A.Y. 2022-2023

PROCESSOR ORGANIZATION AND ARCHITECTURE

AYUSH JAIN

COMPUTER ENGINEERING | TE – B2 | 60004200132

EXPERIMENT – 3

AIM: Implement Sequential memory organization and Printing Location of Desired Word along with Average Memory Access Time (AMAT)

CODE:

1. FIRST FIT:

```c
1.  #include <stdio.h> int main()
2.  {
3.  int totalMem = 0;
4.  int part[] = {200, 400, 600, 500, 300, 250};
5.  int i, j;
6.  printf("\nEnter number of process to be added to main memory:");
7.  int n ;
8.  scanf("%d", &n);
9.  int mem_p[n];
10. int flag[6];
11. for(i = 0; i < n; i++)
12. {
13.  printf("Enter memory to be assigned to process %d : ",(i+1)); scanf("%d", &mem_p[i]);
14. }
15. for(i = 0; i < 6; i++)
16.  flag[i] = 0;
17. int id;
18. for(i = 0; i < n; i++)
19. {
20. id = -1;
21. for(j = 0; j < 6; j++)
22. {
23. if((flag[j] == 0) && (mem_p[i] <= part[j]))
24. {
25. id = j;
26. break;
27. }
```

```
28. }
29. if(id != -1)
30. {
31. printf("\nProcess %d\tMemory Allocated\tPartition: %d ",(i+1),part[id]);
32. flag[id] = 1;
33. }
34. else printf("\nProcess %d\tMemory Not Allocated", (i+1));
35. }
36. return 0;
37. }
```

OUTPUT:

```
Enter number of process to be added to main memory:4
Enter memory to be assigned to process 1 : 100
Enter memory to be assigned to process 2 : 400
Enter memory to be assigned to process 3 : 200
Enter memory to be assigned to process 4 : 300

Process 1        Memory Allocated        Partition: 200
Process 2        Memory Allocated        Partition: 400
Process 3        Memory Allocated        Partition: 600
Process 4        Memory Allocated        Partition: 500

...Program finished with exit code 0
Press ENTER to exit console.
```

2. BEST FIT:

```c
#include <stdio.h> int main()
{
int totalMem = 0;
int part[] = {200, 400, 600, 500, 300, 250};
int i, j;
printf("\nEnter number of process to be added to main memory: ");
int n ;
scanf("%d", &n);
int mem_p[n];
int flag[6]; for(i = 0; i < n; i++)
{
printf("Enter memory to be assigned to process %d : ",(i+1));
scanf("%d", &mem_p[i]);
}
for(i = 0; i < 6; i++)
  flag[i] = 0;
int diff = 10000, id;
for(i = 0; i < n; i++)
{
id = -1;
for(j = 0; j < 6; j++)
{
if((flag[j] == 0) && (mem_p[i] <= part[j]) && (part[j] - mem_p[i] < diff))
{
diff = part[j] - mem_p[i];
id = j;
}
}
if(id != -1)
{
printf("\nProcess %d\tMemory Allocated\tPartition: %d ",(i+1),part[id]);
flag[id] = 1;
}
else printf("\nProcess %d\tMemory Not Allocated", (i+1));
diff = 10000;
}
return 0;
}
```

OUTPUT:

```
Enter number of process to be added to main memory: 4
Enter memory to be assigned to process 1 : 100
Enter memory to be assigned to process 2 : 300
Enter memory to be assigned to process 3 : 200
Enter memory to be assigned to process 4 : 400

Process 1        Memory Allocated         Partition: 200
Process 2        Memory Allocated         Partition: 300
Process 3        Memory Allocated         Partition: 250
Process 4        Memory Allocated         Partition: 400

...Program finished with exit code 0
Press ENTER to exit console.
```

3. NEXT FIT:

```c
1.  #include <stdio.h> int main()
2.  {
3.  int part[] = {200, 400, 600, 500, 300, 250};
4.  int i, j;
5.  printf("\nEnter number of process to be added to main memory: ");
6.  int n ;
7.  scanf("%d", &n); int mem_p[n];
8.  int flag[6]; for(i = 0; i < n; i++)
9.  {
10. printf("Enter memory to be assigned to process %d : ",(i+1)); scanf("%d", &mem_p[i]);
11. }
12. for(i = 0; i < 6; i++) flag[i] = 0; int id, prevId = 0; for(i = 0; i < n; i++)
13. {
14. id = -1; for(j = prevId; j < 6; j++)
15. {
16. if((flag[j] == 0) && (mem_p[i] <= part[j]))
17. {
18. id = j;
19. break;
20. }
21. }
22. if(id != -1)
23. {
24. printf("\nProcess %d\tMemory Allocated\tPartition: %d ",(i+1),part[id]);
25. flag[id] = 1; prevId = id;
26. }
27. else printf("\nProcess %d\tMemory Not Allocated", (i+1));
28. }
29. return 0;
30. }
```

OUTPUT:

```
Enter number of process to be added to main memory: 4
Enter memory to be assigned to process 1 : 100
Enter memory to be assigned to process 2 : 400
Enter memory to be assigned to process 3 : 200
Enter memory to be assigned to process 4 : 300

Process 1        Memory Allocated        Partition: 200
Process 2        Memory Allocated        Partition: 400
Process 3        Memory Allocated        Partition: 600
Process 4        Memory Allocated        Partition: 500

...Program finished with exit code 0
Press ENTER to exit console.
```

4. WORST FIT:

```c
1.  #include <stdio.h> int main()
2.  {
3.  int totalMem = 0;
4.  int part[] = {200, 400, 600, 500, 300, 250};
5.  int i, j;
6.  printf("Enter number of process to be added to main memory: ");
7.  int n ;
8.  scanf("%d", &n);
9.  int mem_p[n];
10. for(i = 0; i < n; i++)
11. {
12. printf("Enter memory to be assigned to process %d : ",(i+1)); scanf("%d", &mem_p[i]);
13. }
14. int diff = 0, id;
15. for(i = 0; i < n; i++)
16. {
17. id = -1; for(j = 0; j < 6; j++)
18. {
19. if((mem_p[i] <= part[j]) && (part[j] - mem_p[i] > diff))
20. {
21. diff = part[j] - mem_p[i];
22. id = j;
23. }
24. }
25. if(id != -1)
26. {
27. printf("\nProcess %d\tMemory Allocated\tPartition: %d ",(i+1),part[id]); part[id] =
    part[id] - mem_p[i];
28. }
```

```
29. else printf("\nProcess %d\tMemory Not Allocated", (i+1));
30. diff = 0;
31.
32. }
33. return 0;
34. }
```

OUTPUT:

```
Enter number of process to be added to main memory: 4
Enter memory to be assigned to process 1 : 100
Enter memory to be assigned to process 2 : 400
Enter memory to be assigned to process 3 : 200
Enter memory to be assigned to process 4 : 300

Process 1        Memory Allocated         Partition: 600
Process 2        Memory Allocated         Partition: 500
Process 3        Memory Allocated         Partition: 500
Process 4        Memory Allocated         Partition: 400

...Program finished with exit code 0
Press ENTER to exit console.
```

CONCLUSION: Done implementing Sequential memory organization and Printing Location of Desired Word along with Average Memory Access Time (AMAT)