EXPERIMENT - 6

Operating Systems Lab

AIM

Write a program to implement the first fit, best fit, and worst fit algorithm for memory management.

Syeda Reeha Quasar
14114802719
6C7

EXPERIMENT – 6

Aim:

Write a program to implement the first fit, best fit, and worst fit algorithm for memory management.

Theory:

Memory is the important part of the computer that is used to store the data. Its management is critical to the computer system because the amount of main memory available in a computer system is very limited. At any time, many processes are competing for it. Moreover, to increase performance, several processes are executed simultaneously. For this, we must keep several processes in the main memory, so it is even more important to manage them effectively.

- **First Fit**

  The first fit approach is to allocate the first free partition or hole large enough which can accommodate the process. It finishes after finding the first suitable free partition.

- **Best Fit**

  The best-fit deals with allocating the smallest free partition which meets the requirement of the requesting process. This algorithm first searches the entire list of free partitions and considers the smallest hole that is adequate. It then tries to find a hole that is close to the actual process size needed.

- **Worst fit**

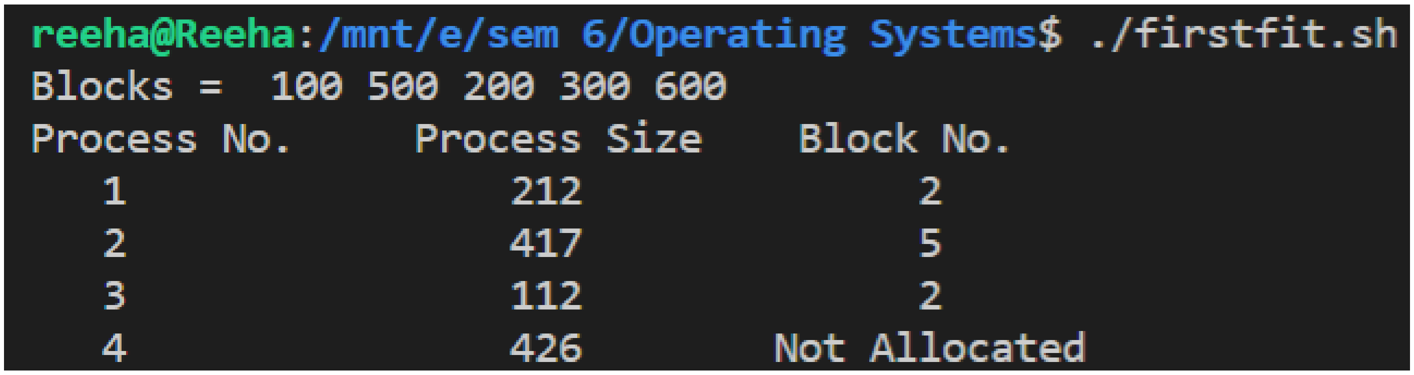
  In worst fit approach is to locate the largest available free portion so that the portion left will be big enough to be useful. It is the reverse of best fit.

Source Code:

a) First Fit

```
#!/bin/bash
block=(100 500 200 300 600)
echo "Blocks = " ${block[@]}
process=(212 417 112 426)
m=5
n=4
declare -a allocation
i=0
while [ $i -lt $n ]
do
    allocation[$i]=-1
    i=$((i+1))
done
j=0
while [ $j -lt $n ]
do
    k=0
    while [ $k -lt $m ]
    do
        if [ ${block[$k]} -ge ${process[$j]} ]
        then
            allocation[$j]=$k
            block[$k]=$((block[$k]-process[$j]))
            break
        fi
        k=$((k+1))
    done
    j=$((j+1))
done
echo "Process No.   Process Size    Block No."
a=0
while [ $a -lt $n ]
do
    if [ ${allocation[$a]} -ne -1 ]
    then
        echo "  " $((a+1)) "              " ${process[$a]} "            " $((alloca-
tion[$a]+1))
    else
        echo "  " $((a+1)) "              " ${process[$a]} "       Not Allocated "
    fi
    a=$((a+1))
done
```

Output:



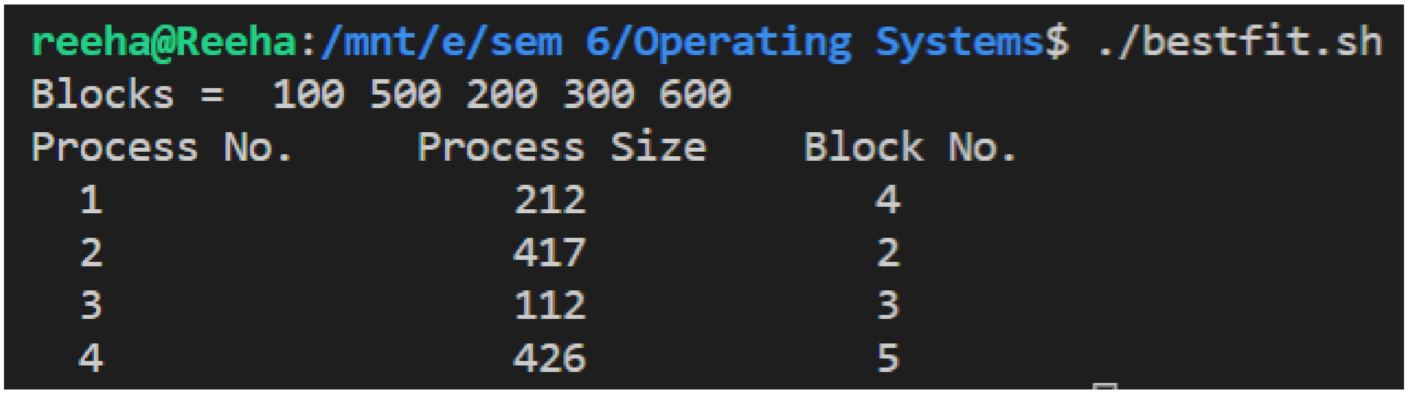
```
reeha@Reeha:/mnt/e/sem 6/Operating Systems$ ./firstfit.sh
Blocks =  100 500 200 300 600
Process No.     Process Size    Block No.
   1               212             2
   2               417             5
   3               112             2
   4               426         Not Allocated
```

b) Best Fit

```
block=(100 500 200 300 600)
process=(212 417 112 426)
echo "Blocks = " ${block[@]}
m=5
```

```
n=4
declare -a allocation
i=0
while [ $i -lt $n ]
do
    allocation[$i]=-1
    i=$((i+1))
done
j=0
while [ $j -lt $n ]
do
    bestindex=-1
    k=0
    while [ $k -lt $m ]
    do
        if [ ${block[$k]} -ge ${process[$j]} ]
        then
            if [ $bestindex -eq -1 ]
            then
                bestindex=$k
            else
                if [ ${block[$bestindex]} -gt ${block[$k]} ]
                then
                    bestindex=$k
                fi
            fi
        fi
        k=$((k+1))
    done
    if [ $bestindex -ne -1 ]
    then
        allocation[$j]=$bestindex
        block[$bestindex]=$((block[$bestindex]-process[$j]))
    fi
    j=$((j+1))
done
echo "Process No.   Process Size    Block No."
a=0
while [ $a -lt $n ]
do
    if [ ${allocation[$a]} -ne -1 ]
    then
        echo " " $((a+1)) "               " ${process[$a]} "          " $((alloca-
tion[$a]+1))
    else
        echo " " $((a+1)) "               " ${process[$a]} "    Not Allocated "
    fi
    a=$((a+1))
done
```

Output:

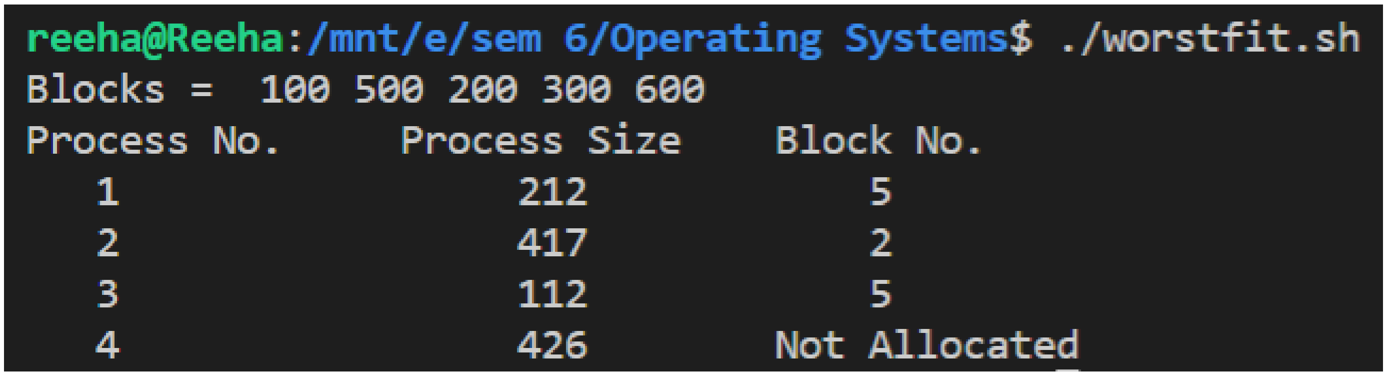


```
reeha@Reeha:/mnt/e/sem 6/Operating Systems$ ./bestfit.sh
Blocks =  100 500 200 300 600
Process No.     Process Size    Block No.
  1                 212            4
  2                 417            2
  3                 112            3
  4                 426            5
```

c) Worst Fit

```
block=(100 500 200 300 600)
process=(212 417 112 426)
echo "Blocks = " ${block[@]}
declare -a allocation
m=5
n=4
i=0
while [ $i -lt $n ]
do
    allocation[$i]=-1
    i=$((i+1))
done
j=0
while [ $j -lt $n ]
do
    worstindex=-1
    k=0
    while [ $k -lt $m ]
    do
        if [ ${block[$k]} -ge ${process[$j]} ]
        then
            if [ $worstindex -eq -1 ]
            then
                worstindex=$k
            else
                if [ ${block[$worstindex]} -lt ${block[$k]} ]
                then
                    worstindex=$k
                fi
            fi
        fi
        k=$((k+1))
    done
    if [ $worstindex -ne -1 ]
    then
        allocation[$j]=$worstindex
        block[$worstindex]=$((block[$worstindex]-process[$j]))
    fi
    j=$((j+1))
done
echo "Process No.   Process Size    Block No."
a=0
while [ $a -lt $n ]
do
    if [ ${allocation[$a]} -ne -1 ]
    then
        echo "   " $((a+1)) "                " ${process[$a]} "     " $((alloca-
tion[$a]+1))
    else
        echo "   " $((a+1)) "                " ${process[$a]} "  Not Allocated"
    fi
    a=$((a+1))
done
```

Output:



```
reeha@Reeha:/mnt/e/sem 6/Operating Systems$ ./worstfit.sh
Blocks =  100 500 200 300 600
Process No.     Process Size    Block No.
   1                212              5
   2                417              2
   3                112              5
   4                426         Not Allocated
```