Учреждение образования Республики Беларусь

«Гомельский государственный технический университет им. П. О. Сухого»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

Отчет по лабораторной работе №6

по дисциплине «Операционные системы»

Выполнил: студент группы ИТП-11

Половцев М. С.

Принял: преподаватель-стажер

Карась О. В.

Гомель, 2022

**Цель:** разработать программу, осуществляющую моделирование режима работы с разделение времени.

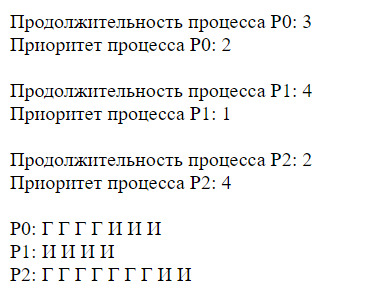
**Задание**

В соответствии с вариантом разработать программу на языке С# (или другом), осуществляющую моделирование работы планировщика процессов.

Моделирование режима разделения времени выполняется в соответствии с заданием. После запуска программа должна в диалоговом режиме ввести информацию процессах.

Вариант 6: Алгоритм SJF. Не вытесняющий, приоритетный.

Результат работы программы представлен на рисунке 1:



**Вывод:** разработал программу, осуществляющую моделирование режима работы по алгоритму SJF (не вытесняющий, приоритетный), на языке JavaScript.

ЛИСТИНГ

HTML:

<html>

<head>

<title>Лабораторная работа 6</title>

<meta charset = 'utf-8'>

</head>

<body>

<script type = 'text/JavaScript' src = 'D:\Операционные системы\Лабораторные работы\Лабораторная работа 6\JS.js'></script>

</body>

</html>

JS:

let n = parseInt(prompt('Введите количество процессов'));

let time = [];

let priority = [];

let sumoftime = 0;

for (let i = 0; i < n; i++)

{

time[i] = parseInt(prompt(`Введите продолжительность процесса P${i}`));

priority[i] = parseInt(prompt(`Введите приоритет процесса P${i}`));

document.write(`Продолжительность процесса P${i}: ${time[i]}<br>`);

document.write(`Приоритет процесса P${i}: ${priority[i]}<br>`);

document.write('<br>');

sumoftime += time[i];

}

let table = [];

for (let i = 0; i < n; i++)

{

table[i] = [];

for (let j = 0; j < sumoftime; j++)

{

table[i][j] = '0';

}

}

let copytime = time;

let sorttime = time.slice(0).sort();

let copyprior = priority;

let sortprior = priority.slice(0).sort();

for (let i = 0; i < n; i++)

{

for (let j = 0; j < n; j++)

{

if (sortprior[i] == copyprior[j])//это сравнение для нахождения процесса, выполняющегося первым

{

for (let k = 0; k < copytime[j]; k++)

{

table[j][k] = 'И';

}

}

}

}

/\*

for (let i = 0; i < n; i++)

{

for (let j = 0; j < sumoftime; j++)

{

document.write(table[i][j] + ' ');

}

document.write('<br>');

}

\*/

let sum = 0;

let num = 0;

for (let i = 1; i < n; i++)

{

for (let j = 0; j < n; j++)

{

if (sortprior[i] == copyprior[j])

{

for (let k = 0; k < n; k++)

{

if (sortprior[i - 1] == copyprior[k])

{

num = k;//для нахождения номера предыдущего процесса

}

}

sum += copytime[num];

for (let z = 0; z < sum; z++)

{

table[j][z + copytime[j]] = 'Г';

}

}

}

}

for (let i = 0; i < n; i++)

{

document.write(`P${i}: `);

for (let j = sumoftime - 1; j >= 0; j--)

{

if (table[i][j] == 0)

continue;

else

document.write(table[i][j] + ' ');

}

document.write('<br>');

}