21BDS0340

Abhinav Dinesh Srivatsa

Operating Systems Lab

Assignment – IV

Question 1

- First Fit

Code:

```
#include <iostream>
using namespace std;

int main()
{
    // getting input
    int n, m;
    cout << "Enter number of blocks available: ";
    cin >> n;
    cout << "Enter number of processes: ";
    cin >> m;

    int *block = new int[n], *process = new int[m];
    for (int x = 0; x < n; x++)
    {
        cout << "Enter block size " << (x + 1) << ": ";
        cin >> block[x];
    }
    for (int x = 0; x < m; x++)
    {
        cout << "Enter process size " << (x + 1) << ": ";
        cin >> process[x];
    }

    bool assigned = false;
    for (int x = 0; x < m; x++)
    {
        assigned = false;
        for (int y = 0; y < n; y++)
        {
            if (process[x] <= block[y])
            {
                cout << "Process " << (x + 1) << ", size " << process[x]
                     << ": allocated to block " << (y + 1) << " of size " <<
block[y]
                     << "\n";
                block[y] -= process[x];
```

```
                assigned = true;
                break;
            }
        }
        if (!assigned)
            cout << "Process " << (x + 1) << ", size " << process[x] << ": Not
allocated"
                 << "\n";
    }
}
```

Output:

```
Enter number of blocks available: 5
Enter number of processes: 4
Enter block size 1: 100
Enter block size 2: 500
Enter block size 3: 200
Enter block size 4: 300
Enter block size 5: 600
Enter process size 1: 212
Enter process size 2: 427
Enter process size 3: 112
Enter process size 4: 426
Process 1, size 212: allocated to block 2 of size 500
Process 2, size 427: allocated to block 5 of size 600
Process 3, size 112: allocated to block 2 of size 288
Process 4, size 426: Not allocated
```

- Best Fit

Code:

```
#include <iostream>
using namespace std;

void swap(int *x, int *y)
{
    int temp = *x;
    *x = *y;
    *y = temp;
}

void sortAscending(int n, int *array)
{
    int length = n;
    for (int x = 0; x < length - 1; x++)
        for (int y = x + 1; y < length; y++)
            if (array[x] > array[y])
                swap(array + x, array + y);
}

int main()
```

```
{
    // getting input
    int n, m;
    cout << "Enter number of blocks available: ";
    cin >> n;
    cout << "Enter number of processes: ";
    cin >> m;

    int *block = new int[n], *process = new int[m];
    for (int x = 0; x < n; x++)
    {
        cout << "Enter block size " << (x + 1) << ": ";
        cin >> block[x];
    }
    for (int x = 0; x < m; x++)
    {
        cout << "Enter process size " << (x + 1) << ": ";
        cin >> process[x];
    }

    bool assigned = false;
    for (int x = 0; x < m; x++)
    {
        sortAscending(n, block);
        assigned = false;
        for (int y = 0; y < n; y++)
        {
            if (process[x] <= block[y])
            {
                cout << "Process " << (x + 1) << ", size " << process[x]
                     << ": allocated to block of size " << block[y]
                     << "\n";
                block[y] -= process[x];
                assigned = true;
                break;
            }
        }
        if (!assigned)
            cout << "Process " << (x + 1) << ", size " << process[x] << ": Not
allocated"
                 << "\n";
    }
}
```

Output:

```
Enter number of blocks available: 5
Enter number of processes: 4
Enter block size 1: 100
Enter block size 2: 500
Enter block size 3: 200
Enter block size 4: 300
Enter block size 5: 600
Enter process size 1: 212
Enter process size 2: 417
Enter process size 3: 112
Enter process size 4: 426
Process 1, size 212: allocated to block of size 300
Process 2, size 417: allocated to block of size 500
Process 3, size 112: allocated to block of size 200
Process 4, size 426: allocated to block of size 600
```

- Worst Fit

Code:

```
#include <iostream>
using namespace std;

void swap(int *x, int *y)
{
    int temp = *x;
    *x = *y;
    *y = temp;
}

void sortDescending(int n, int *array)
{
    int length = n;
    for (int x = 0; x < length - 1; x++)
        for (int y = x + 1; y < length; y++)
            if (array[x] < array[y])
                swap(array + x, array + y);
}

int main()
{
    // getting input
    int n, m;
    cout << "Enter number of blocks available: ";
    cin >> n;
    cout << "Enter number of processes: ";
    cin >> m;

    int *block = new int[n], *process = new int[m];
    for (int x = 0; x < n; x++)
    {
```

```
        cout << "Enter block size " << (x + 1) << ": ";
        cin >> block[x];
    }
    for (int x = 0; x < m; x++)
    {
        cout << "Enter process size " << (x + 1) << ": ";
        cin >> process[x];
    }

    bool assigned = false;
    for (int x = 0; x < m; x++)
    {
        sortDescending(n, block);
        assigned = false;
        for (int y = 0; y < n; y++)
        {
            if (process[x] <= block[y])
            {
                cout << "Process " << (x + 1) << ", size " << process[x]
                     << ": allocated to block of size " << block[y]
                     << "\n";
                block[y] -= process[x];
                assigned = true;
                break;
            }
        }
        if (!assigned)
            cout << "Process " << (x + 1) << ", size " << process[x] << ": Not
allocated"
                 << "\n";
    }
}
```

Output:

```
Enter number of blocks available: 5
Enter number of processes: 4
Enter block size 1: 100
Enter block size 2: 500
Enter block size 3: 200
Enter block size 4: 300
Enter block size 5: 600
Enter process size 1: 212
Enter process size 2: 417
Enter process size 3: 112
Enter process size 4: 426
Process 1, size 212: allocated to block of size 600
Process 2, size 417: allocated to block of size 500
Process 3, size 112: allocated to block of size 388
Process 4, size 426: Not allocated
```

Question 2

```c
#include <stdio.h>
#include <pthread.h>

void *threadProcess1(void *arg)
{
    printf("Thread 1 starting...\n");
    printf("Printing values 0 - 9\n");
    for (int x = 0; x < 10; x++)
        printf("%d\n", x);
    printf("Thread 1 executed!\n");
    return NULL;
}

void *threadProcess2(void *arg)
{
    printf("Thread 2 starting...\n");
    printf("Printing letters a - j\n");
    for (int x = 97; x < 107; x++)
        printf("%c\n", x);
    printf("Thread 2 executed!\n");
    return NULL;
}

int main()
{
    pthread_t thread1, thread2;

    pthread_create(&thread1, NULL, threadProcess1, NULL);
    pthread_create(&thread2, NULL, threadProcess2, NULL);

    pthread_join(thread1, NULL);
    pthread_join(thread2, NULL);
}
```

Output:

```
Thread 1 starting...
Printing values 0 - 9
0
1
2
3
4
5
6
7
8
9
Thread 1 executed!
Thread 2 starting...
Printing letters a - j
a
b
c
d
e
f
g
h
i
j
Thread 2 executed!
```