**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5

по дисциплине «Операционные системы»

по теме «Планирование процессов»

Выполнил студент

Группы ИТ-12

Кононович Е.М.

Принял преподаватель

Карась О. В.

Гомель 2022

**Лабораторная работа №5**

**«Планирование процессов»**

**Цель работы:** ознакомиться с основными алгоритмами планирования процессов, вычислить среднее время исполнения и ожидания для конкретных процессов.

**1. Теоретические сведения**

**1. Алгоритмы планирования**

Существует достаточно большой набор разнообразных алгоритмов планирования, которые предназначены для достижения различных целей и эффективны для разных классов задач. Многие из них могут использоваться на нескольких уровнях планирования. В этом разделе мы рассмотрим некоторые наиболее употребительные алгоритмы применительно к процессу кратковременного планирования.

**1.1 First-Come, First-Served (FCFS)**

Простейшим алгоритмом планирования является алгоритм, который принято обозначать аббревиатурой FCFS по первым буквам его английского названия – First-Come, First-Served (первым пришел, первым обслужен). Представим себе, что процессы, находящиеся в состоянии готовность, выстроены в очередь. Когда процесс переходит в состояние готовность, он, а точнее, ссылка на его PCB помещается в конец этой очереди. Выбор нового процесса для исполнения осуществляется из начала очереди с удалением оттуда ссылки на его PCB. Очередь подобного типа имеет в программировании специальное наименование – FIFO1), сокращение от First In, First Out (первым вошел, первым вышел).

**1.2 Round Robin (RR)**

Модификацией алгоритма FCFS является алгоритм, получивший название Round Robin (Round Robin – это вид детской карусели в США) или сокращенно RR. По сути дела, это тот же самый алгоритм, только реализованный в режиме вытесняющего планирования. Можно представить себе все множество готовых процессов организованным циклически – процессы сидят на карусели. Карусель вращается так, что каждый процесс находится около процессора небольшой фиксированный квант времени, обычно 10 – 100 миллисекунд. Пока процесс находится рядом с процессором, он получает процессор в свое распоряжение и может исполняться.

**1.3 Shortest-Job-First (SJF)**

При рассмотрении алгоритмов FCFS и RR мы видели, насколько существенным для них является порядок расположения процессов в очереди процессов, готовых к исполнению. Если короткие задачи расположены в очереди ближе к ее началу, то общая производительность этих алгоритмов значительно возрастает. Если бы мы знали время следующих CPU burst для процессов, находящихся в состоянии готовность, то могли бы выбрать для исполнения не процесс из начала очереди, а процесс с минимальной длительностью CPU burst. Если же таких процессов два или больше, то для выбора одного из них можно использовать уже известный нам алгоритм FCFS. Квантование времени при этом не применяется. Описанный алгоритм получил название "кратчайшая работа первой" или Shortest Job First (SJF).

SJF-алгоритм краткосрочного планирования может быть как вытесняющим, так и невытесняющим. При невытесняющем SJF-планировании процессор предоставляется избранному процессу на все необходимое ему время, независимо от событий, происходящих в вычислительной системе. При вытесняющем SJF-планировании учитывается появление новых процессов в очереди готовых к исполнению (из числа вновь родившихся или разблокированных) во время работы выбранного процесса. Если CPU burst нового процесса меньше, чем остаток CPU burst у исполняющегося, то исполняющийся процесс вытесняется новым.

**1.4 Приоритетное планирование**

Алгоритмы SJF и гарантированного планирования представляют собой частные случаи приоритетного планирования. При приоритетном планировании каждому процессу присваивается определенное числовое значение – приоритет, в соответствии с которым ему выделяется процессор. Процессы с одинаковыми приоритетами планируются в порядке FCFS. Для алгоритма SJF в качестве такого приоритета выступает оценка продолжительности следующего CPU burst. Чем меньше значение этой оценки, тем более высокий приоритет имеет процесс. Для алгоритма гарантированного планирования приоритетом служит вычисленный коэффициент справедливости. Чем он меньше, тем больше у процесса приоритет.

**2. Индивидуальные задания**

**2.1 Не вытесняющие алгоритмы планирования процессов**

Выполнить различные алгоритмы планирований – First-Come, First-Served (FCFS) (прямой и обратный), Round Robin (RR), Shortest-Job-First (SJF) (не вытесняющий), Shortest-Job-First (SJF) (не вытесняющий приоритетный) для данных приведенных в таблице 2.1 в соответствии со своим вариантом (номер по журналу). Вычислить полное время выполнения все процессов и каждого в отдельности, время ожидание для каждого процесса. Рассчитать среднее время выполнения процесса и среднее время ожидания. Результаты оформить в виде таблиц иллюстрирующих работу процессов.

**2.2 Вытесняющие алгоритмы планирования процессов**

Выполнить различные алгоритмы планирований – Shortest-Job-First (SJF) (вытесняющий) и Shortest-Job-First (SJF) (приоритетный) для данных приведенных в таблице 2.1 в соответствии со своим вариантом. Вычислить полное время выполнения все процессов и каждого в отдельности, время ожидание для каждого процесса. Рассчитать среднее время выполнения процесса и среднее время ожидания. Результаты оформить в виде таблиц иллюстрирующих работу процессов.

Таблица 2.1

|  |  |  |  |
| --- | --- | --- | --- |
| Вариант | Продолжительность процессов | Время появления в очереди | Приоритеты процессов |
| 14 | P0 – 7; P1 – 8; P2 – 3; P3 – 4 | P0 – 4; P1 – 0; P2 – 3; P3 – 5; | P0 – 1; P1 – 2; P2 – 3; P3 – 4; |

Результат выполнения лабораторной работы приведен на рисунке 1

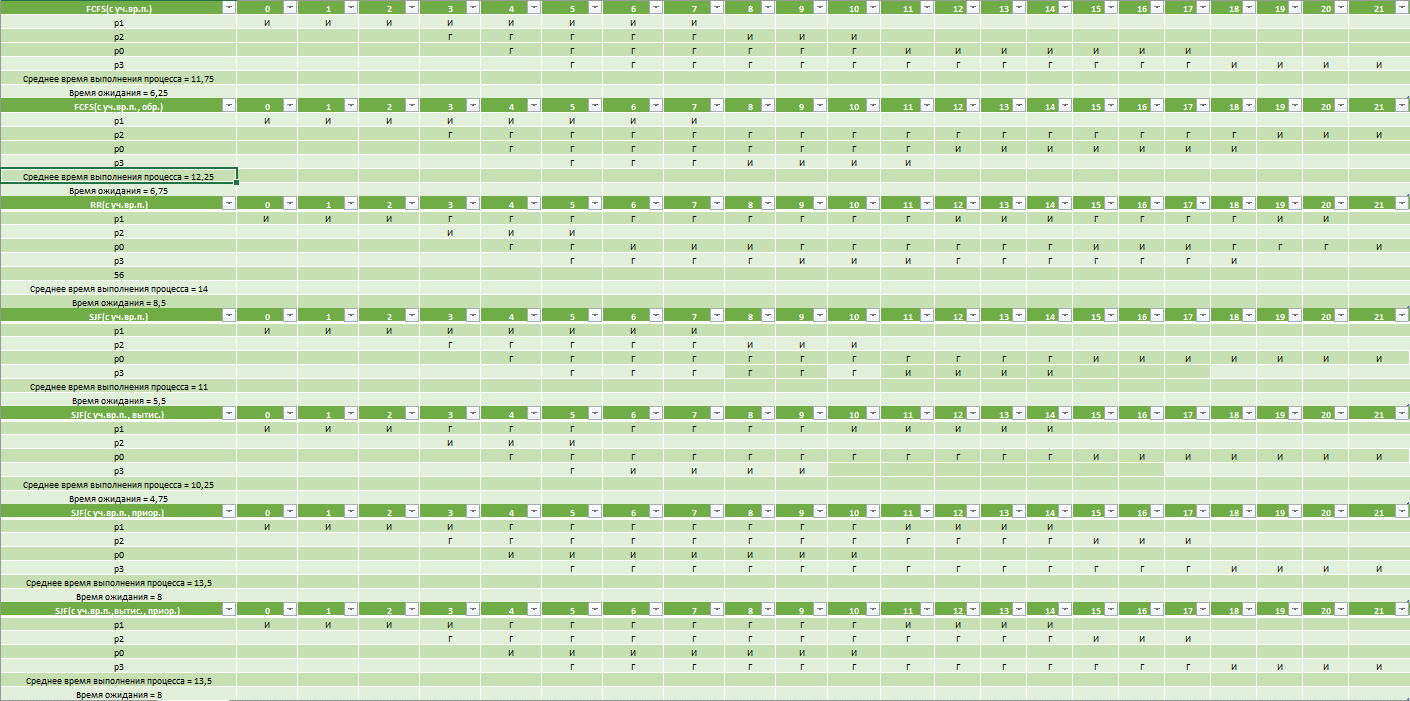


Рисунок 1 – результат выполнения лабораторной работы.

**Вывод:** Мы ознакомились с основными алгоритмами планирования процессов, были вычислены среднее время исполнения и ожидания для конкретных процессов.