

## CERTIFICATE

Name of the Lab : OPERATING SYSTEMS

Name of the Student :

Student Regd. No. : 18BQ1A05K3

CLASS : III B.TECH. I SEM CSE – D

GIT HUB LINK: [Click to view my repository](#)

## INDEX

S. No.	Name of the Experiment	Page No.
3 (a)	Simulate Multiprogramming with a fixed number of tasks (MFT)	
3 (b)	Simulate Multiprogramming with a variable number of tasks (MVT)	

### **EXPERIMENT NO: 3 (a)**

**AIM :** Simulate Multiprogramming with a fixed number of tasks (MFT)

**DESCRIPTION :** Main task of this program is to allocate the incoming processes to the suitable holes into the physical memory, Where physical memory(RAM) is splitted into fixed number of partitions, each one with variable size. The incoming number of processes or tasks are fixed by default.

**LIBRARIES USED:** Language : python 3, Systems having python version  $\geq 3$  will produce output as expected. No additional libraries have been used.

#### **PROGRAM-1:**

```
def firstfit(partitions,processes):  
  
    result = [-1]*len(partitions)  
  
    internalfrag = [-1]*len(partitions)  
  
    for i in range(0,len(processes)):  
        for j in range(0,len(partitions)):  
            if processes[i] <= partitions[j] and result[j] == -1:  
                result[j] = i+1  
                internalfrag[j] = partitions[j] - processes[i]  
                break  
  
    fitname = "First fit"  
  
    output(partitions,processes,result,internalfrag,fitname)
```

```
def bestfit(partitions,processes):  
  
    result = [-1]*len(partitions)  
  
    internalfrag = [-1]*len(partitions)  
  
    for i in range(0,len(processes)):  
  
        mini = sum(partitions)  
  
        minindex = -1  
  
        flagbit = False  
  
        for j in range(0,len(partitions)):  
  
            if processes[i] <= partitions[j] and result[j] == -1 and  
partitions[j]-processes[i]<=mini:  
  
                minindex = j  
  
                mini = partitions[j] - processes[i]  
  
                flagbit = True  
  
            if flagbit and minindex!=1:  
  
                result[minindex] = i+1  
  
                internalfrag[minindex] = mini  
  
    fitname = "Best Fit"  
  
    output(partitions,processes,result,internalfrag,fitname)  
  
def worstfit(partitions,processes):  
  
    result = [-1]*len(partitions)  
  
    internalfrag = [-1]*len(partitions)
```

```
for i in range(0, len(processes)):

    maxi = -1

    maxindex = -1

    flagbit = False

    for j in range(0, len(partitions)):

        if processes[i] <= partitions[j] and result[j] == -1 and
partitions[j]-processes[i]>=maxi:

            maxindex = j

            maxi = partitions[j] - processes[i]

            flagbit = True

        if maxindex != -1 and flagbit:

            result[maxindex] = i+1

            internalfrag[maxindex] = maxi

        else:

            continue

    fitname = "Worst Fit"

    output(partitions, processes, result, internalfrag, fitname)

def output(partitions, processes, result, internalfrag, fitname):

    print("\n")

    print("-----"+fitname+"-----")

    print("Partitions: ".ljust(23), partitions)
```

```
print("Processes: ".ljust(23),processes)

print("Allocation: ".ljust(23),result)

print("Internal Fragmentation:".ljust(23),internalfrag)

#unallocated processes

print("-----Unallocated processes-----")

isunalloc = False

for i in range(len(processes)):

    if i+1 not in result:

        isunalloc = True

        print("Process "+str(i+1)+" cannot be allocated")

if isunalloc == False:

    print("All processes have been allocated successfully")

print("\n")


#execution begins here.

phymem = int(input("Enter the size of physical memory : "))

partitions = []

no_partitions = int(input("Enter number of partitions : "))

for i in range(0,no_partitions):

    partitions.append(int(input("Enter partition"+str(i+1)+" size: ")))
```

```
if sum(partitions) == phymem:
    processes = []
    no_processes = int(input("Enter number of processes incoming :"))
    for j in range(0,no_processes):
        processes.append(int(input("Enter process"+str(j+1)+" size: ")))
    #input has been taken
    firstfit(partitions,processes)
    bestfit(partitions,processes)
    worstfit(partitions,processes)
```

**ASK USER THE SIZE OF PHYSICAL MEMORY FOR USER PROCESSES:**

**ASK USER SIZE OF PARTITION1:**

**ASK USER SIZE OF PARTITION1:**

**NOTE: SUM OF SIZE OF ALL PARTITIONS = SIZE OF PHYSICAL MEMORY**

**ASK USER SIZE OF PROCESS1:**

**ALLOCATE PARTITION WHERE THIS PROCESS CAN FIT**

**DO YOU WANT TO CONTINUE: Y/N**

**IF YES**

**ASK USER SIZE OF PROCESS2:**

**DO YOU WANT TO CONTINUE: Y/N**

**N**

PARTITION NUMBER	PARTITION SIZE (MB)
1	100

2	80
3	120
4	200

PROCESS NUMBER	PROCESS SIZE (MB)
1	70
2	90
3	110
4	150

**OUTPUT:** \* it should be same when your program gets executed

#### BEST FIT ALGORITHM

PARTITION NUMBER	PARTITION SIZE (MB)	PROCESS NUMBER	PROCESS SIZE (MB)	INTERNAL FRAGMENTATI ON
1	100	2	90	10
2	80	1	70	10
3	120	3	110	10
4	200	4	150	50

#### FIRST FIT ALGORITHM

PARTITION NUMBER	PARTITION SIZE (MB)	PROCESS NUMBER	PROCESS SIZE (MB)	INTERNAL FRAGMENTATI ON



1	100	1	70	30
2	80			
3	120	2	90	30
4	200	3	110	90

PROCESS 4 CAN NOT FIT AS THERE IS NO FREE SPACE.

**WORST FIT ALGORITHM**

PARTITION NUMBER	PARTITION SIZE (MB)	PROCESS NUMBER	PROCESS SIZE (MB)	INTERNAL FRAGMENTATI ON
1	100			
2	80			
3	120	2	90	30
4	200	1	70	130

**OUTPUT SCREEN SHOTS:**

```
nag-1211@hp-laptop:~/Documents/3-1/OS Lab$ python3 MultitaskingWithFixedPartitions.py
Enter the size of physical memory : 500
Enter number of partitions : 4
Enter partition1 size: 100
Enter partition2 size: 80
Enter partition3 size: 120
Enter partition4 size: 200
Enter number of processes incoming : 4
Enter process1 size: 70
Enter process2 size: 90
Enter process3 size: 110
Enter process4 size: 150

-----First fit-----
Partitions:      [100, 80, 120, 200]
Processes:       [70, 90, 110, 150]
Allocation:      [1, -1, 2, 3]
Internal Fragmentation: [30, -1, 30, 90]
-----Unallocated processes-----
Process 4 cannot be allocated

-----Best Fit-----
Partitions:      [100, 80, 120, 200]
Processes:       [70, 90, 110, 150]
Allocation:      [2, -1, 3, 4]
Internal Fragmentation: [10, -1, 10, 50]
-----Unallocated processes-----
Process 1 cannot be allocated

-----Worst Fit-----
Partitions:      [100, 80, 120, 200]
Processes:       [70, 90, 110, 150]
Allocation:      [-1, -1, 2, 1]
Internal Fragmentation: [-1, -1, 30, 130]
```

```
Enter partition1 size: 100
Enter partition2 size: 80
Enter partition3 size: 120
Enter partition4 size: 200
Enter number of processes incoming :4
Enter process1 size: 70
Enter process2 size: 90
Enter process3 size: 110
Enter process4 size: 150

-----First fit-----
Partitions:          [100, 80, 120, 200]
Processes:           [70, 90, 110, 150]
Allocation:          [1, -1, 2, 3]
Internal Fragmentation: [30, -1, 30, 90]
-----Unallocated processes-----
Process 4 cannot be allocated

-----Best Fit-----
Partitions:          [100, 80, 120, 200]
Processes:           [70, 90, 110, 150]
Allocation:          [2, -1, 3, 4]
Internal Fragmentation: [10, -1, 10, 50]
-----Unallocated processes-----
Process 1 cannot be allocated

-----Worst Fit-----
Partitions:          [100, 80, 120, 200]
Processes:           [70, 90, 110, 150]
Allocation:          [-1, -1, 2, 1]
Internal Fragmentation: [-1, -1, 30, 130]
-----Unallocated processes-----
Process 3 cannot be allocated
Process 4 cannot be allocated
```

### **EXPERIMENT NO: 3 (B)**

**AIM :** Simulate Multiprogramming with a variable number of tasks (MVT)

**DESCRIPTION :** Main task of this program is to allocate the incoming processes to the suitable holes into the physical memory. The incoming number of processes or tasks are not fixed by default, it is the choice of user to give any number of tasks.

**LIBRARIES USED:** Language : python 3, Systems having python version  $\geq 3$  will produce output as expected. No additional libraries have been used.

#### **PROGRAM-1:**

```
phymem = int(input("Enter the size of Physical memory: "))
Again = True
process_number = 0
processes = {}
unallocated = {}
cumulativesum = 0
while Again:
    Again = False
    process_number+=1
    process_name = input("\nEnter the name of the process: ")
    process_size = int(input("Enter the size of the process: "))
```

```
if cumulativesum + process_size <= phymem:

    cumulativesum += process_size

    processes[process_name] = [process_number, process_size]

    confirm = input("Is there process incoming?[Y/N]")

    if confirm.upper() == 'Y':

        Again = True

else:

    unallocated[process_name] = [process_number, process_size]

    confirm = input("Current process didn't fit in the memory!! Is
there any process still incoming?[Y/N]")

    if confirm.upper() == 'Y':

        Again = True


print("\nProcessName\tPartitionNumber\tProcessSize")
print("-----|-----|-----")
for k,v in processes.items():

    print(k.ljust(15)+str(v[0]).center(16)+str(v[1]).center(11))

print("\nExternal Fragmentation =", phymem-cumulativesum)

print("\n-----Unallocated processes-----")
print("ProcessName\tProcessSize")
for k,v in unallocated.items():
```

```
print(k.ljust(15)+str(v[1]).center(11))
```

**ASK USER THE SIZE OF PHYSICAL MEMORY FOR USER PROCESSES: 500 MB**

**ASK USER SIZE OF PROCESS1:**

**ALLOCATE PARTITION WHERE THIS PROCESS CAN FIT**

**DO YOU WANT TO CONTINUE: Y/N**

**IF YES**

**ASK USER SIZE OF PROCESS2:**

**DO YOU WANT TO CONTINUE: Y/N**

**N**

PROCESS NUMBER	PROCESS SIZE (MB)
1	70
2	90
3	110
4	150
5	100

**OUTPUT :** \* it should be same when your program gets executed



PROCESS NUMBER	PROCESS SIZE (MB)	PARTITION SIZE (MB)
1	70	70
2	90	90
3	110	110
4	150	150
5	100	CAN NOT FIT IN MEMORY

EXTERNAL FRAGMENTATION :  $500 - (70+90+110+150) = 80$  MB

### OUTPUT SCREEN SHOTS:

```
nag-1211@hp-laptop:~/Documents/3-1/05 Lab$ python3 MultiTaskingwithVariableTasks.py
Enter the size of Physical memory: 500
Enter the name of the process: chrome
Enter the size of the process: 70
Is there process incoming?[Y/N]y
Enter the name of the process: teams
Enter the size of the process: 90
Is there process incoming?[Y/N]y
Enter the name of the process: instagram
Enter the size of the process: 110
Is there process incoming?[Y/N]y
Enter the name of the process: youtube
Enter the size of the process: 150
Is there process incoming?[Y/N]n

ProcessName      PartitionNumber ProcessSize
-----|-----|-----
chrome           1              70
teams            2              90
instagram        3             110
youtube          4             150

External Fragmentation = 80

-----Unallocated processes-----
ProcessName      ProcessSize
nag-1211@hp-laptop:~/Documents/3-1/05 Lab$
```



## **VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY**

Permanently Affiliated to JNTU Kakinada, Approved by AICTE

Accredited by NAAC with 'A' Grade, ISO 9001:2008 Certified

Nambur, Pedakakani (M), Guntur (Dt) - 522508

### **DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

B.Tech Program is Accredited by NBA