EXPERIMENT NO : 8

AIM: Implement Variable Sized Partitioning and Fixed Sized Partitioning for First Fit, Best Fit and Worst Fit:

- **Variable Sized Partitioning:**
- **First Fit:**
- **CODE:**

```
def FirstFit(block_Size, m, process_Size, n):
    # code to store the block id of the block that needs to be allocated to a process
    allocate = [-1] * n

    # Any process is assigned with the memory at the initial stage

    # find a suitable block for each process
    # the blocks are allocated as per their size

    for i  in  range(n):
        for j in range(m):
            if block_Size[j]  >= process_Size[i]:
                # assign the block j to p[i] process
                allocate[i] = j

                # available block memory is reduced
                block_Size[j] -= process_Size[i]
                break

    print("Process No. Process Size Block No.")

    for i in range(n):
        print(str(i + 1) + "\t\t\t" + str(process_Size[i]) + "\t\t\t", end=" ")

        if allocate[i] != -1:
            print(allocate[i] + 1)
        else:
            print("Not Allocated")


# Driver code

block_Size = [100, 50, 30, 120, 35]
process_Size = [20, 60, 70, 40]
m = len(block_Size)
n = len(process_Size)

FirstFit(block_Size, m, process_Size, n)
```

- **OUTPUT:**

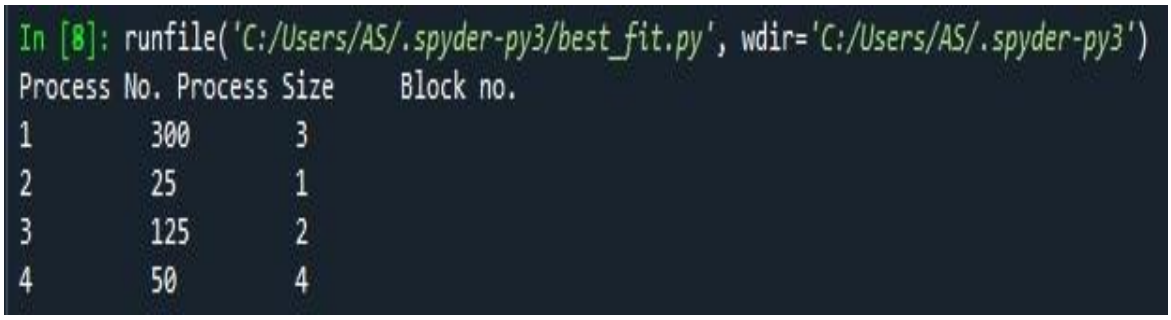
```
In [6]: runfile('C:/Users/AS/.spyder-py3/first_fit.py', wdir='C:/Users/AS/.spyder-py3')
Process No. Process Size Block No.
1           300          3
2           25           1
3           125          2
4           50           4
```

- **Best Fit:**
- **CODE:**

```
def bestFit(blockSize, m, processSize, n):

        # Stores block id of the block
        # allocated to a process
        allocation = [-1] * n

        # pick each process and find suitable
        # blocks according to its size ad
        # assign to it
        for i in range(n):

          # Find the best fit block for
          # current process
          bestIdx = -1
          for j in range(m):
                if blockSize[j] >= processSize[i]:
                      if bestIdx == -1:
                            bestIdx = j
                      elif blockSize[bestIdx] > blockSize[j]:
                            bestIdx = j

          # If we could find a block for
          # current process
          if bestIdx != -1:

                # allocate block j to p[i] process
                allocation[i] = bestIdx

                # Reduce available memory in this block.
                blockSize[bestIdx] -= processSize[i]
```

```
                print("Process No. Process Size        Block no.")
                for i in range(n):
                  print(i + 1, "              ", processSize[i],
                                                          end = "          ")
                  if allocation[i] != -1:
                        print(allocation[i] + 1)
                  else:
                        print("Not Allocated")

# Driver code
if __name___== '__main__':
                blockSize =[100, 50, 30, 120, 35]
                processSize = [20, 60, 70, 40]
                m = len(blockSize)
                n = len(processSize)

                bestFit(blockSize, m, processSize, n)
```

- **OUTPUT:**

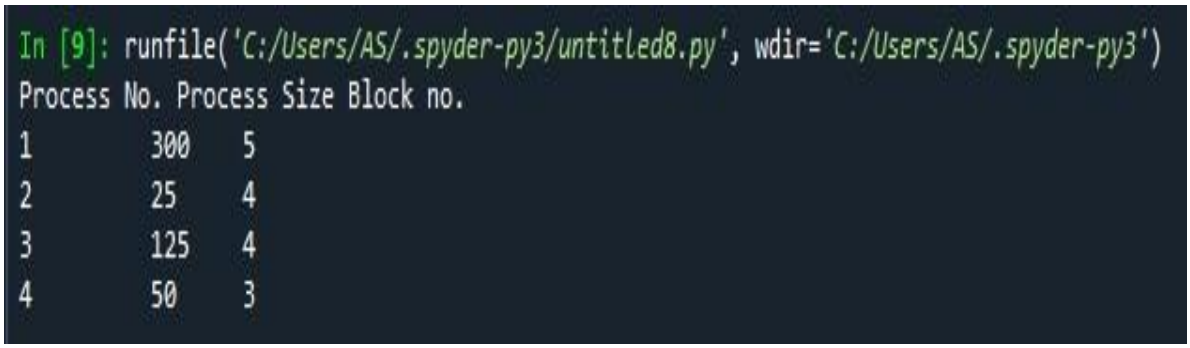


- **Worst Fit:**
- **CODE:**

```
def worstFit(blockSize, m, processSize, n):

        # Stores block id of the block
        # allocated to a process

        # Initially no block is assigned
        # to any process
        allocation = [-1] * n

        # pick each process and find suitable blocks
        # according to its size ad assign to it
        for i in range(n):

          # Find the best fit block for
          # current process
          wstIdx = -1
          for j in range(m):
                if blockSize[j] >= processSize[i]:
                      if wstIdx == -1:
                            wstIdx = j
                      elif blockSize[wstIdx] < blockSize[j]:
                            wstIdx = j

          # If we could find a block for
          # current process
          if wstIdx != -1:

                # allocate block j to p[i] process
                allocation[i] = wstIdx

                # Reduce available memory in this block.
                blockSize[wstIdx] -= processSize[i]

        print("Process No. Process Size Block no.")
        for i in range(n):
          print(i + 1, "               ",
                processSize[i], end = "           ")
          if allocation[i] != -1:
                print(allocation[i] + 1)
          else:
                print("Not Allocated")
```

```
# Driver code
if __name__== '__main__':
        blockSize = [100, 500, 200, 300, 600]
        processSize = [212, 417, 112, 426]
        m = len(blockSize)
        n = len(processSize)

        worstFit(blockSize, m, processSize, n)
```

- **OUTPUT:**



- **Fixed Sized Partitioning:**

- **First Fit:**
- **CODE:**

```
def FirstFit(block_Size, m, process_Size, n):
    # code to store the block id of the block that needs to be allocated to a process
    allocate = [-1] * n

    # Any process is assigned with the memory at the initial stage

    # find a suitable block for each process
    # the blocks are allocated as per their size

    for i  in  range(n):
        for j in range(m):
            if block_Size[j]  >= process_Size[i]:
                # assign the block j to p[i] process
                allocate[i] = j

                # available block memory is reduced
                block_Size[j] -= process_Size[i]
                break

    print("Process No. Process Size Block No.")

    for i in range(n):
        print(str(i + 1) + "\t\t\t" + str(process_Size[i]) + "\t\t\t", end=" ")

        if allocate[i] != -1:
            print(allocate[i] + 1)
        else:
            print("Not Allocated")


# Driver code

blockSize = [200,400,600,500,300,250]
processSize = [357,210,468,491]
m = len(block_Size)
n = len(process_Size)

FirstFit(block_Size, m, process_Size, n)
```

- **OUTPUT:**

```
In [10]: runfile('C:/Users/AS/.spyder-py3/first_fit.py', wdir='C:/Users/AS/.spyder-py3')
Process No. Process Size Block No.
1           357          2
2           210          3
3           468          4
4           491          Not Allocated
```

- **Best Fit:**
- **CODE:**

```
def bestFit(blockSize, m, processSize, n):

        # Stores block id of the block
        # allocated to a process
        allocation = [-1] * n

        # pick each process and find suitable
        # blocks according to its size ad
        # assign to it
        for i in range(n):

          # Find the best fit block for
          # current process
          bestIdx = -1
          for j in range(m):
                if blockSize[j] >= processSize[i]:
                      if bestIdx == -1:
                            bestIdx = j
                      elif blockSize[bestIdx] > blockSize[j]:
                            bestIdx = j

          # If we could find a block for
          # current process
          if bestIdx != -1:

                # allocate block j to p[i] process
                allocation[i] = bestIdx

                # Reduce available memory in this block.
                blockSize[bestIdx] -= processSize[i]

        print("Process No. Process Size        Block no.")
        for i in range(n):
          print(i + 1, "              ", processSize[i],
                                                    end = "           ")
          if allocation[i] != -1:
```

```
                print(allocation[i] + 1)
            else:
                print("Not Allocated")

# Driver code
if __name__== '__main__':

        blockSize = [200,400,600,500,300,250]
        processSize = [357,210,468,491]
          m = len(blockSize)
          n = len(processSize)

          bestFit(blockSize, m, processSize, n)
```

- **OUTPUT:**

```
In [11]: runfile('C:/Users/AS/.spyder-py3/best_fit.py', wdir='C:/Users/AS/.spyder-py3')
Process No. Process Size     Block no.
1           357          2
2           210          6
3           468          4
4           491          3
```

- **Worst Fit:**
- **CODE:**

```
def worstFit(blockSize, m, processSize, n):

        # Stores block id of the block
        # allocated to a process

        # Initially no block is assigned
        # to any process
        allocation = [-1] * n

        # pick each process and find suitable blocks
        # according to its size ad assign to it
        for i in range(n):

          # Find the best fit block for
          # current process
          wstIdx = -1
          for j in range(m):
                if blockSize[j] >= processSize[i]:
                      if wstIdx == -1:
                            wstIdx = j
                      elif blockSize[wstIdx] < blockSize[j]:
                            wstIdx = j

          # If we could find a block for
          # current process
          if wstIdx != -1:

                # allocate block j to p[i] process
                allocation[i] = wstIdx

                # Reduce available memory in this block.
                blockSize[wstIdx] -= processSize[i]

        print("Process No. Process Size Block no.")
        for i in range(n):
          print(i + 1, "               ",
                processSize[i], end = "           ")
          if allocation[i] != -1:
                print(allocation[i] + 1)
          else:
                print("Not Allocated")
```

```
# Driver code
if __name__== '__main__':

                blockSize = [200,400,600,500,300,250]
                processSize = [357,210,468,491]
                  m = len(blockSize)
                  n = len(processSize)

                  worstFit(blockSize, m, processSize, n)
```

- **OUTPUT:**