytes. Allocated memory space is again maintained as a linked list of nodes with each node having the process id, starting byte address and the ending byte address of the allocated space. When a process finishes (taken as input), the appropriate node from the allocated list should be deleted and this free disk space should be added to the free space list. [Care should be taken to merge contiguous free blocks into one single block. This results in deleting more than one node from the free space list and changing the start and end address in the appropriate node]. For allocation use first fit, worst fit and best fit algorithms.

Steps:

1. Get the algorithm to work with. (First Fit, Best Fit, Worst Fit)

2. Implement this using Linked List / Arrays.

3. Let the options be 1. Entry / Allocate 2. Exit / Deallocate 3. Display (Memory allocated LL, Free Pool LL, Total Physical memory by Merging 2 LL) 4.Coalescing Holes.

4. Create a node using structure with the following details. Starting address, ending address, Size, Status (contains process id or H to denote that it is a hole).

5. Create a Allotted memory Linked list with a collection of nodes (Initially no nodes).

6. Create a Free Pool Linked List with a collection of nodes (Initially all partitions

will be in free pool LL).

7. Get the memory representation as input from user. No of partitions, starting address and ending address of each partition. Calculate the Size of each partition. Initially all the partitions will be in free pool LL.

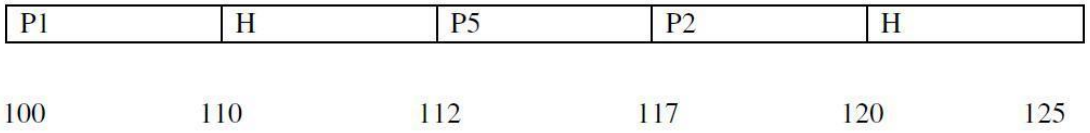
8. Now get whether a process enters / exits.

9. If ”Entry” then get the process id and its size in bytes. Using the algorithm allocate memory for it and store it in allotted memory linked list. Modify in Free pool Linked list also. [delete from free pool LL, insert into memory allocation LL]

10.If “exit” then get the process id and then deallocate the process from memory allocation linked list and include it in free pool linked list. [delete from memory allocation LL, insert into free pool LL]

11.On display you have to display both LL separately and combined memory structure should be displayed by merging both LL.

Merged Output should be



12.For Coalescing of Holes scan the free pool LL and check for continuous holes. If

so merge them as single hole by modifying ending address and size and delete the other hole.

13.For first fit select the first hole that satisfies the memory request.

14.For best fit select the hole that is large enough to accommodate the process but incurs minimum wastage of memory.

15.For worst fit select the hole that is the largest.

Sample Input and output:

Enter the Memory Representation:

Enter the no. of partitions in memory : 5

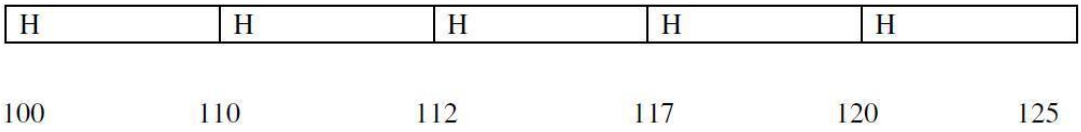
Starting and ending address of partition 1: 100 110

Starting and ending address of partition 2: 110 112

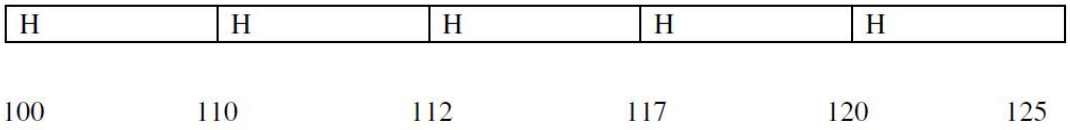
.....

Allotted Memory LL -----> Null

Free Pool >



Physical Memory--->



Memory Allocation Algorithm:

1.First Fit

2.Best Fit

3.Worst Fit

4. Exit from
program Enter choice :
1

First Fit Memory Allocation Algorithm

1.Entry / Allocate

2. Exit / Deallocate

3. Display

4. Coalescing of Holes

5. Back to Algorithm

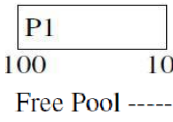
Enter choice : 1

Enter process id : P1

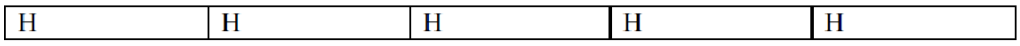
Enter size needed : 5

Memory is allotted to P1

Allotted Memory LL---- >

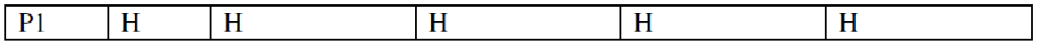


Free Pool ------>



105 110 112 117 120 125

Physical Memory------>



100 105 110 112 117 120 125

First Fit Memory Allocation Algorithm

1.Entry / Allocate

2. Exit / Deallocate

3. Display

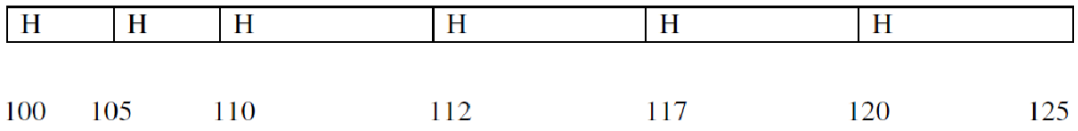
4. Coalescing of Holes

5. Back to
Algorithm
Enter choice : 2 Enter
process id : P1

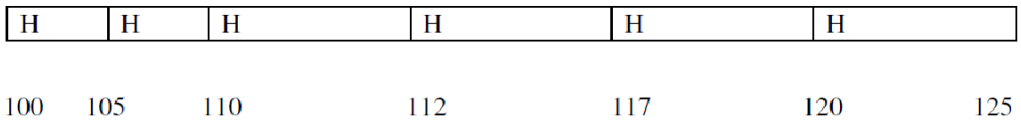
Memory is deallocated.

Allotted Memory LL ----> Null

Free Pool --



Physical Memory------>



M.Rohith
3122 21 5001 085

Program code:

Llinkedlist.h

```
typedef Partition *Data;

typedef struct Node
{
    Data d;
    struct Node *next;
} Node;

typedef Node *List;

List createEmptyList()
{
    Node *head = (Node *)malloc(sizeof(Node));
    head->d = NULL;
    head->next = NULL;
    return head;
}

void insert(List head, const Data d)
{
    Node *new = (Node *)malloc(sizeof(Node));
    new->d = d;

    Node *tmp = head;

    while (tmp->next != NULL)
    {
        if (tmp->next->d->start > d->start)
            break;
        tmp = tmp->next;
    }
    new->next = tmp->next;
    tmp->next = new;
}

Data delete (List prev)
{
    Data rVal=NULL;
    if (!prev)
```

M.Rohith
3122 21 5001 085

```
        return rVal;
    if (!prev->next)
        return rVal;

    Node *tmp = prev->next;
    rVal = tmp->d;
    prev->next = prev->next->next;
    free(tmp);

    return rVal;
}

void display(List head)
{
    Node *tmp = head->next;

    if (tmp == NULL)
    {
        printf(" Empty!\n");
        return;
    }

    int count = 0;
    while(tmp != NULL){
        tmp = tmp -> next;
        count++;
    }
    printf("\n  ");
    for(int i = 0; i < count; i++)
        printf("+----------+   ");
    printf("\n  ");

    tmp = head -> next;
    while (tmp != NULL)
    {
        printf("|   %-4s   |   ", printState(*(tmp->d)));
        tmp = tmp->next;
    }
    printf("\n  ");
    for(int i = 0; i < count; i++)
        printf("+----------+   ");
    printf("\n ");
```

M.Rohith
3122 21 5001 085

```
    tmp = head -> next;
    for(int i = 0; i < count; i++){
        printf("%-3d        %-3d ", tmp -> d -> start, tmp -> d -> end);
        tmp = tmp -> next;
    }

}
```

MemoryAllocation.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct Partition
{
    unsigned int start;
    unsigned int end;
    unsigned int size;
    int state;
} Partition;

char *const printState(const Partition P)
{
    static char str[5];
    if (P.state < -1)
        exit(1);
    else if (P.state == -1)
        strcpy(str, "Hole");
    else
    {
        str[0] = 'P';
        str[1] = (P.state / 10) + 48;
        str[2] = (P.state % 10) + 48;
        str[3] = ' ';
        str[4] = '\0';
    }
    return str;
}

#define HOLE -1
```

M.Rohith
3122 21 5001 085

```c
#include "LinkedList.h"

typedef enum Mode
{
    FirstFit = 1,
    BestFit,
    WorstFit
} Mode;

void FFAlloc(List, List, List, const int, const unsigned int);
void BFAlloc(List, List, List, const int, const unsigned int);
void WFAlloc(List, List, List, const int, const unsigned int);
void Dealloc(List, List, List, const int);
void Coalesce(List, List);

int main()
{
    int n, pid, choice = -1;
    unsigned int size;

    Mode m;

    Partition *tmp;

    List memory = createEmptyList();
    List free = createEmptyList();
    List allocated = createEmptyList();

    printf(" Enter the number of partitions: ");
    scanf("%d", &n);

    for (int i = 0; i < n; i++)
    {
        tmp = (Partition *)malloc(sizeof(Partition));
        printf(" Enter the start and end address: ");
        scanf("%d %d", &(tmp->start), &(tmp->end));
        tmp->size = tmp->end - tmp->start;
        tmp->state = HOLE;
        insert(memory, tmp);
        insert(free, tmp);
    }
```

M.Rohith
3122 21 5001 085

```
    while (1)
    {
        printf("\t\tMEMORY ALLOCATION TECHNIQUES\n");
        printf(" 1 - First Fit\n");
        printf(" 2 - Best Fit\n");
        printf(" 3 - Worst Fit\n");
        printf(" 0 - Exit\n");
        printf(" ------------------------\n");
        printf(" Enter your choice: ");
        scanf("%d", &m);

        if (m < 0 || m > 3)
        {
            printf("Invalid Mode!\n");
            continue;
        }

        if (m == 0)
            return 0;

        while (1)
        {
            printf("\n\n");
            printf("\t\t\t\tOPTIONS\n");
            printf(" 1 - Entry / Allocate\n");
            printf(" 2 - Exit / De-Allocate\n");
            printf(" 3 - Display\n");
            printf(" 4 - Coalescing of Holes\n");
            printf(" 5 - Back\n");
            printf(" ------------------------\n");
            printf(" Enter your choice: ");
            scanf("%d", &choice);

            if (choice < 1 || choice > 5)
            {
                printf(" Invalid Input\n");
                continue;
            }

            if (choice == 5)
                break;
```

M.Rohith
3122 21 5001 085

```
            switch (choice)
            {
            case 1:
                printf("\n Enter the PID of process: ");
                scanf("%d", &pid);
                printf(" Enter the size required: ");
                scanf("%d", &size);

                switch (m)
                {
                case FirstFit:
                    FFAlloc(memory, free, allocated, pid, size);
                    break;
                case BestFit:
                    BFAlloc(memory, free, allocated, pid, size);
                    break;
                case WorstFit:
                    WFAlloc(memory, free, allocated, pid, size);
                default:
                    break;
                }
                break;
            case 2:
                printf(" Enter PID of process to exit: ");
                scanf("%d", &pid);
                Dealloc(memory, free, allocated, pid);
                break;
            case 3:
                printf(" ALLOCATED PARTITIONS:\n");
                display(allocated);
                printf("\n FREE PARTITIONS:\n");
                display(free);
                printf("\n ALL PARTITIONS:\n");
                display(memory);
                break;
            case 4:
                Coalesce(memory, free);
            default:
                break;
            }
        }
    }
```

M.Rohith
3122 21 5001 085

```
}

void FFAlloc(List memory, List free, List alloc, const int pid, const unsigned int
size)
{
    Partition *fragment;

    if (free->next == NULL)
    {
        printf(" No Free Space Available!\n");
        return;
    }

    int flag = 0;
    unsigned int total_size;

    Partition *p;

    List tmp = free;
    while (tmp->next != NULL)
    {
        if (tmp->next->d->state != HOLE)
        {
            tmp = tmp->next;
            continue;
        }
        if (tmp->next->d->size >= size)
        {
            flag = 1;
            if (tmp->next->d->size == size)
            {
                p = delete (tmp);
                p->state = pid;
                insert(alloc, p);
                break;
            }
            else
            {
                p = delete (tmp);
                fragment = (Partition *)malloc(sizeof(Partition));
                fragment->end = p->end;
                fragment->start = p->start + size;
```

M.Rohith
3122 21 5001 085

```
                fragment->state = HOLE;
                fragment->size = fragment->end - fragment->start;

                p->end = p->start + size;
                p->state = pid;
                p->size = size;

                insert(memory, fragment);
                insert(free, fragment);
                insert(alloc, p);
                break;
            }
        }
        tmp = tmp->next;
    }
    if (!flag)
        printf(" Unable to Allocate Required Memory!\n");
    else
        printf(" Successfully Allocated!\n");
}

void BFAlloc(List memory, List free, List alloc, const int pid, const unsigned int
size)
{

    if (free->next == NULL)
    {
        printf(" No Free Space Available!\n");
        return;
    }

    unsigned int left_over = 999;

    Node *ptr = NULL;

    Partition *p, *fragment;

    List tmp = free;
    while (tmp->next != NULL)
    {
        if (tmp->next->d->state != HOLE)
        {
```

M.Rohith
3122 21 5001 085

```
            tmp = tmp->next;
            continue;
        }
        if (tmp->next->d->size >= size)
            if (tmp->next->d->size - size < left_over)
            {
                left_over = tmp->next->d->size - size;
                ptr = tmp;
            }
        tmp = tmp->next;
    }

    if (!ptr)
    {
        printf(" Unable to allocate required memory!\n");
        return;
    }

    p = delete (ptr);
    p->state = pid;
    p->size = size;
    if (left_over == 0)
        insert(alloc, p);
    else
    {
        fragment = (Partition *)malloc(sizeof(Partition));
        fragment->start = p->start + size;
        fragment->end = p->end;
        fragment->state = HOLE;
        fragment->size = fragment->end - fragment->start;
        p->end = p->start + size;
        insert(alloc, p);
        insert(memory, fragment);
        insert(free, fragment);
    }

    printf(" Successfully Allocated Memory!\n");
}

void WFAlloc(List memory, List free, List alloc, const int pid, const unsigned int
size)
{
```

M.Rohith
3122 21 5001 085

```
if (free->next == NULL)
{
    printf(" No Free Space Available!\n");
    return;
}

unsigned int left_over = 0;

Node *ptr = NULL;

Partition *p, *fragment;

List tmp = free;
while (tmp->next != NULL)
{
    if (tmp->next->d->state != HOLE)
    {
        tmp = tmp->next;
        continue;
    }
    if (tmp->next->d->size >= size)
        if (tmp->next->d->size - size > left_over)
        {
            left_over = tmp->next->d->size - size;
            ptr = tmp;
        }
    tmp = tmp->next;
}

if (!ptr)
{
    printf(" Unable to allocate required memory!\n");
    return;
}

p = delete (ptr);
p->state = pid;
p->size = size;

if (left_over == 0)
    insert(alloc, p);
```

M.Rohith
3122 21 5001 085

```
    else
    {
        fragment = (Partition *)malloc(sizeof(Partition));
        fragment->start = p->start + size;
        fragment->end = p->end;
        fragment->state = HOLE;
        fragment->size = fragment->end - fragment->start;
        p->end = p->start + size;
        insert(alloc, p);
        insert(memory, fragment);
        insert(free, fragment);
    }

    printf(" Successfully Allocated Memory!\n");
}

void Dealloc(List memory, List free, List alloc, const int pid)
{
    if (alloc->next == NULL)
    {
        printf(" No Process Allocated!\n");
        return;
    }
    Partition *p;
    Node *tmp = alloc;
    int flag = 0;

    while (tmp->next != NULL)
    {
        if (tmp->next->d->state == pid)
        {
            flag = 1;
            break;
        }
        tmp = tmp->next;
    }

    if (flag == 0)
    {
        printf(" No such Process Found!\n");
        return;
    }
```

```
    p = delete (tmp);
    p->state = HOLE;
    insert(free, p);

    printf(" Successfully De-Allocated Memory\n");
}

void Coalesce(List memory, List free)
{
    if (!free->next)
        return;
    if (!free->next->next)
        return;
    Node *l = NULL,
         *r = NULL;
    Partition *left = NULL,
              *right = NULL,
              *p = NULL;

    Node *tmp = free, *tmp2 = memory;

    while (tmp->next != NULL && tmp->next->next != NULL)
    {
        if (tmp->next->d->end == tmp->next->next->d->start)
        {
            l = tmp;
            left = tmp->next->d;
            r = tmp->next;
            right = tmp->next->next->d;
            p = (Partition *)malloc(sizeof(Partition));
            p->start = left->start;
            p->end = right->end;
            p->size = p->end - p->start;
            p->state = HOLE;
            delete (r);
            delete (l);
            insert(free, p);
            while (tmp2->next != NULL && tmp2->next->next != NULL)
            {
                if (tmp2->next->d == left)
                {
```

M.Rohith
3122 21 5001 085

```
                    l = tmp2;
                    r = tmp2->next;
                    delete (r);
                    delete (l);
                    insert(memory, p);
                    break;
                }
                tmp2 = tmp2->next;
            }
        }
        tmp = tmp->next;
    }
}
```

M.Rohith
3122 21 5001 085

Output:

```
rohith@Rohith: ~/Desktop/OS
rohith@Rohith:~/Desktop/OSlab/Assignment-8$ gcc MemoryAllocation.c -o run
MemoryAllocation.c: In function 'main':
MemoryAllocation.c:83:17: warning: format '%d' expects argument of type 'int *', but argument 2 has type 'Mode *' [-Wformat=]
   83 |         scanf("%d", &m);
      |                ~^   ~~
      |                 |   |
      |                 |   Mode *
      |                 int *
rohith@Rohith:~/Desktop/OSlab/Assignment-8$ ./run
 Enter the number of partitions: 5
 Enter the start and end address: 100 150
 Enter the start and end address: 160 170
 Enter the start and end address: 200 250
 Enter the start and end address: 275 300
 Enter the start and end address: 350 450
                MEMORY ALLOCATION TECHNIQUES
 1 - First Fit
 2 - Best Fit
 3 - Worst Fit
 0 - Exit
 ------------------------
 Enter your choice: 1


                                OPTIONS
 1 - Entry / Allocate
 2 - Exit / De-Allocate
 3 - Display
 4 - Coalescing of Holes
 5 - Back
 ------------------------
 Enter your choice: 1

 Enter the PID of process: 1
 Enter the size required: 10
 Successfully Allocated!


                                OPTIONS
 1 - Entry / Allocate
 2 - Exit / De-Allocate
```

```
rohith@Rohith: ~/Desktop/OS
 3 - Display
 4 - Coalescing of Holes
 5 - Back
 ------------------------
 Enter your choice: 3
 ALLOCATED PARTITIONS:

  +----------+
  |   P01    |
  +----------+
 100        110
 FREE PARTITIONS:

  +----------+   +----------+   +----------+   +----------+   +----------+
  |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
  +----------+   +----------+   +----------+   +----------+   +----------+
 110        150 160        170 200        250 275        300 350        450
 ALL PARTITIONS:

  +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
  |   P01    |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
  +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
 100        110 110        150 160        170 200        250 275        300 350        450

                                OPTIONS
 1 - Entry / Allocate
 2 - Exit / De-Allocate
 3 - Display
 4 - Coalescing of Holes
 5 - Back
 ------------------------
 Enter your choice: 1

 Enter the PID of process: 2
 Enter the size required: 25
 Successfully Allocated!


                                OPTIONS
 1 - Entry / Allocate
 2 - Exit / De-Allocate
```

Ex-no: 8 M.Rohith
Date: 09-05-2023 3122 21 5001 085

```
rohith@Rohith: ~/Desktop/O:
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 3
ALLOCATED PARTITIONS:

 +----------+   +----------+
 |   P01    |   |   P02    |
 +----------+   +----------+
100        110 110        135
FREE PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+
135        150 160        170 200        250 275        300 350        450
ALL PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
 |   P01    |   |   P02    |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
100        110 110        135 135        150 160        170 200        250 275        300 350        450

                            OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 2
Enter PID of process to exit: 2
Successfully De-Allocated Memory


                            OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
```

```
rohith@Rohith: ~/Desktop/O:
5 - Back
------------------------
Enter your choice: 4


                            OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 3
ALLOCATED PARTITIONS:

 +----------+
 |   P01    |
 +----------+
100        110
FREE PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+
110        150 160        170 200        250 275        300 350        450
ALL PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
 |   P01    |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
100        110 110        150 160        170 200        250 275        300 350        450

                            OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 2
Enter PID of process to exit: 1
```

Ex-no: 8 M.Rohith
Date: 09-05-2023 3122 21 5001 085

```
rohith@Rohith: ~/Desktop/O!
Successfully De-Allocated Memory


                                OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 4


                                OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 3
ALLOCATED PARTITIONS:
Empty!

FREE PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        450
ALL PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        450

                                OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
```

```
rohith@Rohith: ~/Desktop/O!
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 5
              MEMORY ALLOCATION TECHNIQUES
1 - First Fit
2 - Best Fit
3 - Worst Fit
0 - Exit
------------------------
Enter your choice: 2


                                OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 1

Enter the PID of process: 3
Enter the size required: 10
Successfully Allocated Memory!


                                OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 3
ALLOCATED PARTITIONS:

 +----------+
 |   P03    |
 +----------+
160        170
```

Ex-no: 8
Date: 09-05-2023

M.Rohith
3122 21 5001 085

```
rohith@Rohith: ~/Desktop/O:
FREE PARTITIONS:

 +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+
100        150 200        250 275        300 350        450
ALL PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   P03    |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        450

                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 1

Enter the PID of process: 4
Enter the size required: 25
Successfully Allocated Memory!


                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 3
ALLOCATED PARTITIONS:

 +----------+   +----------+
 |   P03    |   |   P04    |
 +----------+   +----------+
160        170 275        300
```

```
rohith@Rohith: ~/Desktop/O:
FREE PARTITIONS:

 +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+
100        150 200        250 350        450
ALL PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   P03    |   |   Hole   |   |   P04    |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        450

                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 2
Enter PID of process to exit: 3
Successfully De-Allocated Memory


                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 3
ALLOCATED PARTITIONS:

 +----------+
 |   P04    |
 +----------+
275        300
FREE PARTITIONS:
```

Ex-no: 8 M.Rohith
Date: 09-05-2023 3122 21 5001 085

```
rohith@Rohith: ~/Desktop/O:

 +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 350        450
ALL PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   P04    |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        450

                            OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 2
Enter PID of process to exit: 4
Successfully De-Allocated Memory


                            OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 3
ALLOCATED PARTITIONS:
Empty!

FREE PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        450
```

```
rohith@Rohith: ~/Desktop/O:

ALL PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        450

                            OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 5
              MEMORY ALLOCATION TECHNIQUES
1 - First Fit
2 - Best Fit
3 - Worst Fit
0 - Exit
------------------------
Enter your choice: 3


                            OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 1

Enter the PID of process: 5
Enter the size required: 10
Successfully Allocated Memory!


                            OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
```

Ex-no: 8 M.Rohith
Date: 09-05-2023 3122 21 5001 085

```
rohith@Rohith: ~/Desktop/O!
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 3
ALLOCATED PARTITIONS:

 +----------+
 |   P05    |
 +----------+
350        360
FREE PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 360        450
ALL PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   P05    |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        360 360        450

                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 1

Enter the PID of process: 6
Enter the size required: 25
Successfully Allocated Memory!


                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
```

```
rohith@Rohith: ~/Desktop/O!
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 3
ALLOCATED PARTITIONS:

 +----------+   +----------+
 |   P05    |   |   P06    |
 +----------+   +----------+
350        360 360        385
FREE PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 385        450
ALL PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   P05    |   |   P06    |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        360 360        385 385        450

                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 2
Enter PID of process to exit: 5
Successfully De-Allocated Memory


                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
```

Ex-no: 8
Date: 09-05-2023

M.Rohith
3122 21 5001 085

```
rohith@Rohith: ~/Desktop/O:
5 - Back
------------------------
Enter your choice: 4


                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 2
Enter PID of process to exit: 6
Successfully De-Allocated Memory


                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 3
ALLOCATED PARTITIONS:
Empty!

FREE PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        360 360        385 385        450
ALL PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        360 360        385 385        450
```

```
rohith@Rohith: ~/Desktop/O:
                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 4


                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
Enter your choice: 3
ALLOCATED PARTITIONS:
Empty!

FREE PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        385 385        450
ALL PARTITIONS:

 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
 |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |   |   Hole   |
 +----------+   +----------+   +----------+   +----------+   +----------+   +----------+
100        150 160        170 200        250 275        300 350        385 385        450

                              OPTIONS
1 - Entry / Allocate
2 - Exit / De-Allocate
3 - Display
4 - Coalescing of Holes
5 - Back
------------------------
```

```
Enter your choice: 5
                MEMORY ALLOCATION TECHNIQUES
1 - First Fit
2 - Best Fit
3 - Worst Fit
0 - Exit
------------------------
Enter your choice: 0
rohith@Rohith:~/Desktop/OSlab/Assignment-8$ ^C
rohith@Rohith:~/Desktop/OSlab/Assignment-8$
```

Learning outcomes:

Through the implementation of above programs, the solution and output has been obtained for memory management techniques.