МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО

ОБРАЗОВАНИЯ

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ АВТОМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

КАФЕДРА ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ



Расчетно-Графическое Задание

По дисциплине «Операционные Системы»

Студент: Долматов Максим Максимович

Группа: АВТ-342

Исходный шифр: 993

Срок представления работы (проекта) к защите « » 2025 г.

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дыминский И. И. (подпись, дата) (инициалы, фамилия)

Новосибирск, 2025

Оглавление

[**I. Теоретическая часть** 3](#_Toc199839670)

[**1. Теоретические основы планирования и диспетчеризации** 3](#_Toc199839671)

[1.1 Планирование 3](#_Toc199839672)

[1.2 Диспетчеризация 5](#_Toc199839673)

[**II. Практическая часть** 8](#_Toc199839674)

[**1. Раздел №1** 8](#_Toc199839675)

[1.1 Задание 8](#_Toc199839676)

[1.2 Исходные данные 8](#_Toc199839677)

[1.3 Временные диаграммы использования FIFO и SJF. Таблицы результатов 11](#_Toc199839678)

[1.4 Вывод 20](#_Toc199839679)

[**2. Раздел №2** 21](#_Toc199839680)

[2.1 Задание 21](#_Toc199839681)

[2.2 Исходные данные 21](#_Toc199839682)

[2.3 Временные диаграммы работы LIFO и SJF. Таблицы трассировки, расчет 22](#_Toc199839683)

[2.4 Выводы 29](#_Toc199839684)

[**Список источников** 30](#_Toc199839685)

# **I. Теоретическая часть**

## **1. Теоретические основы планирования и диспетчеризации**

### 1.1 Планирование

Основной задачей службы управления процессами является эффективное распределение аппаратных ресурсов центрального процессора (CPU). Эта функция включает в себя два ключевых компонента:

* Планировщик заданий — отвечает за выбор заданий, которые будут обрабатываться системой;
* Планировщик задач (или процессов) — управляет выполнением уже созданных процессов.

Задание представляет собой совокупность действий, которые пользователь хочет выполнить на вычислительной машине. Оно может быть разбито на отдельные этапы, называемые шагами задания. На основе этих шагов формируются задачи, а для их исполнения система создаёт соответствующие процессы.

Таким образом, планировщик заданий работает с заявками пользователя, тогда как планировщик задач оперирует процессами внутри системы.

Планировщик заданий определяет, какие именно задания и в какой последовательности будут запущены на выполнение, что делает его своего рода «макропланировщиком». В свою очередь, планировщик задач выполняет роль «микропланировщика», распределяя процессорное время между различными процессами.

В условиях мультипрограммирования планировщик заданий выбирает сразу несколько заявок из общего пула и загружает их в систему. Для каждого такого задания создается JCB (Job Control Block) — структура данных, описывающая параметры задания. Далее планировщик заданий запускает процессы, необходимые для выполнения задач, и выделяет им требуемые ресурсы (оперативная память, внешние устройства и т.д.).

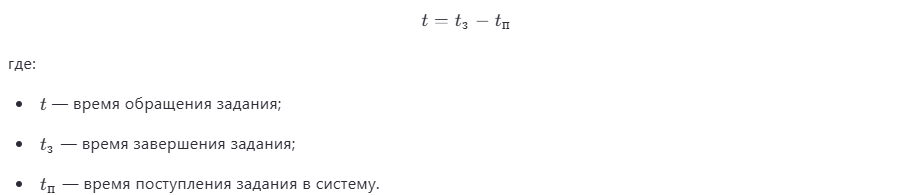
После этого планировщик процессов решает, какому из активных процессов передать CPU, в какой момент времени и на какой срок.

Выбор заданий для обработки осуществляется по определенным правилам, называемым дисциплинами обслуживания. При этом учитываются такие характеристики заданий, как приоритет, объем требуемых ресурсов и другие факторы. Планировщик заданий не только распределяет ресурсы при запуске задания, но также освобождает их после завершения его выполнения.

Дисциплина обслуживания — это правило, согласно которому из очереди выбирается очередное задание для выполнения. Подробная классификация таких дисциплин приведена на рисунке 1.

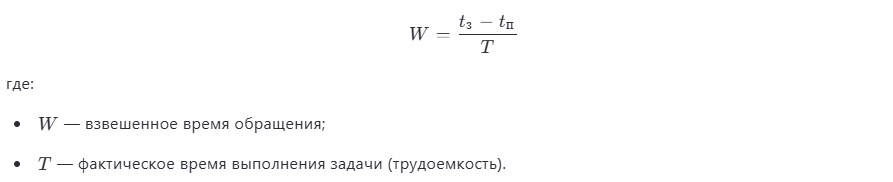
**Оценка эффективности планирования**

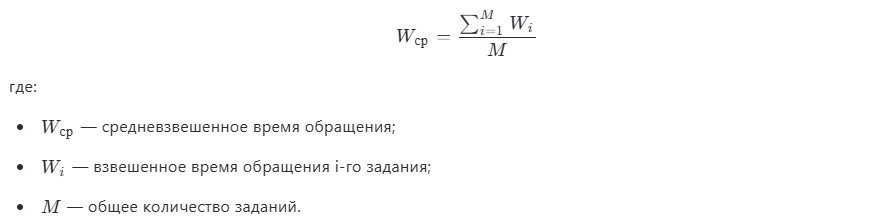
Существует несколько показателей, позволяющих оценить эффективность работы планировщика. Одним из них является время обращения задания — временной интервал от момента поступления задания в систему до окончания его выполнения:



Однако данный показатель не всегда универсален. Например, одночасовое задание будет иметь большее значение времени обращения, чем одноминутное, даже если оба были обработаны одинаково эффективно. Это связано с тем, что минимальное возможное время обращения не может быть меньше самого времени выполнения задачи.

Более объективной мерой служит взвешенное время обращения, которое учитывает длительность выполнения задания:



Для оценки эффективности всей системы можно использовать среднее взвешенное время обращения:

Этот показатель позволяет сравнивать производительность систем независимо от продолжительности выполняемых задач.

### 1.2 Диспетчеризация

Диспетчеризация относится к среднему уровню планирования. На этом этапе диспетчер задач выбирает одну из готовых к выполнению задач и предоставляет ей процессор. Обычно каждая задача получает процессорное время на короткий период, недостаточный для полного завершения. После истечения этого временного кванта происходит повторный выбор задачи, и процессор передается следующей заявке. Таким образом, диспетчер принимает решения в режиме реального времени, исходя из текущей ситуации в системе.

Цели диспетчеризации включают:

* Рациональное распределение процессорного времени с учетом заданных критериев;
* Эффективную реализацию алгоритмов управления задачами;
* Сбалансированное использование системных ресурсов;
* Компромисс между временем отклика и степенью загрузки оборудования.

Диспетчер — это программный модуль, который выбирает задачи из очереди готовых процессов, переводит их в активное состояние и передает управление центральному процессору.

Алгоритм работы диспетчера включает следующие этапы:

1. Анализ причины прерывания текущего процесса (например, истечение кванта времени или завершение задачи);
2. Выбор следующей задачи из очереди готовых процессов в соответствии с выбранной дисциплиной обслуживания;
3. Переключение контекста и передача управления процессором новой задаче.

В системах с поддержкой приоритетов активный процесс может быть прерван, если в очереди появилась задача с более высоким приоритетом.

**Оценка качества диспетчеризации**

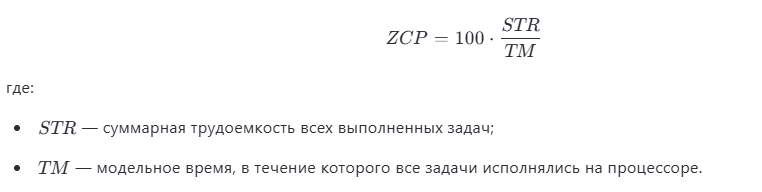
К качеству диспетчеризации предъявляются следующие требования:  
a) равномерное время ожидания (*t* ож​) для всех задач;  
b) минимальная трудоемкость (*TR*) используемой дисциплины обслуживания.

На практике применяются различные дисциплины диспетчеризации:

1. FIFO — «первым пришел — первым обслужен». Время выполнения соответствует трудоемкости заявки.
2. LIFO — «последним пришел — первым обслужен». Похож на FIFO, но с обратным порядком выбора.
3. RR (Round Robin) — круговой алгоритм, при котором каждая задача получает фиксированный квант времени.
4. PRT (Priority Scheduling) — выбор задачи по приоритету, где меньшее число означает более высокий приоритет.
5. RAND — случайный выбор заявки из очереди.
6. SJF (Shortest Job First) — сначала обслуживаются самые короткие задачи.
7. FB (Feedback) — многоуровневый RR, предусматривающий переход задач между уровнями в зависимости от поведения.

Об эффективности дисциплины судят по таким метрикам, как среднее время обслуживания (STO), трудоемкость (TR) и время ожидания (TO).

Еще одним важным показателем является коэффициент загрузки процессора (ZCP), выражаемый в процентах:



Этот параметр отражает степень использования вычислительных мощностей системы.

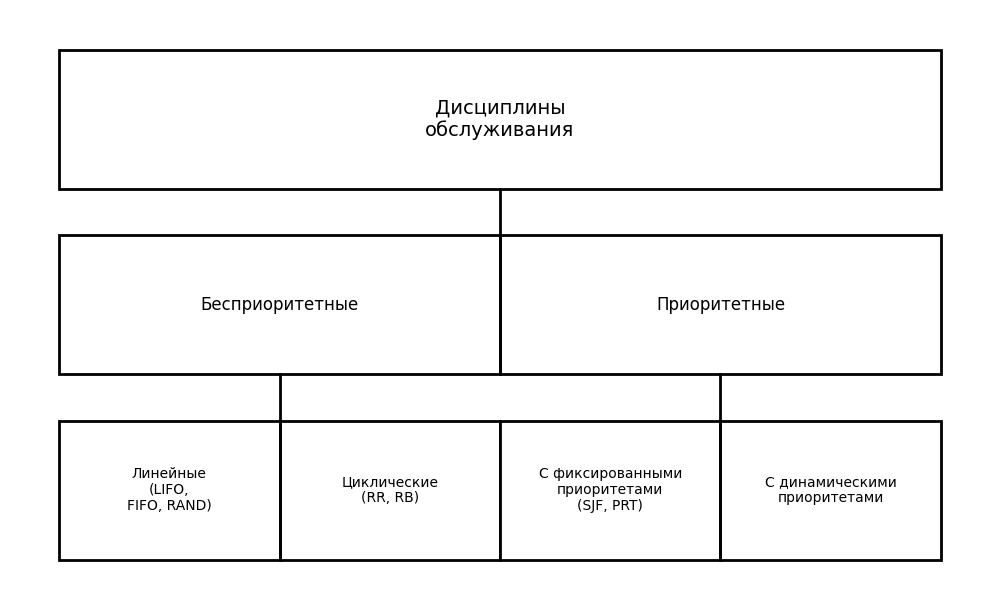


Рис. 1 Дисциплины обслуживания планирования процессов схема

# **II. Практическая часть**

## **1. Раздел №1**

### 1.1 Задание

Вычислительная система располагает оперативной памятью (ОП) V и внешним объемом памяти Н (НМД). ОП память выделяется переме­щаемым разделами, которые исключают влияние фрагментации. Реали­зуется режим мультипрограммирования: если одновременно выполня­ется несколько задач, то процессорное время распределяется между ними равномерно. В систему поступает поток из М заданий, очеред­ное задание поступает через время , для простоты каждое задание состоит из одной задачи и требует объем ОП - , объем внешней памяти , процессорное время. Каждое задание использует свою внешнюю память только для ввода данных в течение времени , после чего начинается счет. Однако закрепленные за каждым зада­нием носители освобождаются только после завершения задания. Предположим, возможно параллельное использование внешней памяти заданиями без задержки друг друга. Если бы задания выполнялись по одному, то на каждое задание было бы затрачено время . Вновь поступившие задания помещаются в очередь. Для выбора заданий из очереди на выполнение используются два алгоритма:

1. среди заданий в очереди, для которых достаточно свободных ре­сурсов, выбирается задание, поступившее первым (правило FIFO);
2. среди заданий в очереди, для которых достаточно свободных ре­сурсов, выбирается задание с наименьшим (правило SJF).

Необходимо построить временную диаграмму мультипрограммной работы при использовании каждого из двух алгоритмов. На диа­грамме выделить события (моменты поступления заданий, моменты назначения на выполнение, моменты начала счета, моменты заверше­ния) и периоды между событиями. Для каждого периода указать про­цессорное время на задание, доступную память, доступную дисковую память, степень мультипрограммирования. Провести сравнение двух случаев по средневзвешенному времени обращения.

### 1.2 Исходные данные

Средневзвешенное время обращения:



где - время завершения задания,- время поступления задания в систему.

Для нормирования различных вариантов последовательностей заданий используется набор из 10 типов задач (см. таблицу 1). Каждое задание включает одну из этих 10 задач. В одном потоке заданий могут встретиться задания, содержащие одинаковые задачи. Номер задачи Кi для очередного задания определяется по формулам:

;

, , ,

где [c] - целая часть числа с, - остаток от деления y на z, , – три последние цифры шифра студента.

Значение используемых параметров: , , , , последовательность периодов времени (интервал поступления заданий) .

Таблица 1. Набор задач для нормировки вариантов.

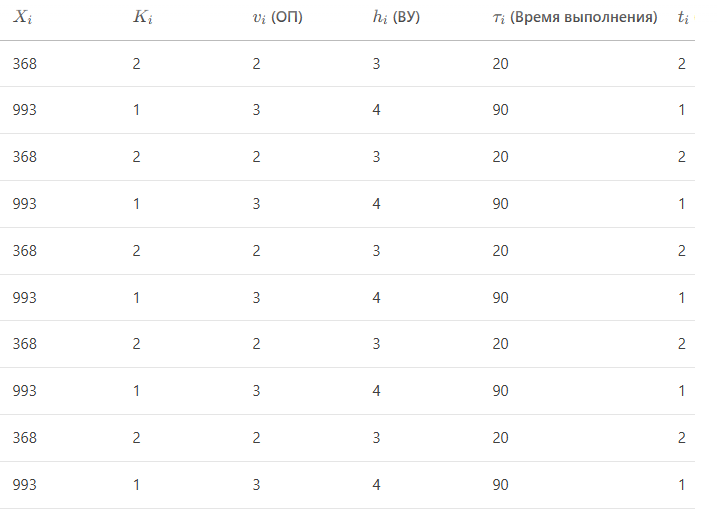
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| K | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| v | 6 | 3 | 2 | 4 | 3 | 5 | 7 | 9 | 4 | 1 |
| h | 2 | 4 | 3 | 1 | 2 | 0 | 4 | 1 | 6 | 3 |
| τ | 60 | 90 | 20 | 10 | 60 | 30 | 70 | 30 | 40 | 20 |

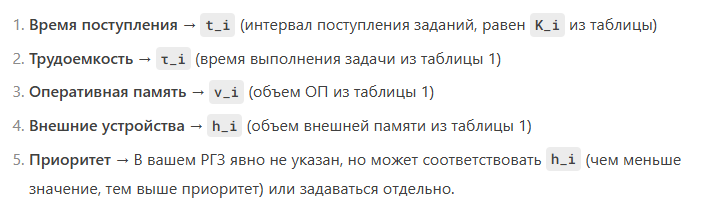
Используя индивидуальный расстановочный код: , были рассчитаны номера задач:

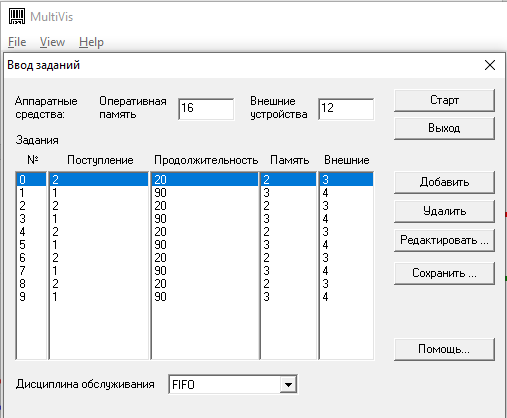
Для  последовательность *Ki*​ (номера задач) выглядит так:

2, 1, 2, 1, 2, 1, 2, 1, 2, 1.

Таблица 2. Характеристики заданий.







### 1.3 Временные диаграммы использования FIFO и SJF. Таблицы результатов

#### **ДО FIFO**

Временная диаграмма мультипрограммной работы ЭВМ при исходных данных для ДО FIFO (см. табл. 3) показана на рис. 3.

Таблица 3. Последовательность событий в системе (FIFO).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Время | Событие | Свободный объем ОП (16) | Свободный объем ВОП (12) |  |
| 0 | Запуск системы. Ничего не происходит. | 16 | 12 | 0 |
| 1 | Так как ресурсов хватает происходит ввод 2(3,4), 4(3,4), 6(3,4) процессов. Процессы 8(3,4) и 10(3,4) помещаются в очередь из-за недостатка ВОП. |  |  | 0 |
| 2 | Поступили остальные процессы 1(2,3), 3(2,3), 5(2,3), 7(2,3), 9(2,3). Так как не хватает ВОП они находятся в очереди. | 7 | 0 | 0 |
| 21 | Процессы 2(3,4), 4(3,4), 6(3,4) поступили на выполнение. Освобождение ресурсов ВОП. | 7 | 0+4+4+4=12 | 3 |
| 291 | Завершение выполнения процессов 2(3,4), 4(3,4), 6(3,4). Освобождение ОП. Переход процессов 1(2,3), 8(3,4), 10(3,4) из состояния очереди на ввод. Использование ОП и ВОП. | 7+3+3+3-2-3-3=8 | 12-4-4-3=1 | 0 |
| 306 | Переход процесса 1(2,3) из состояния ввода в состояние выполнения. Освобождение ВОП. | 8 |  | 1 |
| 311 | Продолжение выполнения 1(2,3) процесса, а также переход 8(3,4) и 10(3,4) процессов из состояния ввода в состояние выполнения. Освобождение ВОП. |  |  | 3 |
| 358 | Завершен процесс 1(2,3). Окончательное освобождение всех ОП, занимающих этот процесс. Переход 3(2,3) процесса в состояние ввода. | 8+2-2=8 |  | 2 |
| 373 | Переход процесса 3(2,3) из состояния ввода в состояние выполнения. Освобождение ВОП. | 8 |  | 3 |
| 433 | Завершен процесс 3(2,3) освобождение ресурсов. Переход процесса 5(2,3) из состояния очереди в состояние ввода. Использования ОП и ВОП. |  | 12-3=9 | 2 |
| 448 | Переход процесса 5(2,3) из состояния ввода в состояние выполнения. Освобождение ВОП. |  |  | 3 |
| 508 | Завершен процесс 5(2,3), ВОП освободилась. Так как ресурсы освободились процесс 7(2,3) переходит из состояния очереди в состояние ввода. Загрузка ВОП и ОП. | 8+3-3=8 |  | 2 |
| 523 | Ресурсов хватает что бы начать выполнение процесса 7(3,2). Процессы 8(3,4) и 10(3,4) все еще находятся на выполнении. Ресурсы ими заняты. Процесс начинает освобождать ВОП. |  |  | 3 |
| 559 | Завершены процессы 8(3,4) и 10(3,4). Происходит освобождение ОП и ВОП. Так как ресурсы освободились начинается ввод 9(2,3). | 8+3+3-2=12 |  | 1 |
| 567 | Завершен процесс 7(2,3), ОП освободилась.  Процесс 9(2,3) находится на вводе. |  | 9 | 0 |
| 574 | Процесс 9(2,3) переходит из состояния ввода в состояние выполнения. Происходит освобождение ВОП. | 14 | 9+3=12 | 1 |
| 594 | Процесс 9(2,3) завершился. ВОП и ОП освобождены. Все процессы завершены. | 14+2=16 | 12 | 0 |

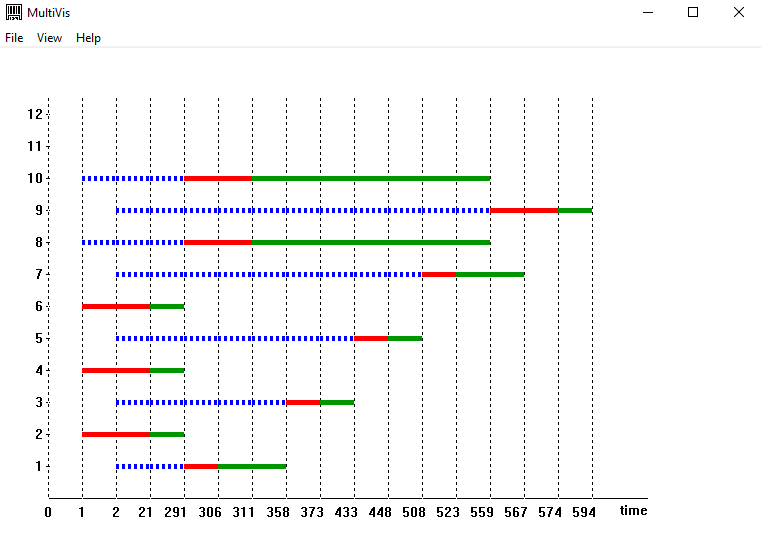


Рис. 3. Временная диаграмма выполнения алгоритма FIFO.

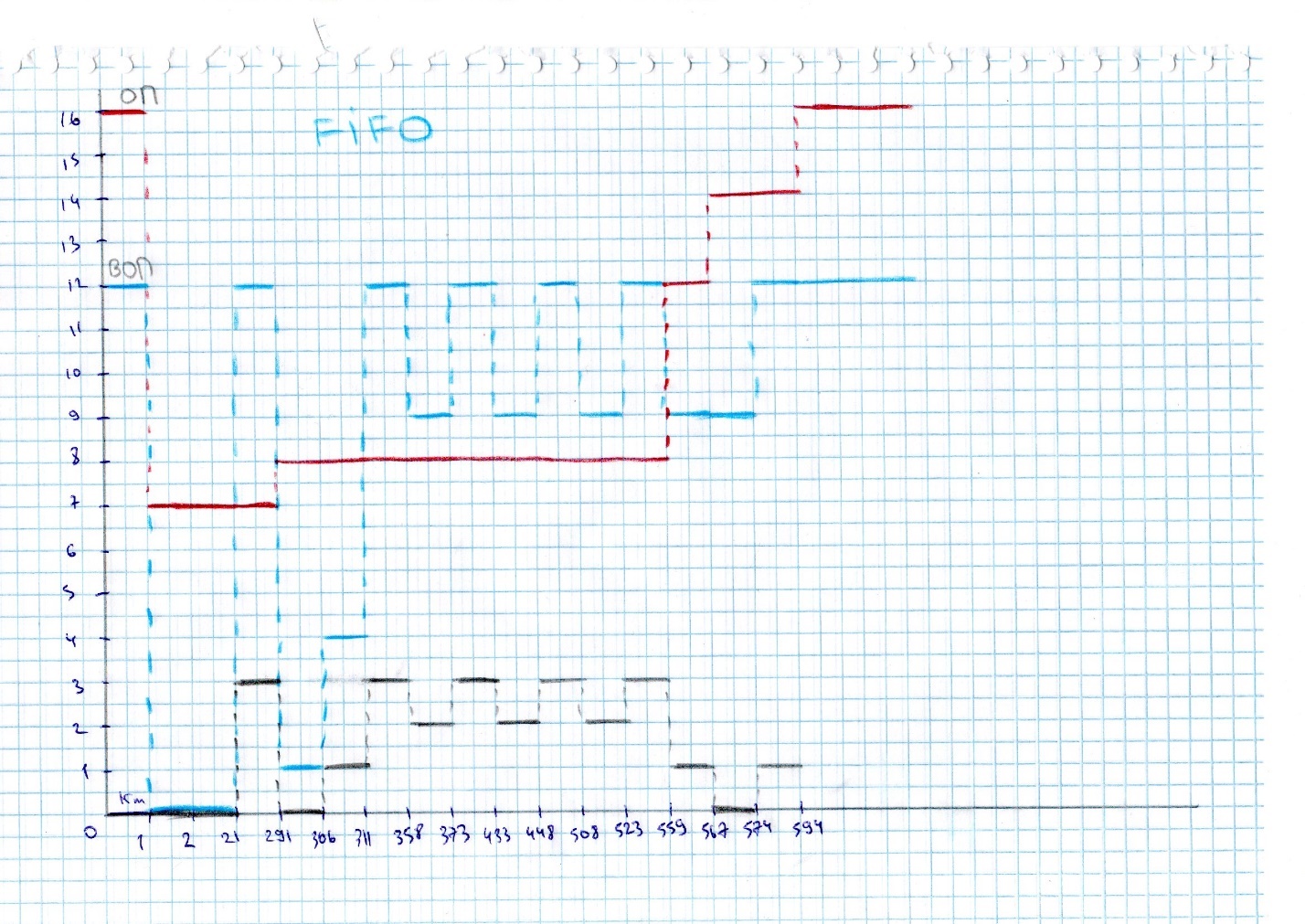


Рис. 4. Временная диаграмма использования ресурсов по алгоритму FIFO.

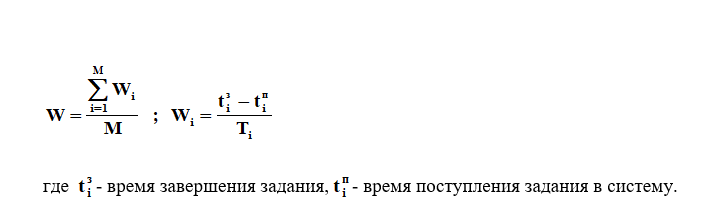
,

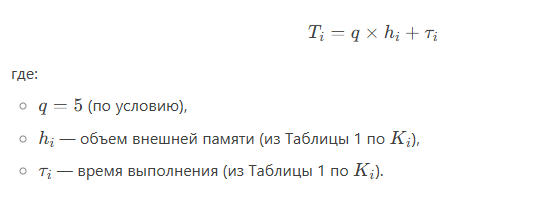
.

Трудоемкость обеспечена верно.

Таблица 4. Результаты работы ДО FIFO.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Время поступления | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 1 |
| Начало загрузки | 291 | 1 | 358 | 1 | 433 | 1 | 508 | 291 | 559 | 291 |
| Начало выполнения | 306 | 21 | 373 | 21 | 448 | 21 | 523 | 311 | 574 | 311 |
| Конец выполнения | 358 | 291 | 433 | 291 | 508 | 291 | 567 | 559 | 594 | 559 |
| Время нахождения в системе | 52 | 270 | 60 | 270 | 60 | 270 | 44 | 248 | 20 | 248 |
| Wi | 10,17 | 2,63 | 12,34 | 2,63 | 14,45 | 2,63 | 16,14 | 5,07 | 16,9 | 5,07 |





Максимальный коэффициент мультипрограммирования, равный 3, был получен на участках времени 21-291, 311-358, 373-433, 448-508, 523-559.

Средневзвешенное время обращения .

#### **ДО SJF**

Временная диаграмма мультипрограммной работы ЭВМ при исходных данных для ДО SJF (см. табл. 5) показана на рисунке 5.

Таблица 5. Последовательность событий в системе (SJF).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Время | Событие | Свободный объем ОП (16) | Свободный объем ВОП (12) |  |
| 0 | Запуск системы. Ничего не происходит. | 16 | 12 | 0 |
| 1 | Так как ресурсов хватает происходит ввод 2(3,4), 4(3,4), 6(3,4) процессов. Процессы 8(3,4) и 10(3,4) помещаются в очередь из-за недостатка ВОП. |  |  | 0 |
| 2 | Поступили остальные процессы 1(2,3), 3(2,3), 5(2,3), 7(2,3), 9(2,3). Так как не хватает ВОП они находятся в очереди. | 7 | 0 | 0 |
| 21 | Процессы 2(3,4), 4(3,4), 6(3,4) поступили на выполнение. Освобождение ресурсов ВОП. | 7 | 0+4+4+4=12 | 3 |
| 291 | Так как произошло освобождение ресурсов поступают новые процессы на ввод 1(2,3), 3(2,3), 5(2,3), 7(2,3). Ресурсы ВОП исчерпаны. | 7+3+3+3=16  16-2-2-2-2=8 | 12-3-3-3-3=0 | 0 |
| 306 | Завершен ввод заданий 1(2,3), 3(2,3), 5(2,3), 7(2,3) ВОП освободилась. Остальные процессы 8(3,4), 9(2,3), 10(3,4) стоят в очереди. | 8 |  | 3 |
| 386 | Завершены задания 1(2,3), 3(2,3), 5(2,3), 7(2,3), ОП освобождена. Из очереди выбираются все оставшиеся процессы так как на всех хватает ОП и ВОП ресурсов. | 8+2+2+2+2  -3-2-3=8 | -4-3-4=1 | 0 |
| 401 | Процесс 9(2,3) переходит из состояния ввода в состояние выполнения. Начинается освобождение ВОП. | 8 |  | 1 |
| 406 | Процессы 8(3,4) и 10(3,4) переходят из состояния ввода в состояние выполнения. Начинается освобождение ВОП. | 8 |  | 3 |
| 453 | Процесс 9(2,3) полностью завершен. Полное освобождение ВОП и ОП, которые занимал данный процесс. | 8+2 = 10 |  | 2 |
| 602 | Процессы 8(3,4) и 10(3,4) завершены. Происходит полное освобождение системы. Все процессы были успешно завершены. |  |  | 0 |

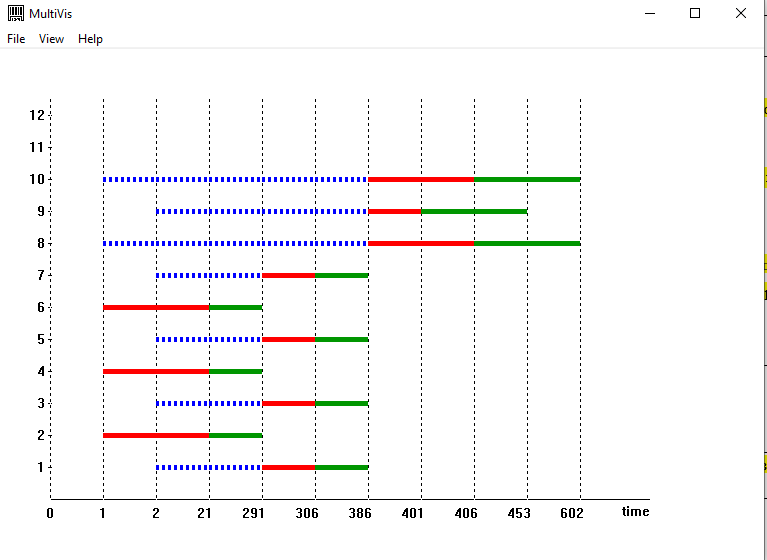


Рис. 5. Временная диаграмма выполнения алгоритма SJF.

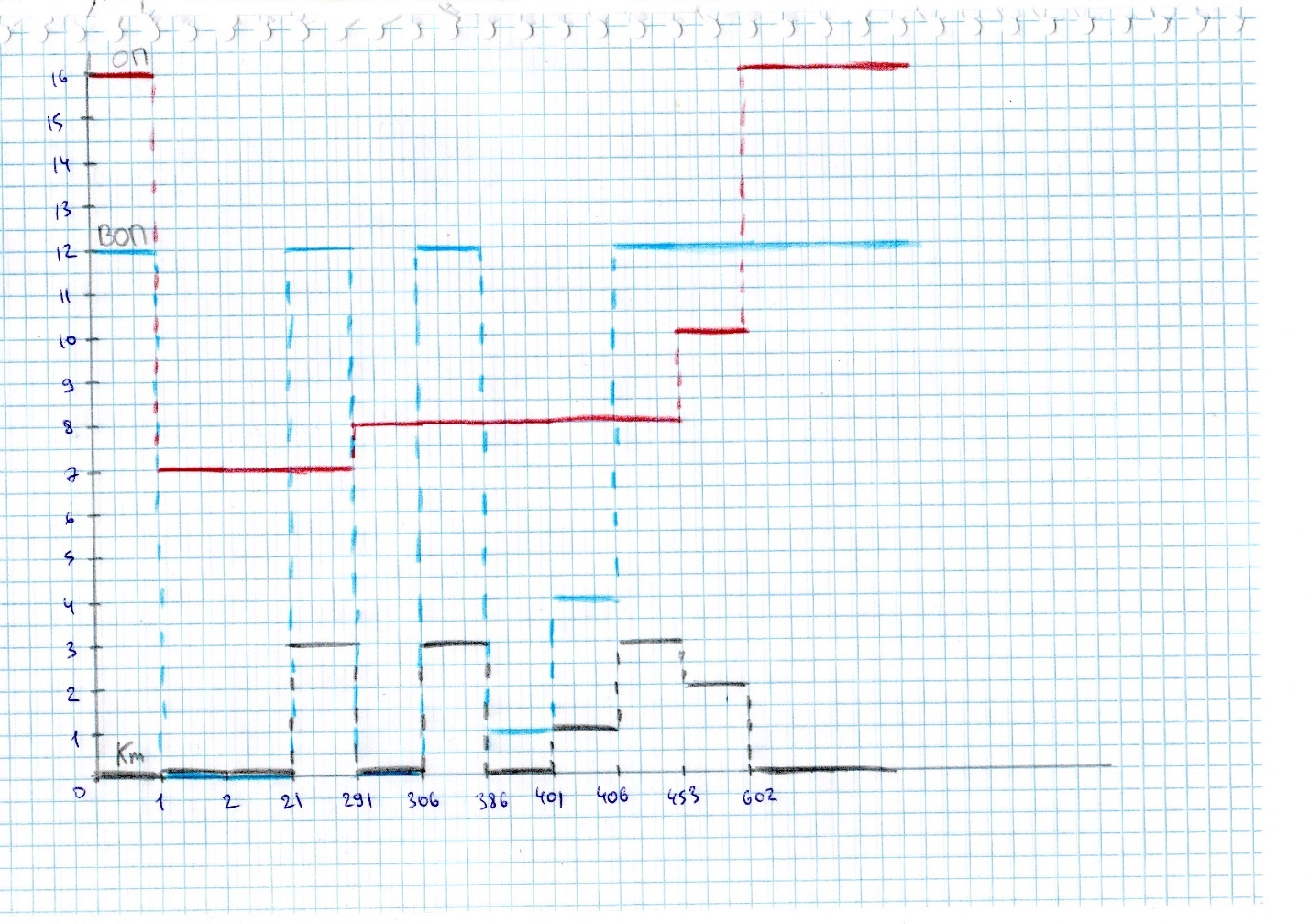
**

Рис. 6. Временная диаграмма использования ресурсов по алгоритму SJF.

.

Трудоемкость обеспечена верно.

Таблица 6. Результаты работы ДО SJF.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Время поступления | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 1 |
| Начало загрузки | 291 | 1 | 291 | 1 | 291 | 1 | 291 | 386 | 386 | 386 |
| Начало выполнения | 306 | 21 | 306 | 21 | 306 | 21 | 306 | 406 | 401 | 406 |
| Конец выполнения | 386 | 291 | 386 | 291 | 386 | 291 | 386 | 602 | 453 | 602 |
| Время нахождения в системе | 80 | 270 | 80 | 270 | 80 | 270 | 80 | 196 | 52 | 196 |
| Wi | 10,97 | 2,63 | 10,97 | 2,63 | 10,97 | 2,63 | 10,97 | 5,46 | 12,88 | 5,46 |

Максимальный коэффициент мультипрограммирования, равный 4, был получен на участке времени 306-386.

Средневзвешенное время обращения .

### 1.4 Вывод

На основе анализа алгоритмов планирования FIFO и SJF можно сделать следующие выводы:

1. Эффективность SJF: Алгоритм кратчайшей задачи (SJF) показал лучшие результаты по сравнению с FIFO. Средневзвешенное время обращения для SJF составило 7,5 , тогда как для FIFO — 8,8. Это связано с тем, что SJF минимизирует среднее время ожидания за счет приоритета коротких задач, что снижает простой системы и ускоряет обработку заданий.
2. Особенности FIFO: FIFO обеспечивает простоту реализации, но менее эффективен в условиях мультипрограммирования. Он не учитывает трудоемкость задач, что приводит к увеличению времени ожидания для коротких заданий, находящихся в очереди после длинных.
3. Мультипрограммирование: Максимальный коэффициент мультипрограммирования для FIFO достигал 3, а для SJF — 4, что указывает на более равномерное распределение ресурсов при использовании SJF.
4. Практические рекомендации: SJF предпочтителен для систем, где важна минимизация времени выполнения задач, однако он требует знания времени выполнения задач заранее, что может быть сложно реализовать. FIFO же целесообразен в простых системах с однородными задачами.

## **2. Раздел №2**

### 2.1 Задание

Разработать структуру функционирования диспетчера работ в вычислительной системе, заданной в разделе 1, на всём интервале работы. Квант времени, выделяемый каждой работе, выбирается исходя из конкретной ситуации: число работ, параллельно занимающих процессор, интервалы времени с коэффициентом многозадачности, дисциплины обслуживания.

Диспетчер использует метод разделения времени в сочетании с приоритетами.

### 2.2 Исходные данные

Вариант 8.

БП – 2) LIFO, П – 4) SJF.

### 2.3 Временные диаграммы работы LIFO и SJF. Таблицы трассировки, расчет

#### **ДО LIFO**

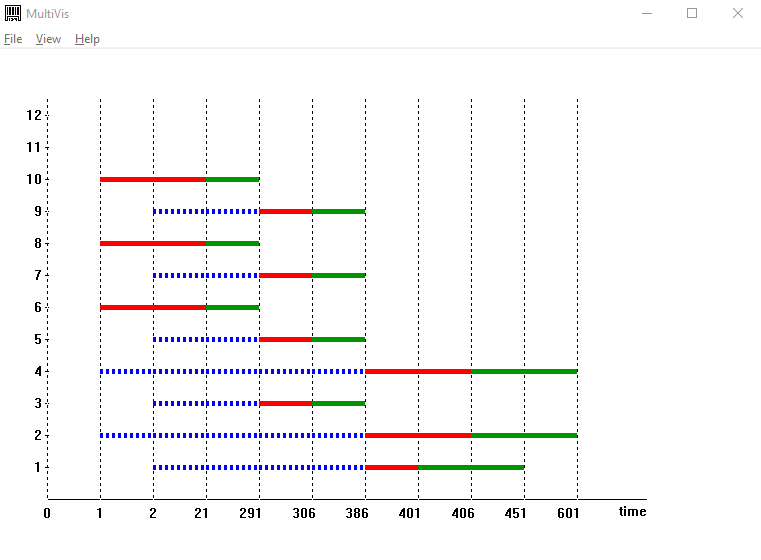


Рис. 7. Исходная ВД для ДО LIFO.

*Расчет:*

Временной диапазон: 21-291

,

.

Задания 6, 8, 10 получают по Δt CPU = 90

, задания выполнены.

Временной диапазон: 306-386

,

.

Задания 3,5,7,9 получают по Δt CPU = 20

. задания выполнены.

Временной диапазон: 401-406

,

.

Задание 1 получает Δt CPU = 5

,

Временной диапазон: 406-451

,

.

Задания 1,2,4 получают по Δt CPU = 15

, задание выполнено.

,

Временной диапазон: 451-601

,

.

Задания 2,4 получают Δt CPU = 75

, задания выполнены.

Таблица 7. Трассировка планировщика для ДО LIFO.

|  |  |  |
| --- | --- | --- |
| Время | Событие |  |
| 0 | ---------------- | 0 |
| 1 | ------------------ | 0 |
| 2 | ------------------ | 0 |
| 21 | Выполняются задания 6, 8, 10 | 3 |
| 291 | Завершение выполнения заданий 6, 8, 10 | 0 |
| 306 | Выполняются задания 3, 5, 7, 9 | 4 |
| 386 | Завершение выполнения заданий 3, 5, 7, 9 | 0 |
| 401 | Выполняется задание 1 | 1 |
| 406 | Выполняются задания 1, 2, 4 | 3 |
| 451 | Завершение выполнения задания 1  Выполняются задания 2, 4 | 2 |
| 601 | Завершение выполнения заданий 2, 4 | 0 |

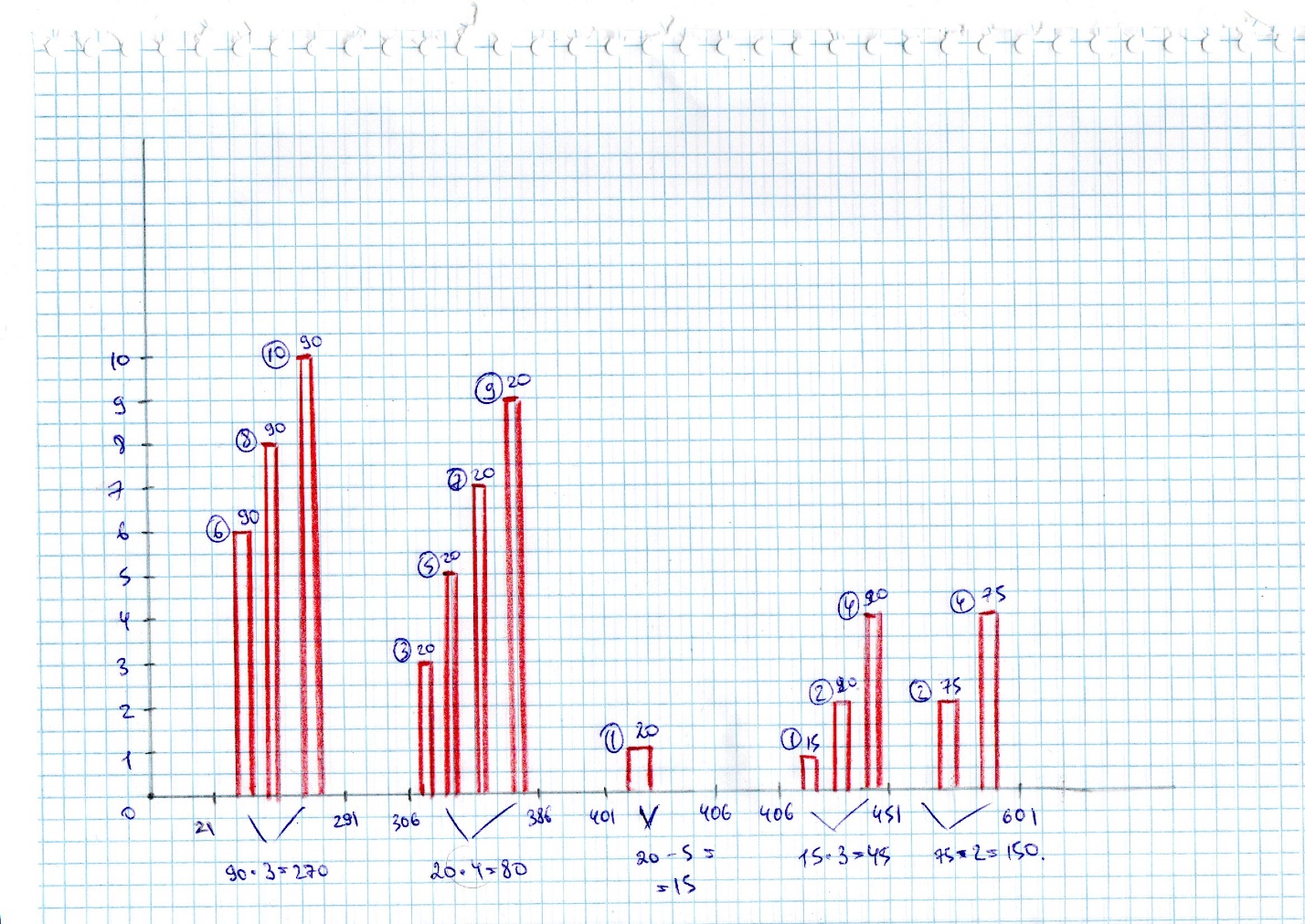
**

Рис. 8. Временная диаграмма работы ДО LIFO.

#### **ДО SJF**

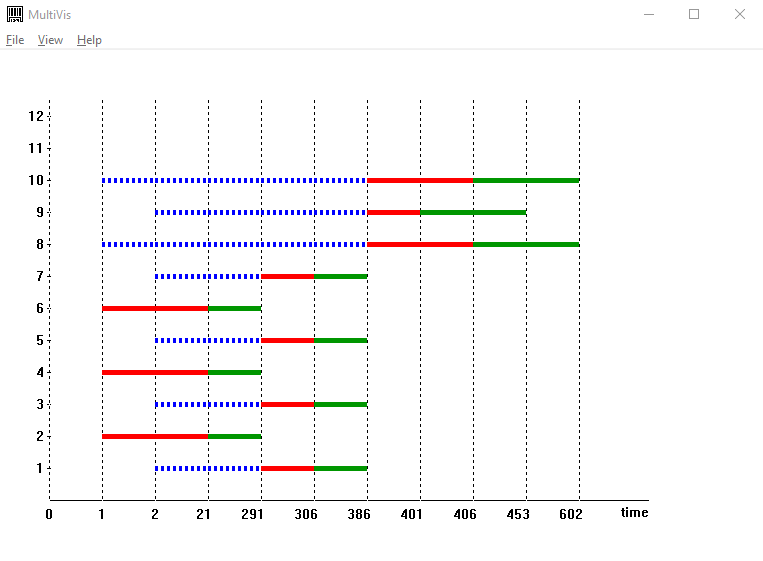


Рис. 9. Исходная ВД для ДО SJF.

*Расчет:*

Временной диапазон: 21-291

,

.

Задания 2, 4, 6 получают по Δt CPU = 90

, задания выполнены.

Временной диапазон: 306-386

,

.

Задания 1,3,5,7 получают по Δt CPU = 20

. задания выполнены.

Временной диапазон: 401-406

,

.

Задание 9 получает Δt CPU = 5

,

Временной диапазон: 406-453

,

.

Задания 8,9,10 получают по Δt CPU = 15,6

, задание выполнено.

,

Временной диапазон: 453-601

,

.

Задания 8,10 получают Δt CPU = .

, задания выполнены.

Таблица 8. Трассировка планировщика для ДО SJF.

|  |  |  |
| --- | --- | --- |
| Время | Событие |  |
| 0 | ---------------- | 0 |
| 1 | ------------------ | 0 |
| 2 | ------------------ | 0 |
| 21 | Выполняются задания 2, 4, 6 | 3 |
| 291 | Завершение выполнения заданий 2, 4, 6 | 0 |
| 306 | Выполняются задания 1, 3, 5, 7 | 4 |
| 386 | Завершение выполнения заданий 1, 3, 5, 7 | 0 |
| 401 | Выполняется задание 9 | 1 |
| 406 | Выполняются задания 8, 9, 10 | 3 |
| 451 | Завершение выполнения задания 9  Выполняются задания 8, 10 | 2 |
| 601 | Завершение выполнения заданий 8, 10 | 0 |

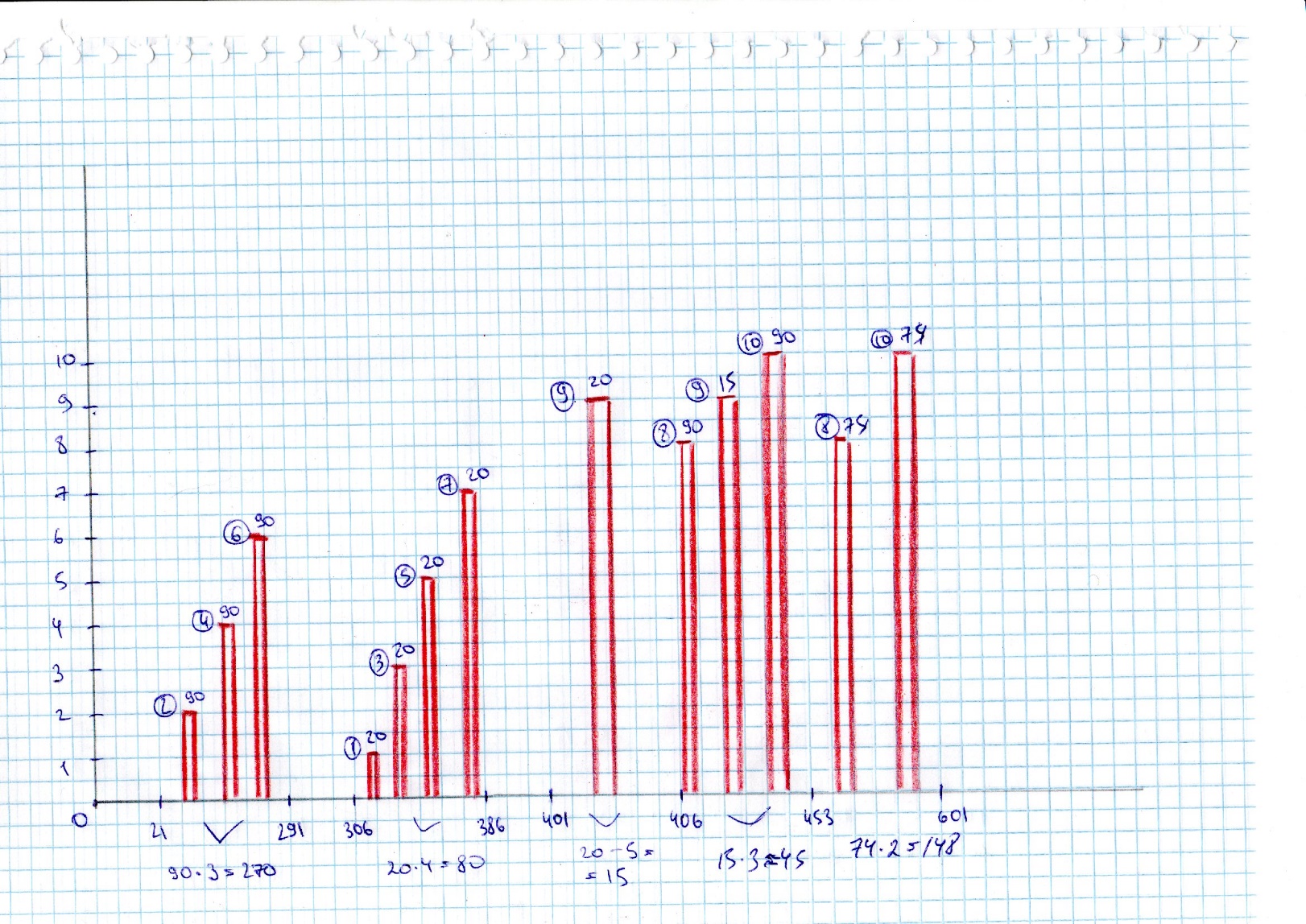


Рис. 10. Временная диаграмма работы ДО SJF.

### 2.4 Выводы

Анализ диспетчеризации с использованием алгоритмов LIFO и SJF позволяет сформулировать следующие выводы:

1. Особенности LIFO: Алгоритм LIFO («последним пришел — первым ушел») демонстрирует высокую эффективность для задач, которые поступают позже, так как они получают приоритет. Однако это может приводить к голоданию ранних задач, особенно если новые задания поступают часто. В данном случае LIFO показал среднее время выполнения, но с риском несправедливого распределения ресурсов.
2. Преимущества SJF в диспетчеризации: SJF в режиме разделения времени обеспечил более равномерное распределение процессорного времени. Короткие задачи выполнялись быстрее, что снизило общее время нахождения заданий в системе. Например, задание №9 (время выполнения 20 ед.) завершилось раньше, чем длинные задачи, что подтверждает эффективность SJF.
3. Влияние квантового времени: Использование квантового времени (RR-подобный подход) позволило комбинировать приоритеты и справедливость распределения ресурсов. Это особенно важно в мультипрограммных системах, где необходимо балансировать между временем ответа и загрузкой процессора.
4. Рекомендации: SJF предпочтителен для минимизации среднего времени выполнения задач, однако его реализация требует точного прогнозирования трудоемкости задач. LIFO может быть эффективен в специфических сценариях, например, для обработки краткосрочных фоновых процессов, но требует дополнительных мер против голодания.

Оба раздела подтверждают, что выбор алгоритма планирования и диспетчеризации должен учитывать специфику задач системы, требования к производительности и справедливости распределения ресурсов.

# **Список источников**

1. Коршикова, Л. А. Операционные системы как системы управления вычислительными ресурсами: учеб.-метод. пособие / Л. А. Коршикова. — Новосибирск: Изд-во НГТУ, 2007. — 88 с.
2. Коршикова, Л. А. Качество программных средств и их оценка / Л. А. Коршикова // Модельное исследование методов, алгоритмов и средств индуктивного анализа данных в приоритетных отраслях: монография. — Москва: Русайнс, 2020. — С. 82–99. — ISBN 978-5-4365-5376-4.
3. Планирование верхнего уровня управления заданиями. Методическое указание к лабораторной работе по курсу СПО (ОВП). № 336.13 — НЭТИ, 1990 г.
4. Таненбаум, Э. Современные операционные системы / Э. Таненбаум. — СПб.: Питер, 2004. — 1040 с. — ISBN 5-318-00474-7.
5. Столлингс, В. Операционные системы: внутреннее устройство и принципы функционирования / В. Столлингс. — М.: Вильямс, 2002. — 992 с. — ISBN 5-8459-0252-1.