

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ

«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

О. В. КАРАВАЕВА

**Операционные системы. Управление памятью и
процессами**

Учебно-методическое пособие

Киров
2016

УДК 004.451(07)
К 21

Рекомендовано к изданию методическим советом факультета автоматизации и вычислительной техники ФГБОУ ВО «ВятГУ» в качестве учебно-методического пособия для студентов направлений 09.03.01 «Информатика и вычислительная техника» и 09.03.04 «Программная инженерия» всех профилей подготовки и всех форм обучения.

Рецензенты

доцент кафедры математических и естественно-научных дисциплин КФ
МФЮА, кандидат технических наук Т.А. Анисимова

доцент кафедры АТ ФГБОУ ВО «ВятГУ»,
кандидат технических наук,
В.Вал. Куклин

Караваева О.В.

К 21 Операционные системы. Управление памятью и процессами:
учебно-методическое пособие / О.В.Караваева – Киров: ФГБОУ
ВО «ВятГУ», 2016. – 39с.

УДК 004.451(07)

Пособие предназначено для студентов направлений 09.03.01 «Информатика и вычислительная техника» и 09.03.04 «Программная инженерия» всех профилей подготовки, всех форм обучения для выполнения лабораторных работ по дисциплине «Операционные системы».

Тех. редактор

© ФГБОУ ВО «ВятГУ», 2016

Учебное издание

Караваева Ольга Владимировна

**Операционные системы. Управление памятью и
процессами**

Учебно-методическое пособие

Подписано к использованию

Подписано в печать

Оглавление

| | |
|---|----|
| Введение..... | 5 |
| 1 Управление памятью..... | 6 |
| 1.1 Интерфейс и функции окна диспетчера памяти..... | 6 |
| 1.2 Алгоритм работы диспетчера памяти..... | 9 |
| 1.3 Алгоритм выбора свободного блока..... | 15 |
| 2 Планирование и диспетчеризация процессов | 17 |
| 2.1 Интерфейс и функции окна диспетчера процессов | 18 |
| 2.2 Алгоритм работы диспетчера задач..... | 21 |
| 2.3 Невытесняющие дисциплины планирования | 25 |
| 2.3.1 Дисциплина FCFS | 25 |
| 2.3.2 Дисциплины SJN, SRT | 27 |
| 2.4 Вытесняющие дисциплины диспетчеризации..... | 29 |
| 2.4.1 Дисциплина RR | 29 |
| 2.4.2 Планирование процессов в ОС UNIX | 29 |
| 2.4.3 Планирование процессов в ОС Windows NT | 33 |
| 2.4.4 Планирование процессов в OS/2..... | 35 |

Введение

В данном методическом пособии рассматривается управление двумя основными ресурсами операционной системы: процессорным временем и оперативной памятью.

Процессор является одним из самых необходимых ресурсов для выполнения вычислений. Поэтому способы распределения времени центрального процессора между выполняющимися задачами сильно влияют и на скорость выполнения отдельных вычислений, и на общую эффективность вычислительной системы.

Оперативная память также является важнейшим ресурсом любой вычислительной системы, поэтому от выбранных механизмов распределения памяти между выполняющимися процессорами очень сильно зависит эффективность использования ресурсов системы, ее производительность и возможности, которыми могут пользоваться программисты при создании своих программ.

В процессе выполнения работы студент по запросу, отображаемому в окне программной модели должен выполнить действия, соответствующие действиям ОС в аналогичной ситуации. В разделе, посвященном изучению работы диспетчера памяти нужно выполнить сорок шагов.

1 Управление памятью

Функциями ОС по управлению памятью в мультипрограммных системах являются:

- отслеживание (учет) свободной и занятой памяти;
- первоначальное и динамическое выделение памяти процессам приложений и самой операционной системе и освобождение памяти по завершении процессов;
- настройка адресов программы на конкретную область физической памяти;
- полное или частичное вытеснение кодов и данных процессов из ОП на диск, когда размеры ОП недостаточны для размещения всех процессов, и возвращение их в ОП;
- защита памяти, выделенной процессу, от возможных вмешательств со стороны других процессов;
- дефрагментация памяти [1].

Процесс может быть загружен в раздел равного или большего размера. В лабораторной работе реализовано распределение памяти динамическими разделами с использованием одного из трех алгоритмов выделения оперативной памяти.

1.1 Интерфейс и функции окна диспетчера памяти

Вид окна диспетчера памяти приведен на рисунке 1.1.

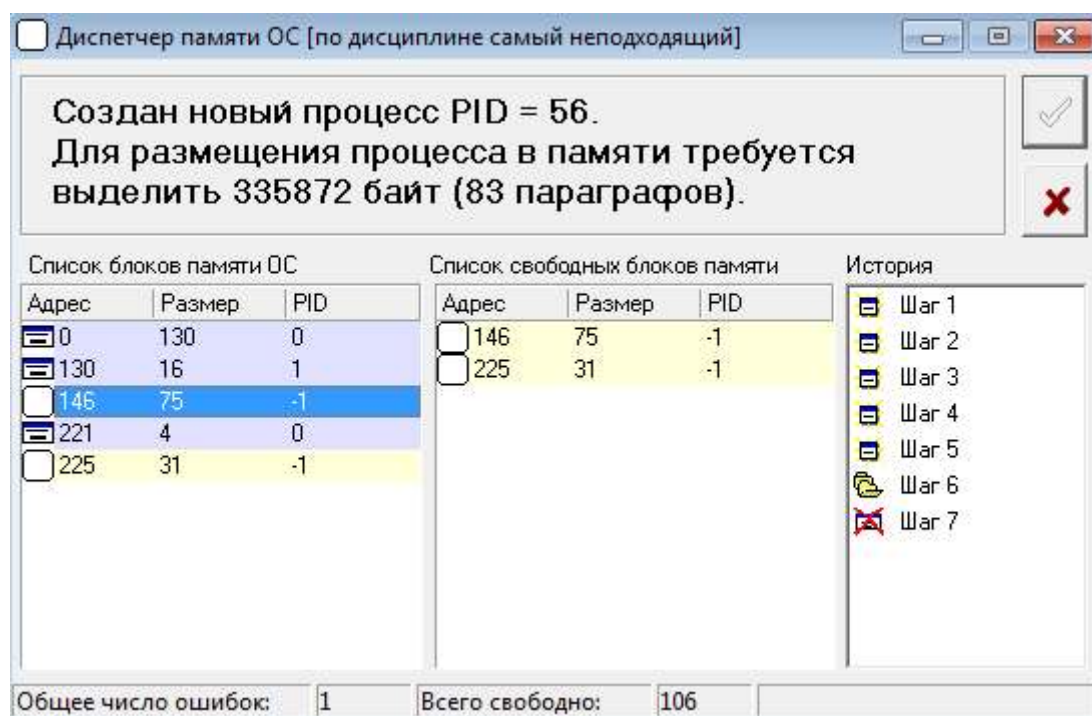


Рис. 1.1 Диспетчер памяти

В верхней части окна присутствует сообщение о наступившем событии. Задача пользователя – корректно на него отреагировать.

В левой части окна находится список блоков памяти. Занятые блоки помечены голубым цветом, свободные – желтым. Для каждого блока указан его начальный адрес, размер (в параграфах), а также идентификатор процесса, владеющего данным блоком (-1 – нет владельца).

Пользователь имеет возможность выполнить следующие действия:

- ответить на запрос отказом (нижняя кнопка);
- выделить память процессу. Для этого необходимо щелкнуть правой кнопкой мыши на свободном блоке в списке блоков памяти ОС и выбрать пункт «Выделить приложению». После этого в появившемся окне нужно ввести идентификатор процесса, которому выделяется память, а также количество выделяемых параграфов. Выполнение данного действия представлено на рисунках 1.2 и 1.4;
- освободить память. Для этого необходимо щелкнуть правой кнопкой мыши на занятом блоке в списке блоков памяти ОС и выбрать пункт «Освободить». Выполнение данного действия представлено на рисунке 1.5;

- объединить два свободных блока в один. Для этого необходимо щелкнуть правой кнопкой мыши на свободном блоке в списке блоков памяти ОС и выбрать пункт «Объединить со следующим», Данное действие необходимо выполнять каждый раз, когда образуются соседние свободные блоки;

- уплотнить (дефрагментировать память). Для этого необходимо щелкнуть правой кнопкой мыши на списке блоков памяти ОС и выбрать пункт «Уплотнить память»;

- подтвердить действия, щелкнув по верхней кнопке.

В средней части окна находится список свободных блоков памяти, который можно отсортировать.

Для сортировки необходимо зажать левую клавишу мыши на одном из свободных блоков и перетащить его на нужное место.

Если при щелчке на заголовке столбца удерживать нажатой клавишу «Shift» признак сортировки в остальных столбцах списка не сбрасывается.

Для отмены сортировки по столбцу необходимо удерживать клавишу «Ctrl» при щелчке на заголовке столбца.

Изменение ширины столбца осуществляется перетаскиванием границы его заголовка.

Чтобы поменять порядок столбцов, необходимо перетащить на соответствующее место заголовков требуемого столбца.

В правой части экрана находится список выполненных шагов:

В окнах диспетчеров памяти и задач присутствуют списки шагов, которые хранят историю действий пользователя.

Элемент в списке помечается красным цветом, если действия пользователя на данном шаге были некорректны.

При наведении указателя мыши на элемент списка отображается информация о событии и действиях пользователя на данном шаге.

Для отмены произвольного числа шагов необходимо щелкнуть мышью на элемент списка, до которого необходимо очистить историю. Отмененные шаги помечаются серым цветом.

До тех пор, пока не выполнено какое-либо действие в окне диспетчера, можно восстановить действия пользователя, щелкнув на записи, соответствующей отмененному шагу.

В нижней части окна расположена строка статуса, в которой отображается общее число ошибок, которые допустил пользователь в процессе выполнения лабораторной работы, а также количество свободных параграфов оперативной памяти.

По достижении заданного числа шагов выводится сообщение о выполнении лабораторной работы, а также количество неисправленных ошибок.

1.2 Алгоритм работы диспетчера памяти

Диспетчер памяти обрабатывает следующие события:

- создание нового процесса;

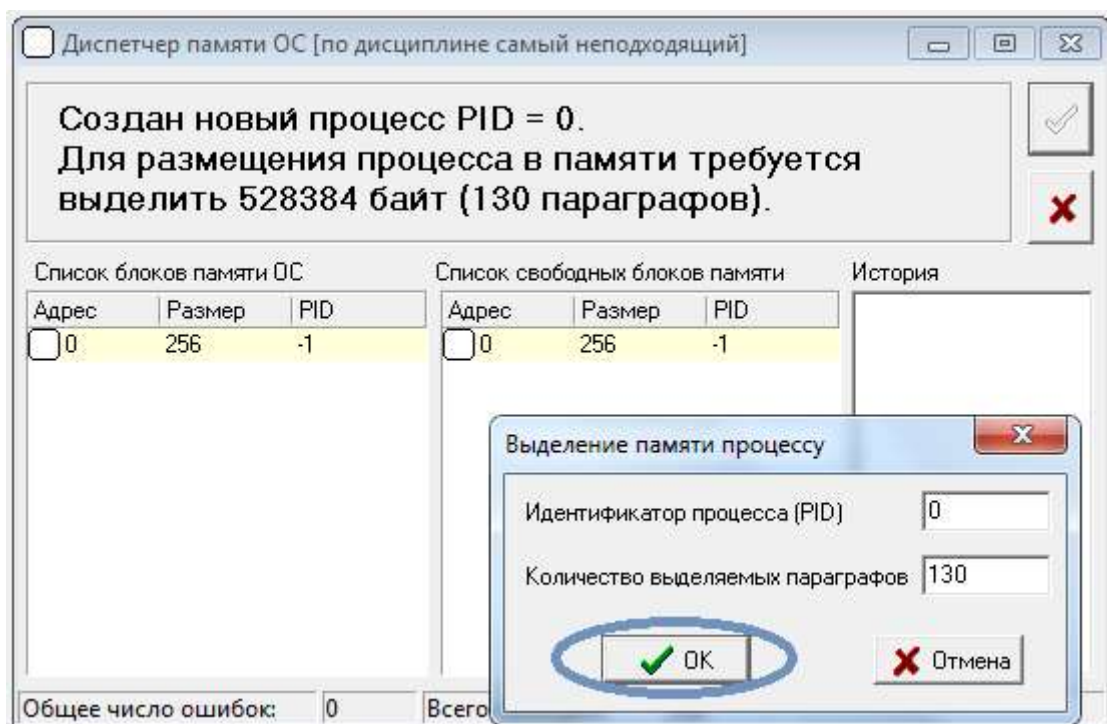


Рис. 1.2 Создание процесса

- завершение процесса;

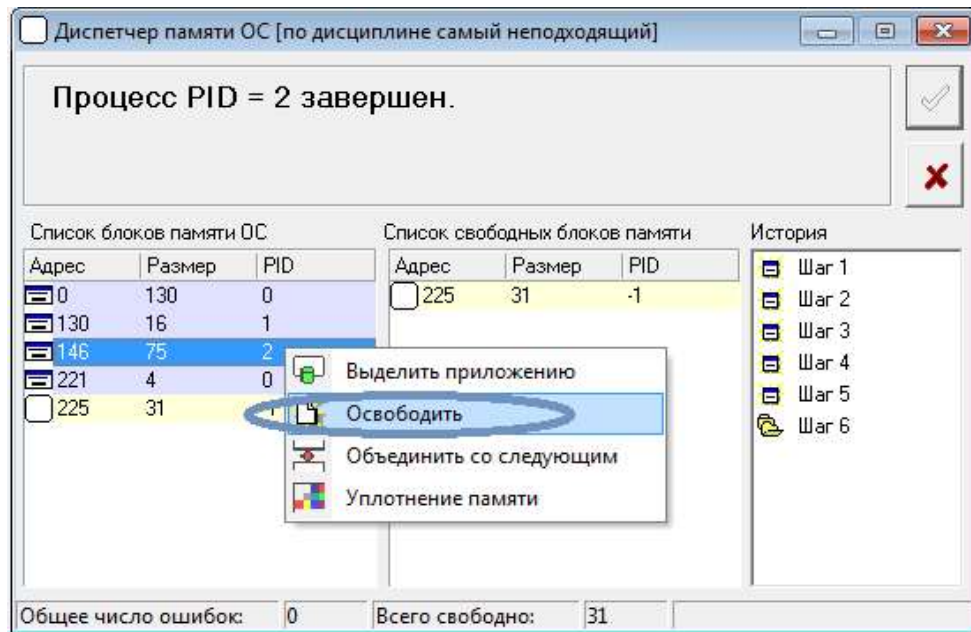


Рис. 1.3 Завершение процесса

- выделение памяти существующему процессу;

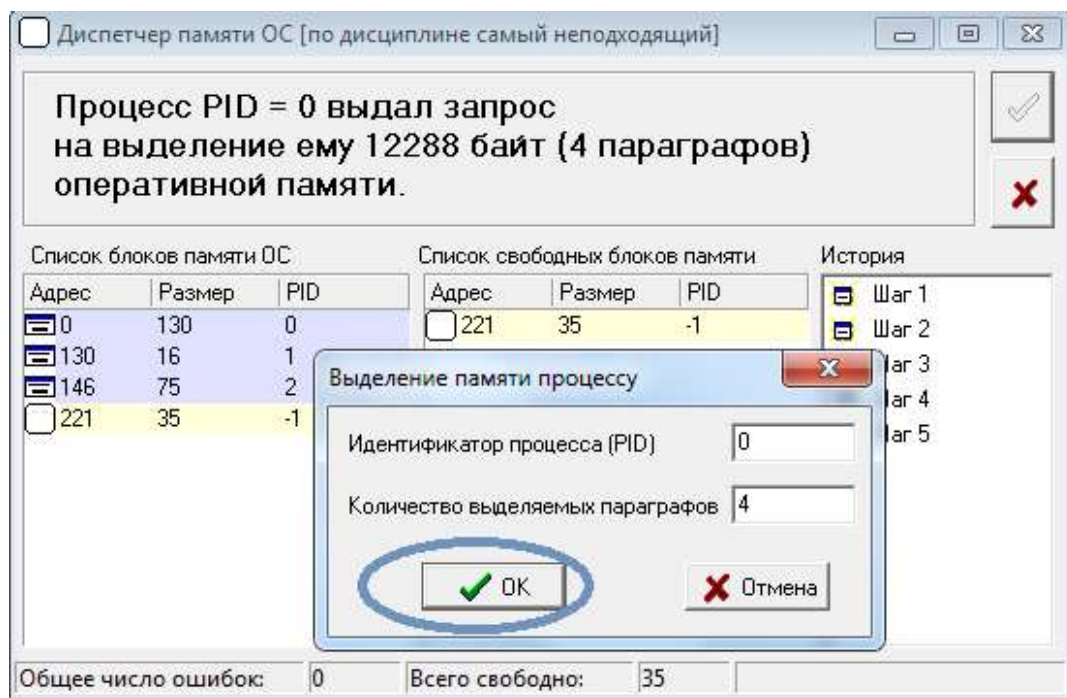


Рис. 1.4 Выделение памяти

- освобождение памяти, которой владеет процесс.

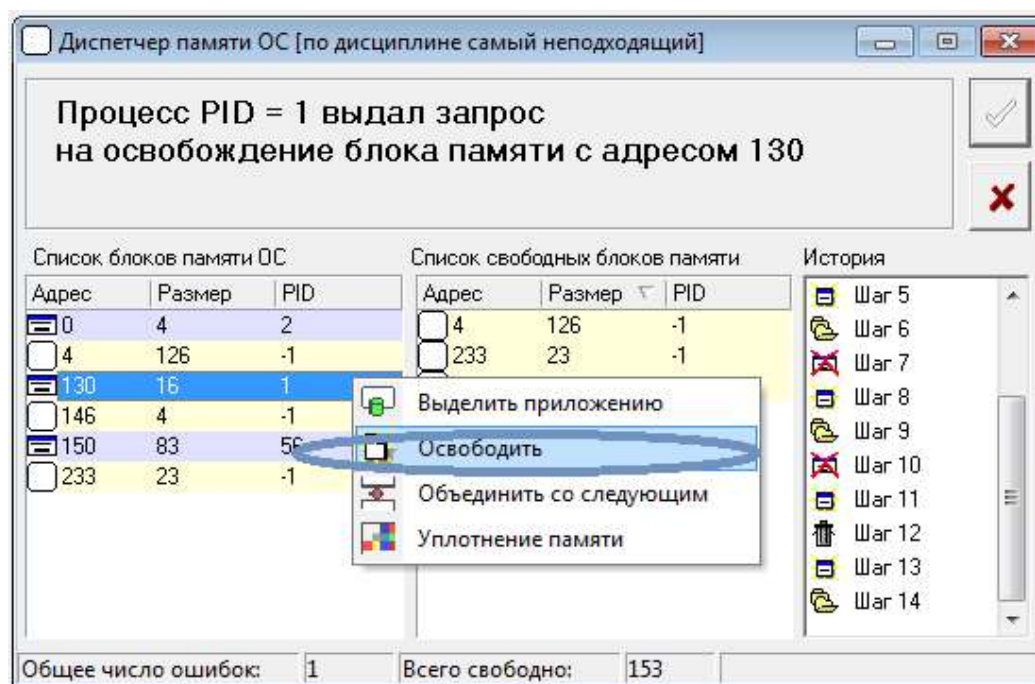


Рис. 1.5 Освобождение памяти

Пользователь, работающий в качестве диспетчера памяти должен корректно отреагировать на данные события:

- при создании нового процесса необходимо проверить корректность идентификатора процесса (он должен находиться в заданном диапазоне и, кроме того, не должен существовать процесс с таким же идентификатором), как показано на рисунке 1.6;

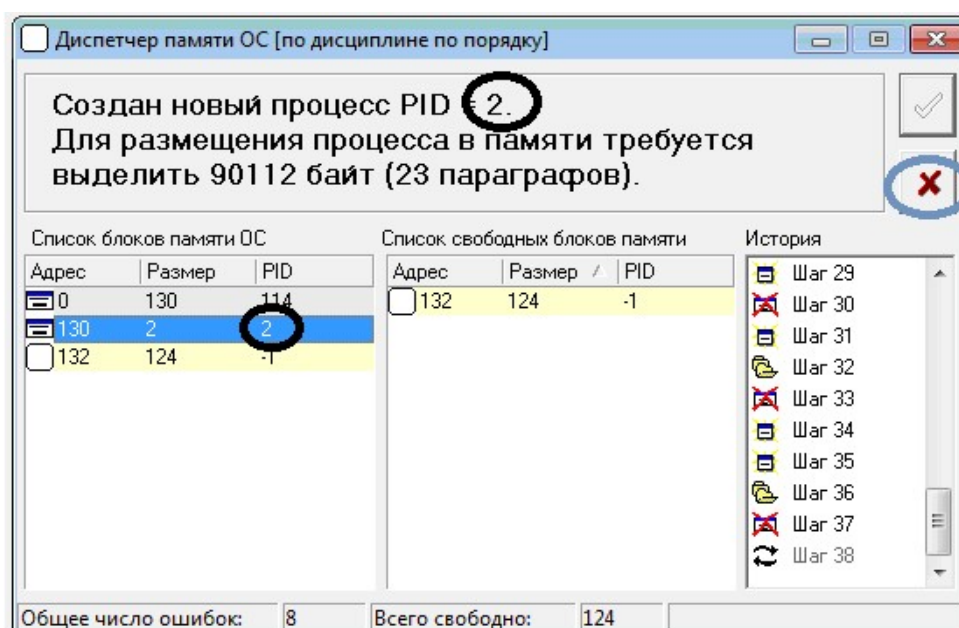


Рис. 1.6 Создание существующего процесса

- для удаления процесса, выделения памяти существующему процессу и освобождение памяти, которой владеет процесс данный процесс должен существовать (рис. 1.7-1.8);

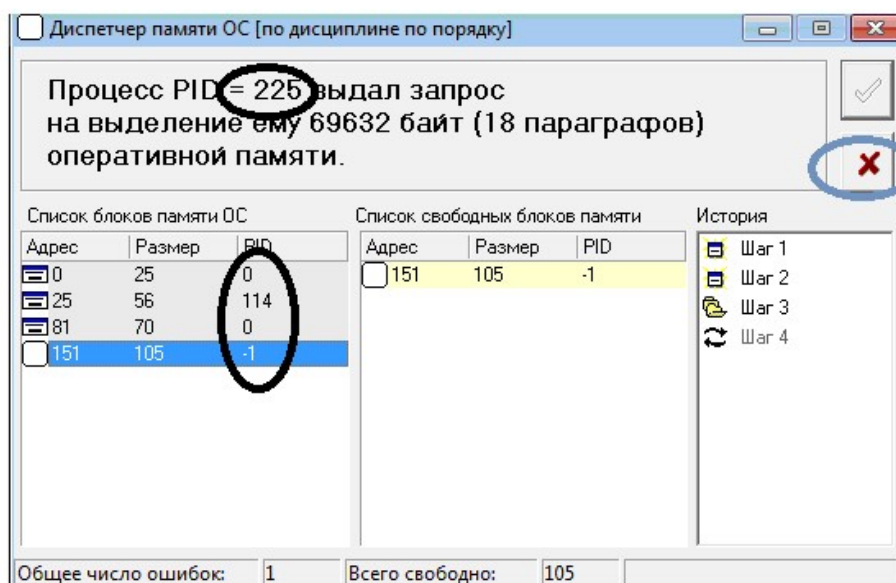


Рис. 1.7 Выделение памяти не существующему процессу

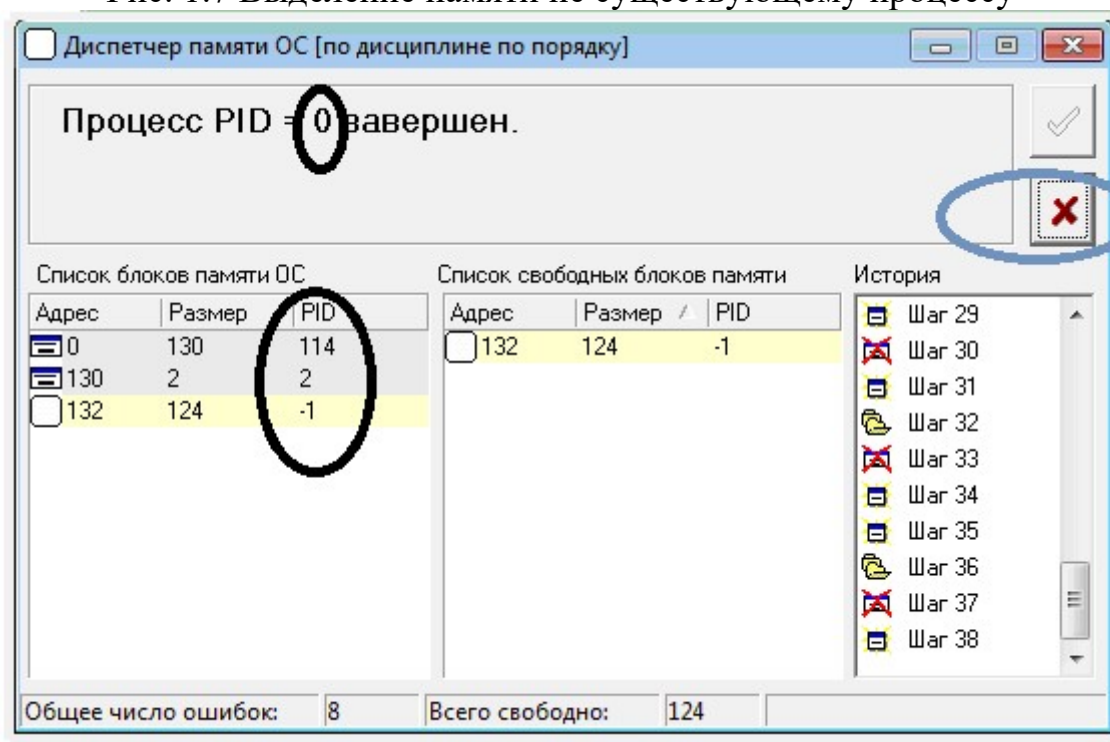


Рис. 1.8 Завершение не существующего процесса

- для освобождения памяти, которой владеет процесс, необходимо проверить, владеет ли указанный процесс блоком памяти, который он освобождает, как показано на рисунке 1.9;

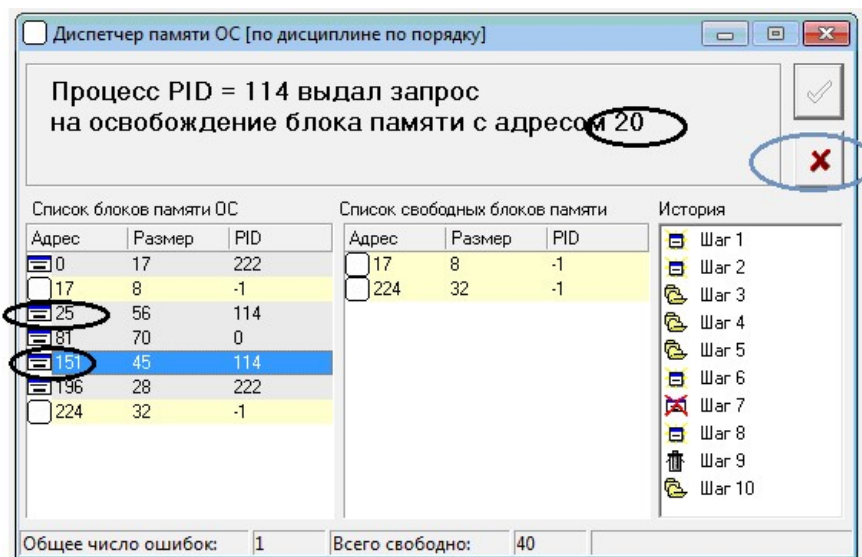


Рис. 1.9 Освобождения блока с неверным адресом

- возможна ситуация, когда в памяти не существует свободного блока нужного размера. В этом случае, если суммарный размер свободных блоков больше или равен требуемому размеру, необходимо выполнить уплотнение (дефрагментацию) памяти. Выполнение данного действия представлено на рисунке 1.10;

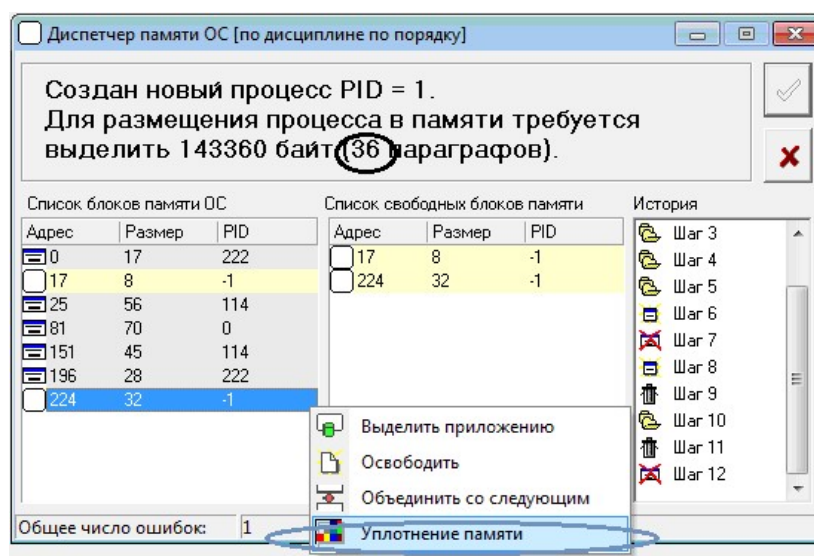


Рис. 1.10 Уплотнение памяти

- в любом случае, если суммарный размер свободной памяти недостаточен, диспетчер памяти выдает отказ на поступивший запрос (рис. 1.11);

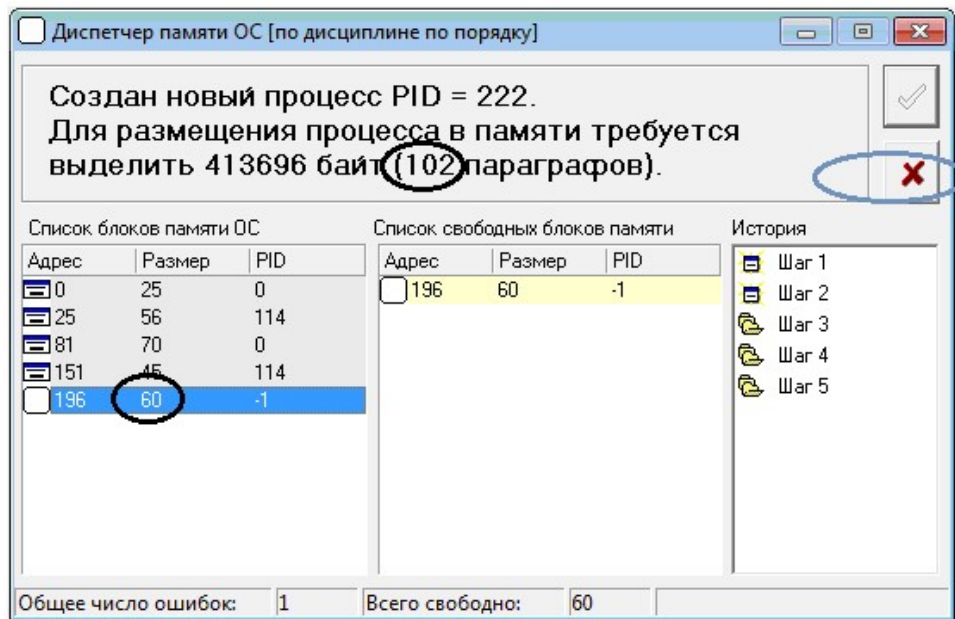


Рис. 1.11 Недостаточность свободной памяти

- при освобождении блока необходимо проверить: нет ли перед ним или после него свободных блоков. Если есть, то все соседние блоки необходимо объединить в один с целью уменьшения фрагментации памяти. Выполнение данного действия представлено на рисунке 1.12;

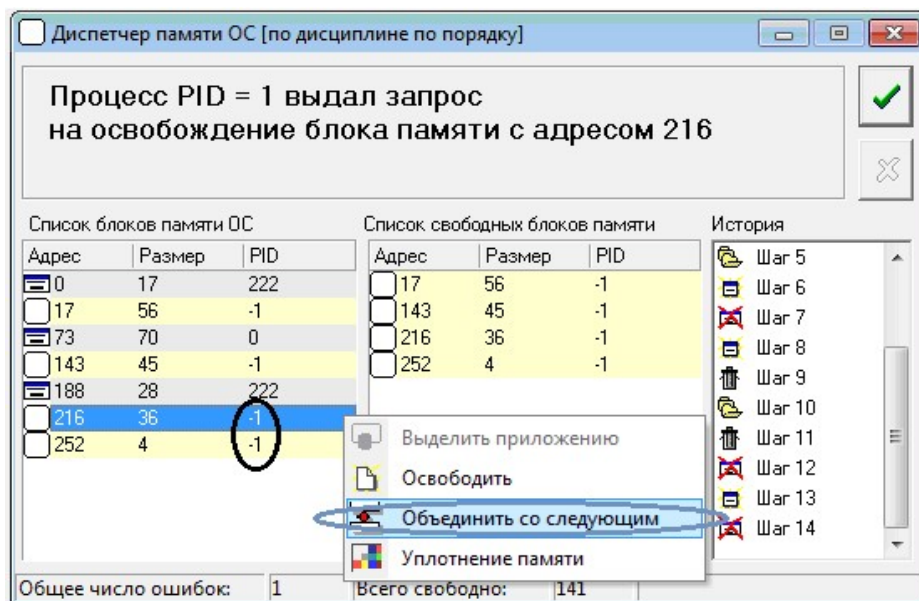


Рис. 1.12 Объединение свободных соседних блоков

- так же стоит обратить внимание, что процесс нельзя поместить в свободный блок, который содержит столько же параграфов, сколько запрашивает данный процесс (рис. 1.13-1.14);

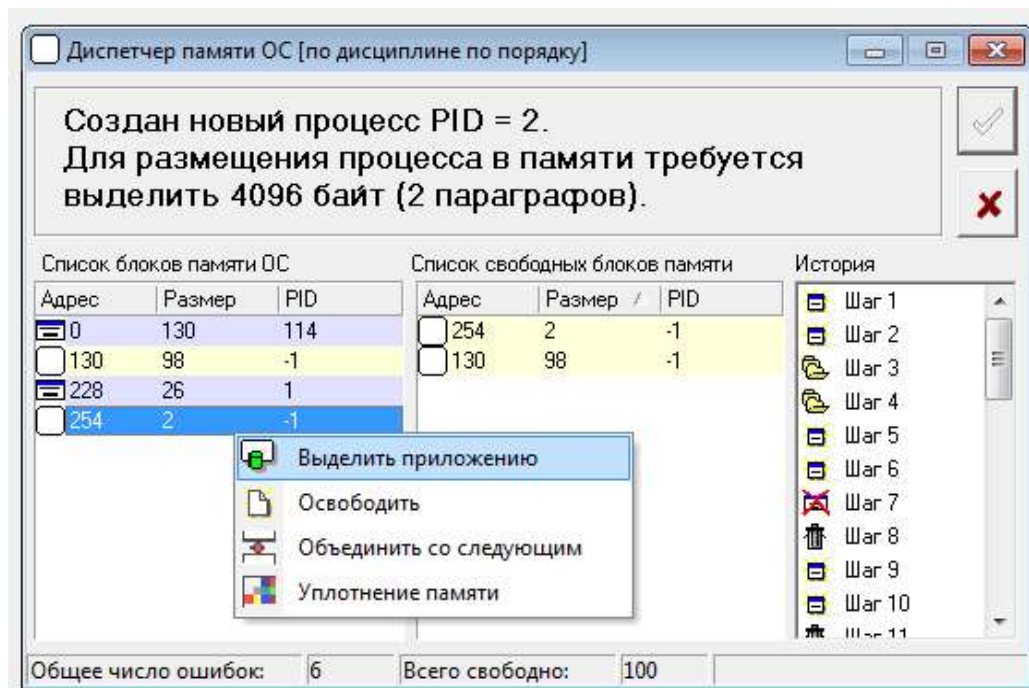


Рис. 1.13 Создание процесса с равным количеством параграфов в свободном блоке.

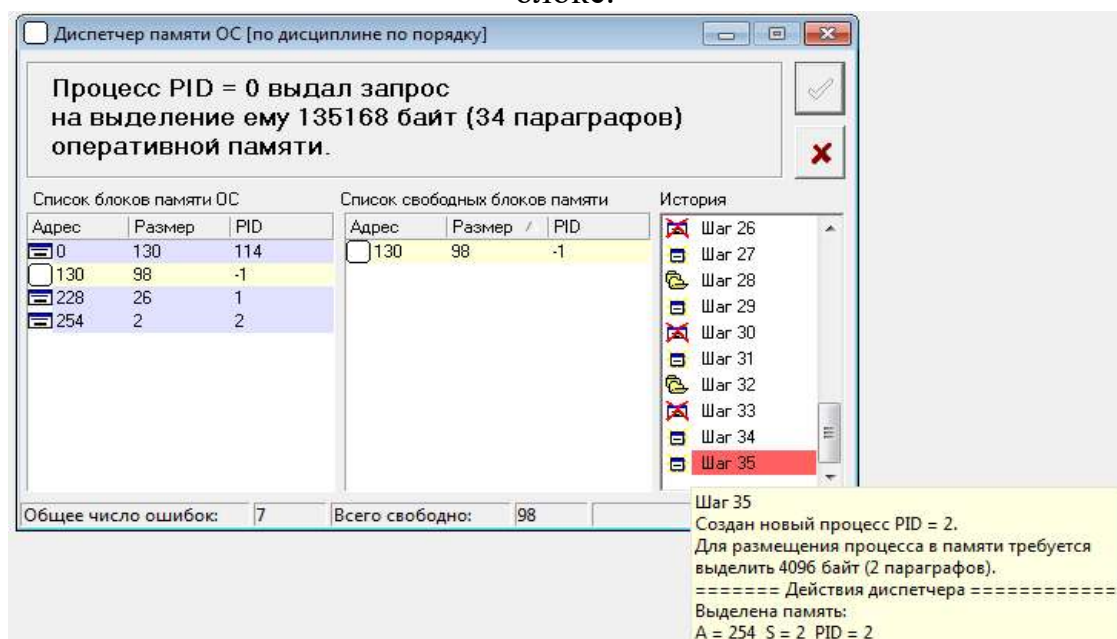


Рис. 1.14 Создание процесса с равным количеством параграфов в свободном блоке.

1.3 Алгоритм выбора свободного блока

При выделении памяти, как правило, приходится делать выбор между несколькими свободными блоками. Существует три стратегии, позволяющие сделать такой выбор (стратегия указана в задании):

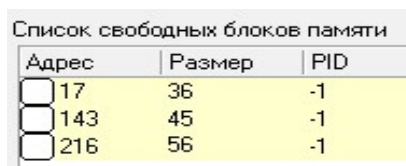
- выбор первого подходящего блока. В этом случае список свободных блоков сортируется по адресу и выбирается первый блок, размер которого больше или равен требуемому размеру, как показано на рисунке 1.15;



| Адрес | Размер | PID |
|--|--------|-----|
| <input checked="" type="checkbox"/> 17 | 56 | -1 |
| <input type="checkbox"/> 143 | 45 | -1 |
| <input type="checkbox"/> 216 | 36 | -1 |
| <input type="checkbox"/> 252 | 4 | -1 |

Рис. 1.15 Первый подходящий блок

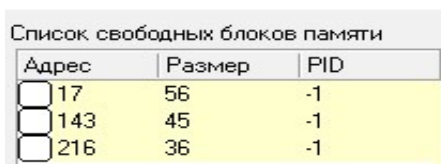
- выбор самого подходящего блока. Список свободных блоков сортируется по размеру (от меньшего к большему) и выбирается первый подходящий по размеру блок, как показано на рисунке 1.16;



| Адрес | Размер | PID |
|--|--------|-----|
| <input checked="" type="checkbox"/> 17 | 36 | -1 |
| <input type="checkbox"/> 143 | 45 | -1 |
| <input type="checkbox"/> 216 | 56 | -1 |

Рис: 1.16 Самый подходящий блок

- выбор самого неподходящего блока. Самый неподходящий блок – это блок с максимальным размером. Список свободных блоков необходимо отсортировать в порядке убывания размера и выбрать первый блок, как показано на рисунке 1.17;



| Адрес | Размер | PID |
|--|--------|-----|
| <input checked="" type="checkbox"/> 17 | 56 | -1 |
| <input type="checkbox"/> 143 | 45 | -1 |
| <input type="checkbox"/> 216 | 36 | -1 |

Рис: 1.17 Самый неподходящий блок

Таким образом выполнив все сорок шагов первого задания, при этом не должно остаться неисправленных ошибок, студент может переходить к выполнению второго задания лабораторной работы

2 Планирование и диспетчеризация процессов

Планирование процессов (задач) – это определение очередности получения ресурсов вычислительной системы для процессов при их активизации. Планирование процесса связано с его переводом из состояния бездействия в состояние готовности. Такой перевод осуществляется однократно на интервале существования процесса.

Стратегия планирования определяет, какие процессы мы планируем на выполнение для того, чтобы достичь поставленной цели. Известно большое количество различных стратегий выбора процесса, которому необходимо предоставить процессор. Среди них, прежде всего, можно назвать следующие стратегии:

- по возможности заканчивать вычисления (вычислительные процессы) в том же самом порядке, в котором они были начаты;
- отдавать предпочтение более коротким процессам;
- предоставлять всем пользователям (процессам пользователей) одинаковые услуги, в том числе и одинаковое время ожидания.

Основными задачами алгоритма планирования процессов являются:

- равнодоступность – предоставление каждому процессу справедливой доли времени центрального процессора;
- принуждение к определенной политике – наблюдение за выполнением определенной политики;
- баланс – поддержка загрузки всех составных частей системы [3].

Диспетчеризация процессов (задач) – это определение очередности получения процессора для процессов (задач), находящихся в состоянии готовности, с целью их выполнения. Диспетчеризация процесса связана с его переводом из состояния готовности в состояние выполнения (счёта). Диспетчеризация, для конкретного процесса, может выполняться многократно, т. е. процесс может несколько раз переходить из состояния

готовности в состояние выполнения и обратно на интервале своего существования. Так как в каждый такт процессорного времени могут выполняться команды только одной задачи, диспетчеризация предполагает создание и модификацию очереди готовых к выполнению задач (процессов).

Вторая часть лабораторной работы посвящена планированию процессов в ОС, использующих невытесняющие алгоритмы планирования. В ответ на событие, описываемое в окне событий, студент должен смоделировать действия системы в подобной ситуации.

2.1 Интерфейс и функции окна диспетчера процессов

Вид окна диспетчера процессов приведен на рисунке 2.1.

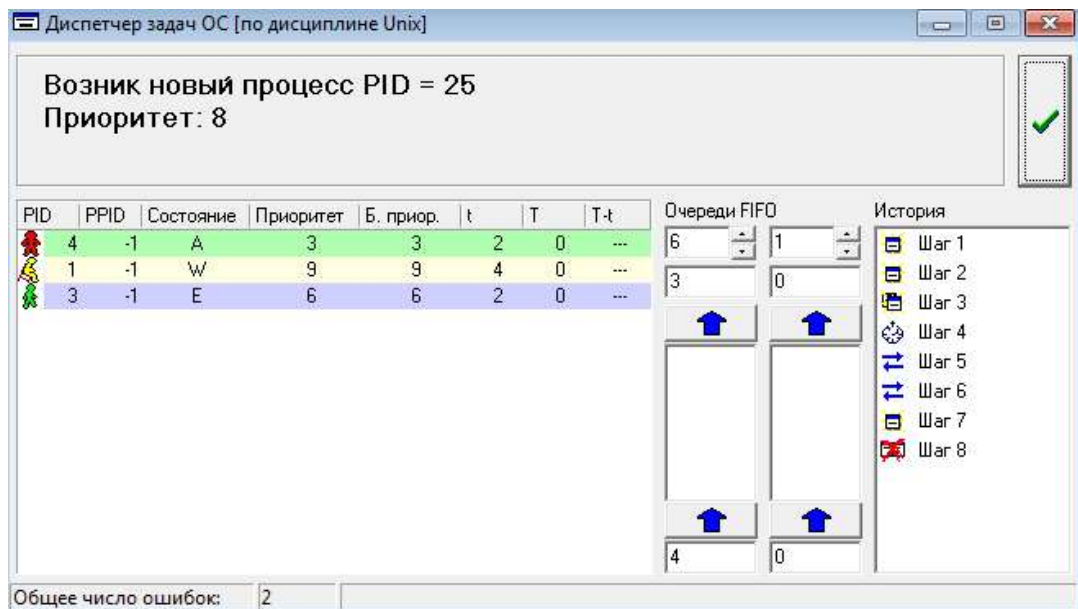


Рис 2.1 Диспетчер процессов

Так же, как и для окна диспетчера памяти, в верхней части окна расположена информация о произошедшем событии.

В центральной части окна находится список процессов, который содержит следующую информацию:

- состояние процесса; отображается цветом строки, значком, а также буквой в столбце «Состояние»:
 - «E» – процесс выполняется;
 - «A» – процесс готов к выполнению;

○ «W» – процесс обратился к устройствам ввода/вывода и ожидает завершения обмена.

- идентификатор процесса – расположен в столбце «PID»;
- идентификатор родительского процесса – расположен в столбце «PPID»;
- текущий приоритет процесса (в системе Windows NT) – расположен в столбце «Приоритет»;
- базовый приоритет процесса (в системе Windows NT) – расположен в столбце «Приоритет»;
- время выполнения процесса (инкрементируется только когда процесс выполняется) – находится в столбце «t»;
- предполагаемое время выполнения процесса – находится в столбце «T»;
- разность между предполагаемым временем и реальным временем выполнения процесса (по сути – время, оставшееся до завершения процесса) – находится в столбце «T-t»; если значение отрицательное (процесс превысил лимит времени), вместо числа выводится «-».

Справа от списка процессов находятся два окна очередей. В любом из них можно отобразить любую очередь (с номерами от 1 до 15). Для добавления значения в конец очереди, необходимо ввести его в нижнее окно ввода и нажать на нижнюю кнопку с синей стрелкой. Для извлечения значения из очереди необходимо нажать на верхнюю кнопку с синей стрелкой. Извлеченное значение помещается в окно ввода над кнопкой. В левом окне отображения очереди имеется возможность изменять порядок элементов в очереди путем перетаскивания мышью.

Пользователь имеет возможность выполнить следующие действия:

- выдать отказ на запрос; для этого необходимо щелкнуть на кнопке «Принять», не выполняя никаких других действий;
- добавить процесс в список процессов – щелкнуть правой кнопкой мыши на списке процессов и выбрать пункт «Добавить» в контекстном

меню, после чего ввести параметры создаваемого процесса: идентификатор, идентификатор родителя (если родителя нет – -1), значение базового и текущего приоритетов, предполагаемое и текущее время выполнения, состояние процесса, и нажать кнопку «ОК»;

- добавить идентификатор процесса в очередь;
- поменять порядок элементов в очереди;
- прочитать значение из начала очереди;
- удалить процесс из списка процессов и из очереди – щелкнуть правой кнопкой мыши на нужном элементе в списке процессов и в проявившемся меню выбрать пункт «Удалить»;
- переключить процесс в состояние ожидания – щелкнуть правой кнопкой мыши на нужном элементе в списке процессов и в проявившемся меню выбрать пункт «Переключить в состояние ожидания»;
- переключить процесс в состояние готовности – щелкнуть правой кнопкой мыши на нужном элементе в списке процессов и в проявившемся меню выбрать пункт «Переключит в состояние готовности»;
- выбрать процесс для выполнения – щелкнуть правой кнопкой мыши на нужном элементе в списке процессов и в проявившемся меню выбрать пункт «Переключиться».

Ответные действия на произошедшее событие необходимо завершать нажатием кнопки «Принять», которая находится в правом верхнем углу окна.

В правой части экрана находится список выполненных шагов.

В нижней части окна расположена строка статуса, в которой отображается общее число ошибок, которые допустил пользователь в процессе выполнения лабораторной работы.

По достижении заданного числа шагов выводится сообщение о выполнении лабораторной работы, а также количество неисправленных ошибок.

2.2 Алгоритм работы диспетчера задач

Диспетчер задач обрабатывает следующие события:

1. создание нового процесса;
2. создание дочернего процесса;
3. завершение процесса;
4. запрос на ввод/вывод;
5. завершение ввода/вывода;
6. передача процессом управление операционной системе;
7. истечение отведенного процессу кванта времени для дисциплин с вытесняющей многозадачностью.

По приходу события диспетчер должен проконтролировать корректность запроса:

- заданные значения должны находиться в соответствующих диапазонах;
- для событий 1 и 2 процесс с таким PID не должен существовать (рис. 2.2);

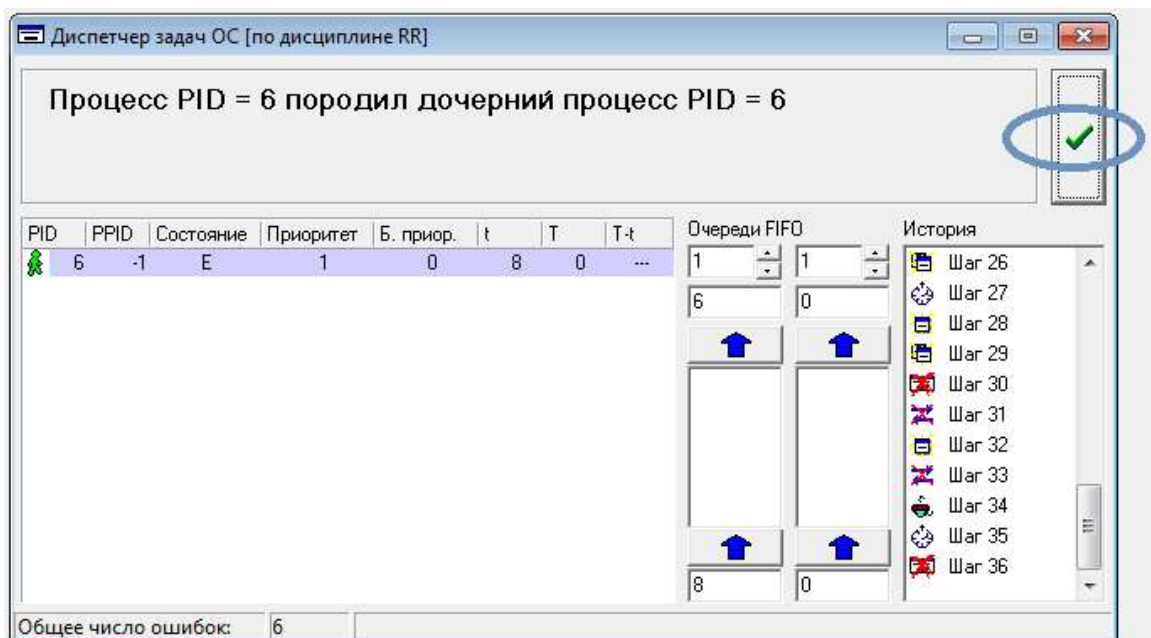


Рис 2.2 Создание уже существующего процесса

- дочерний процесс может быть порожден только процессом, выполняющимся в данный момент (рис. 2.3);

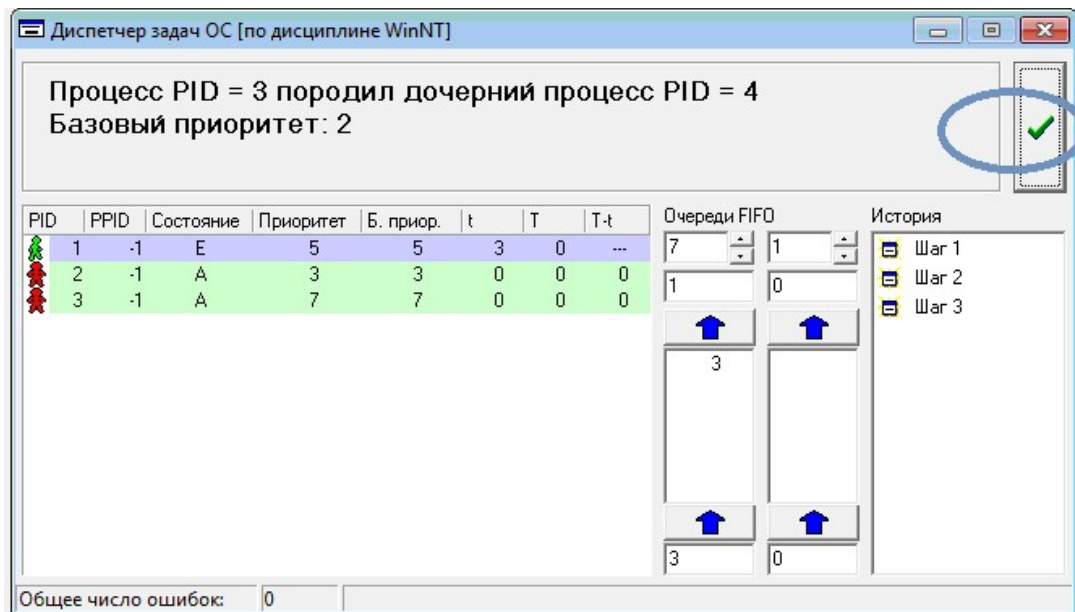


Рис 2.3 Создание дочернего процесса не выполняющимся процессом

• запрос на ввод/вывод, а также запрос на передачу управления ОС может выдать только выполняющийся процесс (рис. 2.4-2.5);

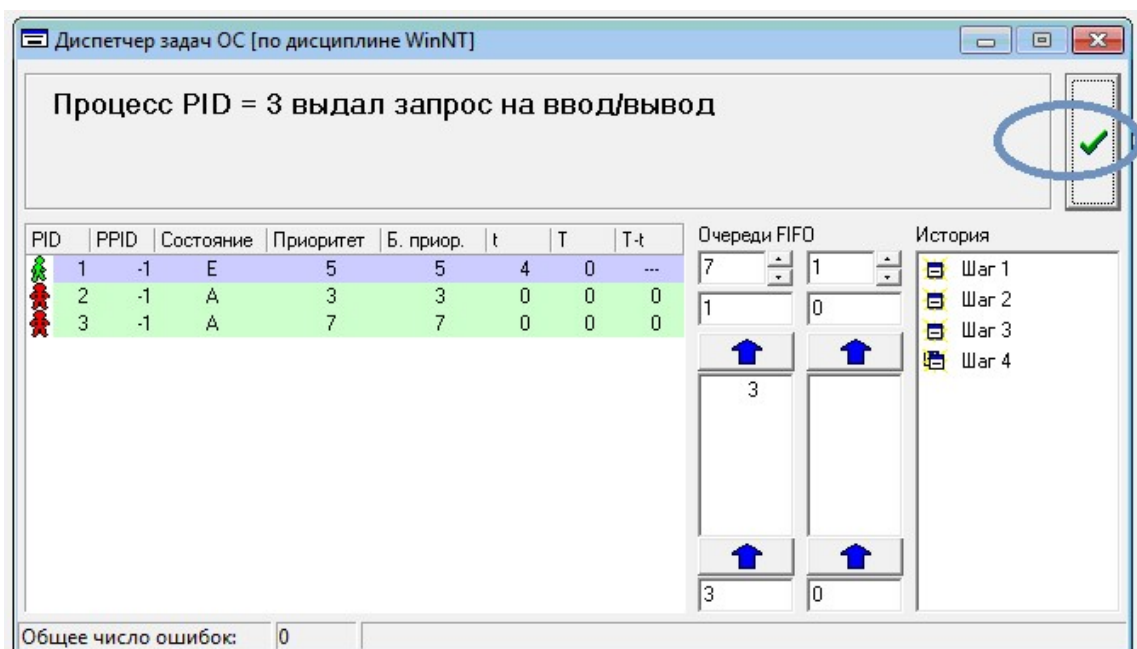


Рис 2.4 Запрос на ввод/вывод не выполняющего процесса

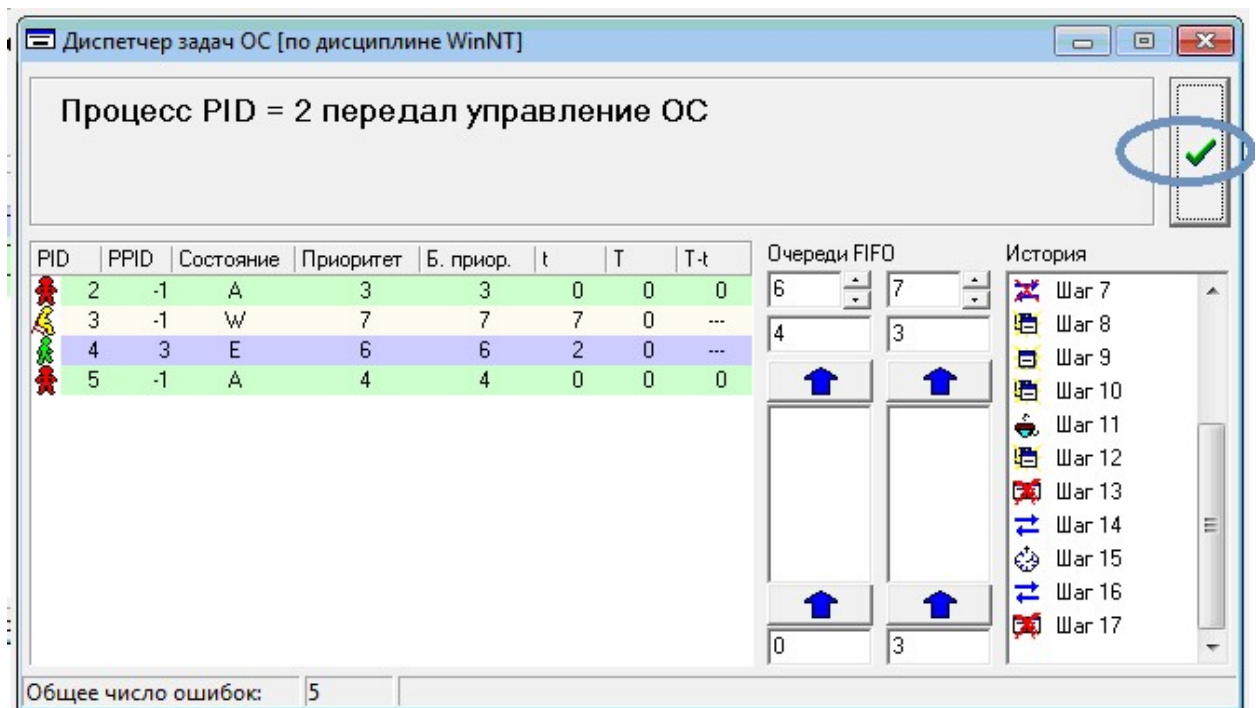


Рис 2.5 Передача управления не выполняющимся процессом

- завершение ввода/вывода может произойти только для процесса, находящегося в данный момент в состоянии ожидания завершения операций ввода/вывода (рис. 2.6);

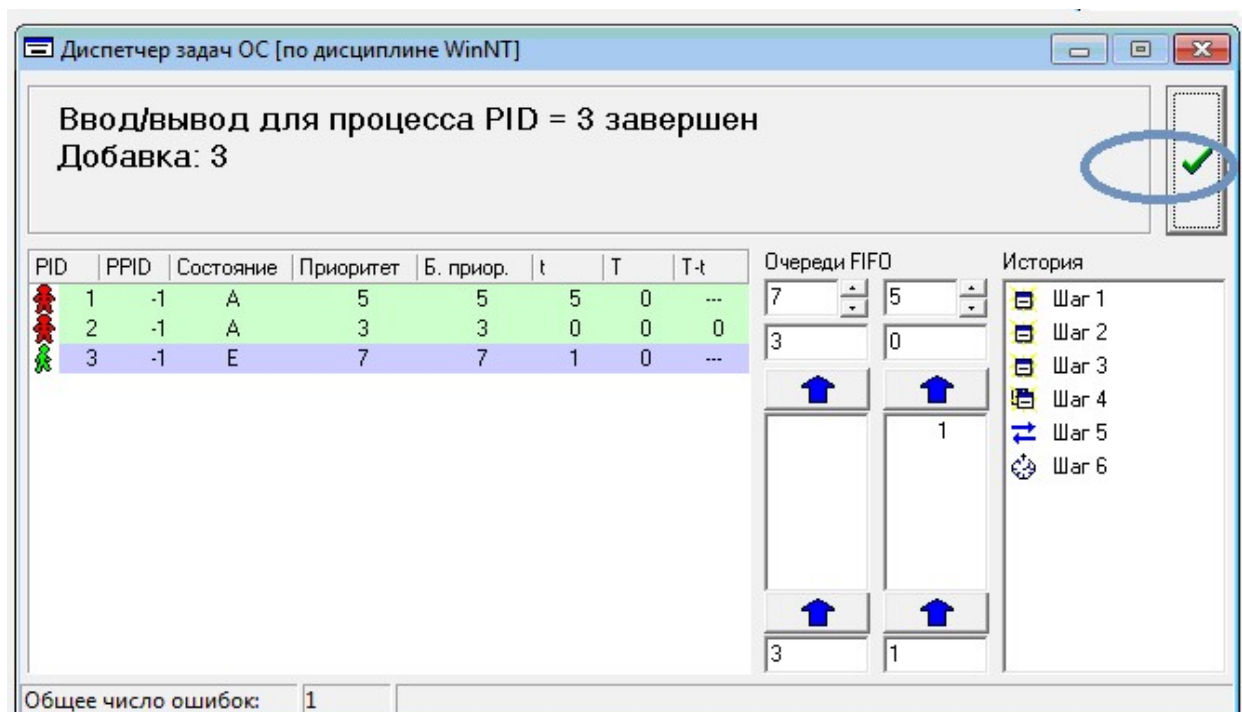


Рис 2.6 Завершения ввода/вывода готового к выполнению процесса

- при завершении процесса удаляются и его дочерние процессы (рис. 2.7);

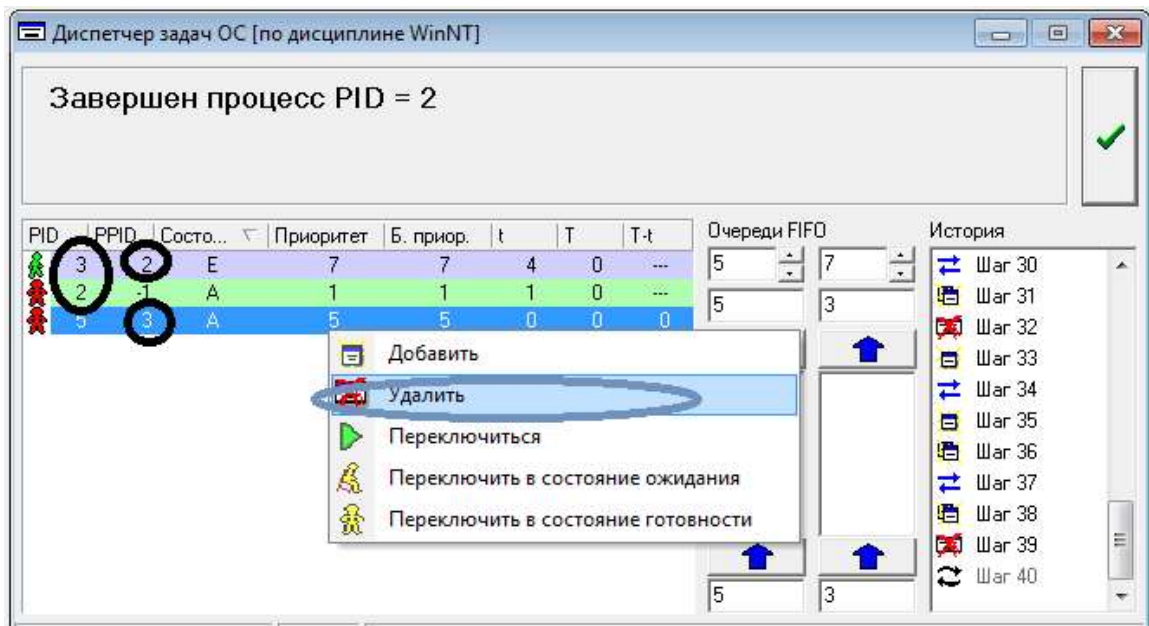


Рис 2.7 Удаление процесса и его дочерних процессов

• При отсутствии активных и готовых процессов не выполняем никаких действий (рис. 2.8);

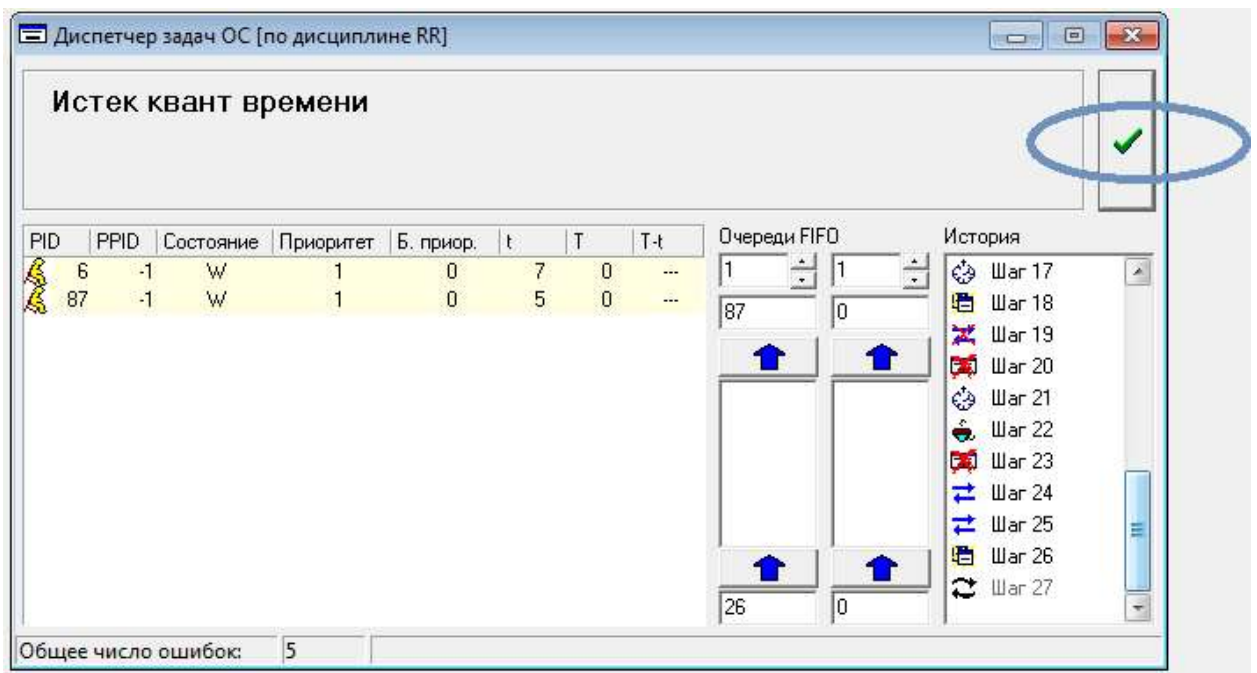


Рис 2.8 Отсутствие готовых процессов

В лабораторной модели реализовано 15 очередей. В них хранятся активные (готовые к выполнению) (но не выполняющиеся!) процессы. Помещение в очередь происходит при создании процесса, при переходе из состояния ожидания в активное состояние, а также при смене выполняемого процесса(не в режиме ожидания ввода/вывода). Извлечение из очереди

происходит при смене выполняющегося процесса для выяснения идентификатора процесса, которому ОС должна передать управление.

Внимание! Даже если в системе находится только один активный процесс и пришло сообщение о передаче управления другому процессу (то есть самому себе), операции помещения в очередь и извлечение из очереди выполнить необходимо, так как диспетчеру заранее неизвестно, что переключение произойдет на тот же процесс (просмотр списка процессов занимает больше времени, чем помещение и извлечение процесса из очереди). Кроме того, если работа происходит в системе с динамическими приоритетами, помещение процесса в очередь во многих случаях необходима для изменения приоритета процесса).

Смена процесса может произойти при передаче управления ОС, при истечении кванта времени, при создании нового процесса и при завершении процесса ввод/вывод.

В лабораторной установке реализованы 7 дисциплин диспетчеризации.

2.3 Невытесняющие дисциплины планирования

В лабораторной установке представлены 3 не вытесняющие дисциплины: FCFS, SJN, SRT. Это дисциплины, не использующие квантование.

2.3.1 Дисциплина FCFS

Новые процессы добавляются в очередь 2, остальные – в очередь 1. Выборка PID процесса для передачи ему управления осуществляется из очереди 1; если она пуста – то из очереди 2. Если есть только 1 активный процесс, очереди не используются. Если очередь 1 пуста и приходит событие передача процессом управление операционной системе, диспетчер не выполняет никаких операций.

При завершение ввода/вывода (ожидающего):

- 1) переключение процесса в состояние готовности;
- 2) добавление процесса в 1 очередь.

Выполнение данных действий представлено на рисунке 2.9.

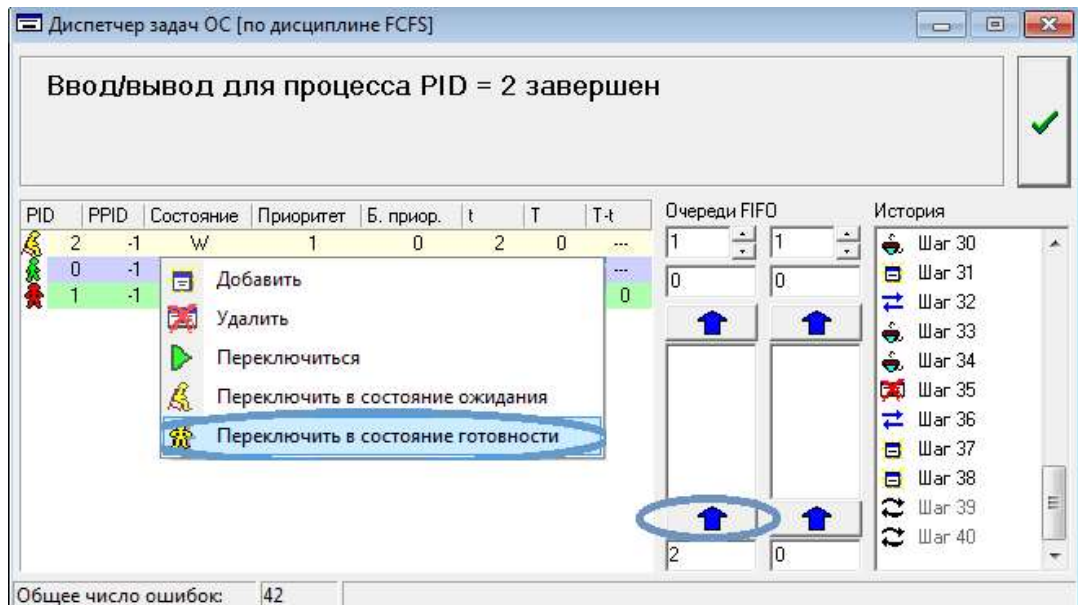


Рис 2.9 Завершение ввода/вывода для FCFS

При передачи управления ОС активным процессом:

- 1) добавление процесса в первую очередь;
- 2) извлечение из очереди 1(если пустая 2) и переключение на новый процесс.

Выполнение данных действий представлено на рисунке 2.10.

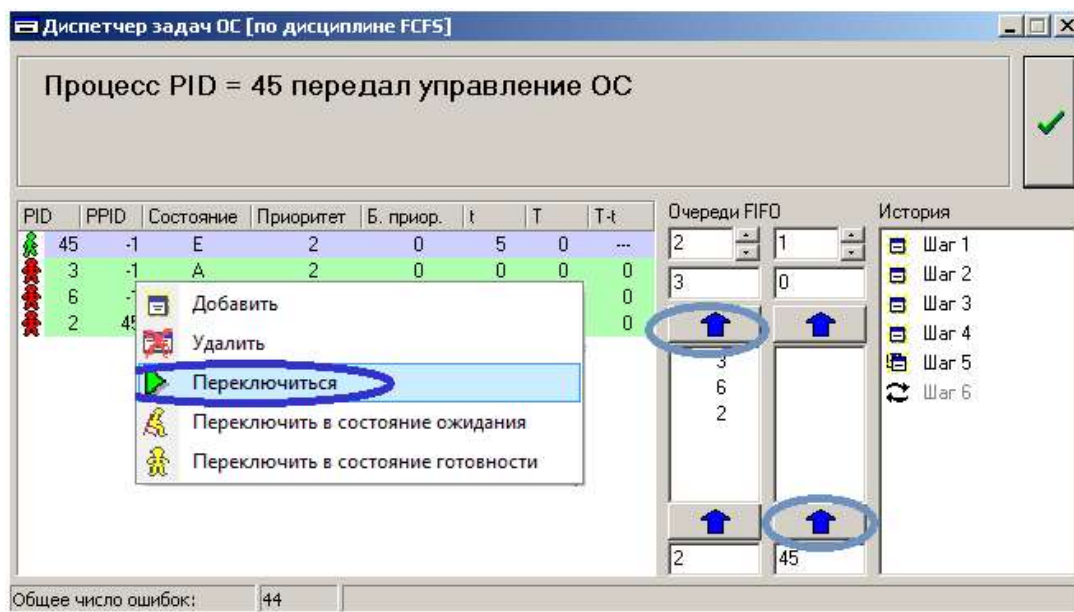


Рис 2.10 Передача управления для FCFS

2.3.2 Дисциплины SJN, SRT

Работает только очередь 1. После помещения процесса в очередь необходимо переместить его на соответствующее место. Для дисциплины SJN вперед помещаются процессы с меньшим заданным временем выполнения (поле T), а для дисциплины SRT – процессы с меньшим оставшимся временем (поле T-t). При этом, если время выполнения процесса превышает заданное, ему назначается штраф. При помещении в очередь такой процесс не перемещается, а остальные процессы, даже с большим временем, при помещении в очередь ставятся перед ним (рис. 2.11).

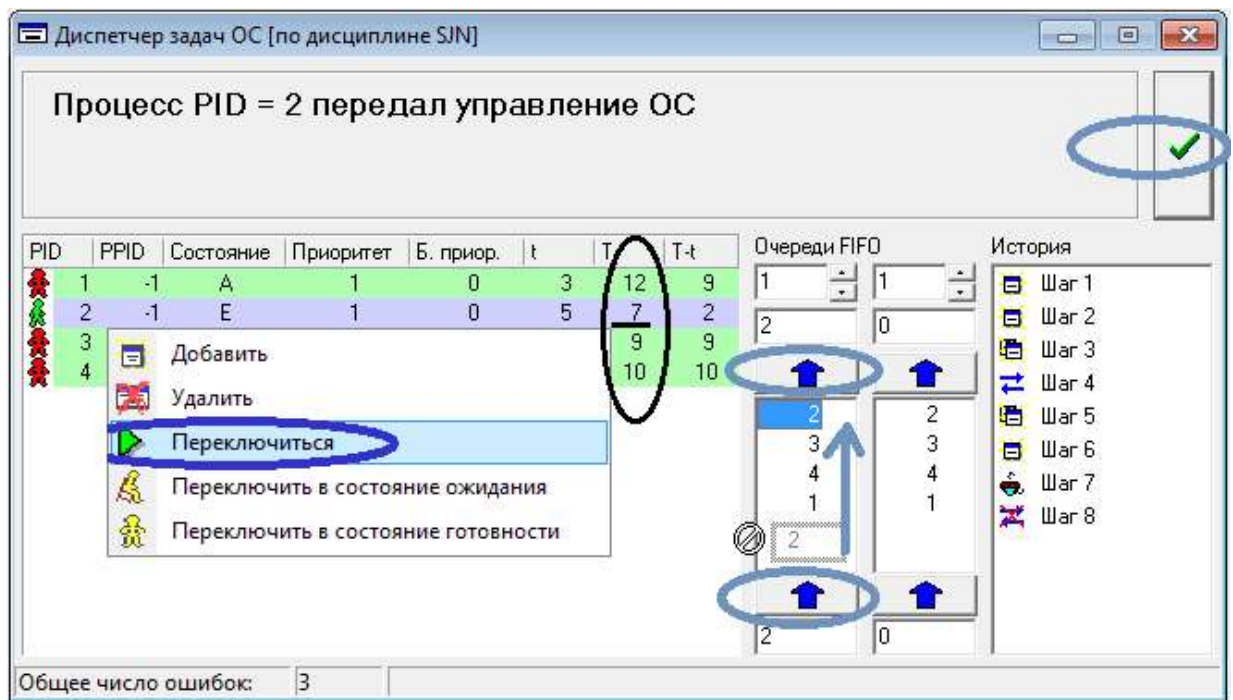


Рис 2.11 Передача управления процессом с меньшим заданным временем выполнения для SJN

При добавлении нового процесса или при завершении ввода/вывода процесса с меньшим временем выполнения (оставшимся временем для SRT) передача управления не выполняется (рис. 2.12).

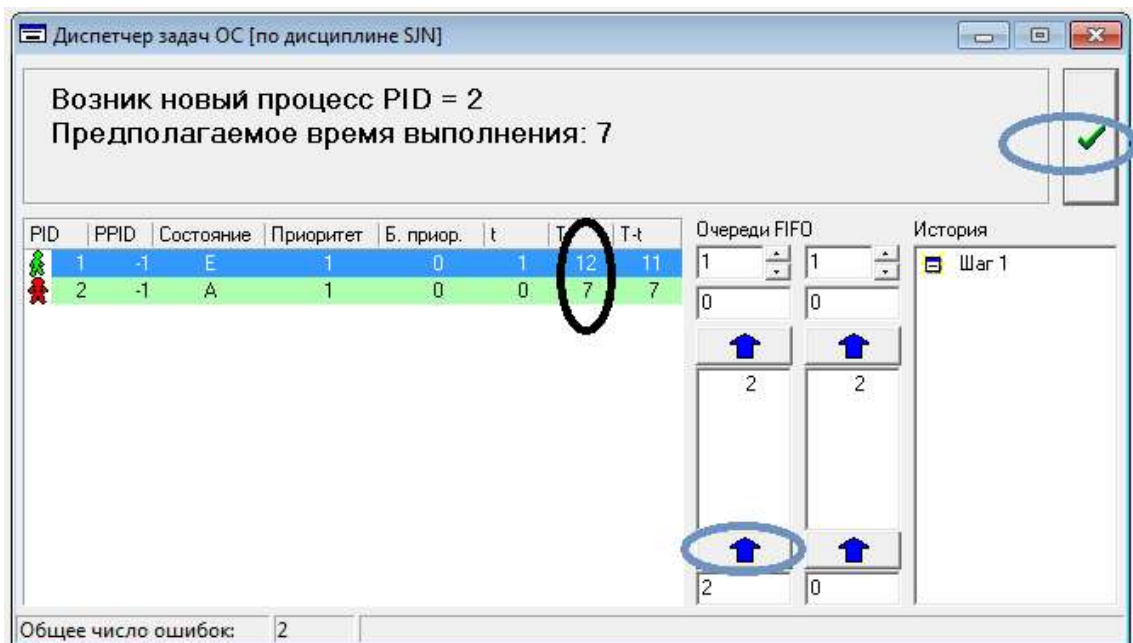


Рис 2.12 Создание процесса с меньшим временем выполнения для SJN

2.4 Вытесняющие дисциплины диспетчеризации

К вытесняющим дисциплинам диспетчеризации, реализованным в лабораторной установке, являются дисциплина RR, дисциплины диспетчеризации ОС UNIX, Windows NT, OS/2.

2.4.1 Дисциплина RR

Используется только очередь 1, которая заполняется в порядке пребывания. Отсутствуют какие-либо приоритеты.

Создание процесса для данной дисциплины представлено на рисунке 2.13.

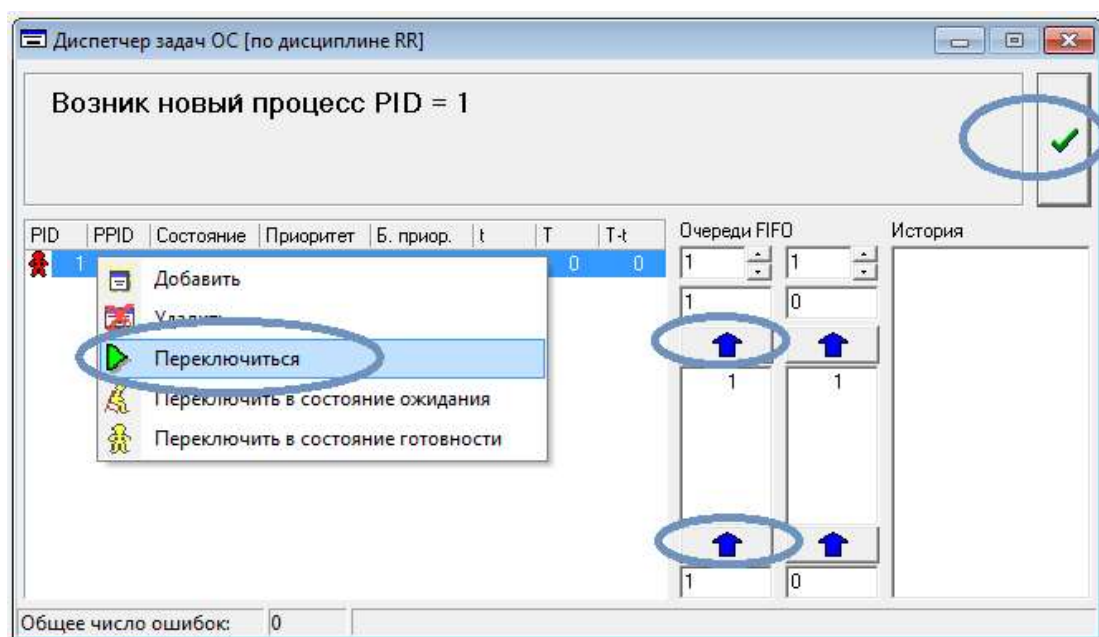


Рис 2.13 Создание процесса для RR

2.4.2 Планирование процессов в ОС UNIX

В лабораторной установке реализована упрощенная модель диспетчеризации процессов с динамическими приоритетами. Для режима задачи приоритет меняется в диапазоне 1-7, для режима ядра — 8 (системный диапазон). Процессы, приоритеты которых лежат в диапазоне 9-15, являются процессами с фиксированным приоритетом.

Процессу, ожидающему недоступного в данный момент ресурса, система определяет значение приоритета сна = 8. Когда процесс пробуждается, ядро устанавливает значение текущего приоритета процесса равным приоритету сна. После завершения системного вызова перед возвращением в режим задачи ядро восстанавливает приоритет режима задачи, сохраненный перед выполнением системного вызова.

При истечении кванта времени или передаче управления ОС процессорное время передается процессу с высшим приоритетом, стоящим первым в очереди (рис. 2.14).

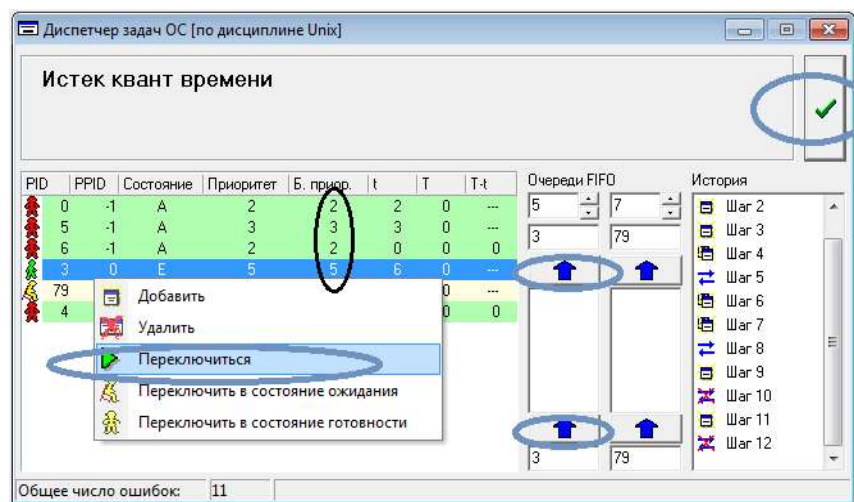


Рис 2.14 Истек квант времени для Unix

При создании процесса или при завершении ввода/вывода процесса с более высоким приоритетом, чем у активного, происходит переключение на данный процесс. Данное действия представлены на рисунках 2.15-2.17.

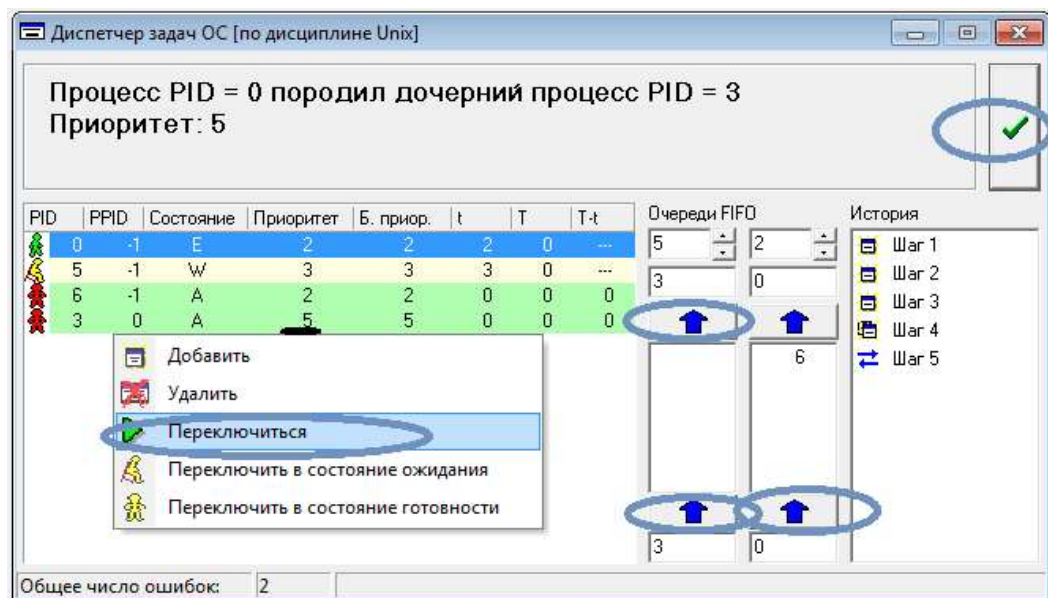


Рис 2.15 Создание процесса с высшим приоритетом для Unix

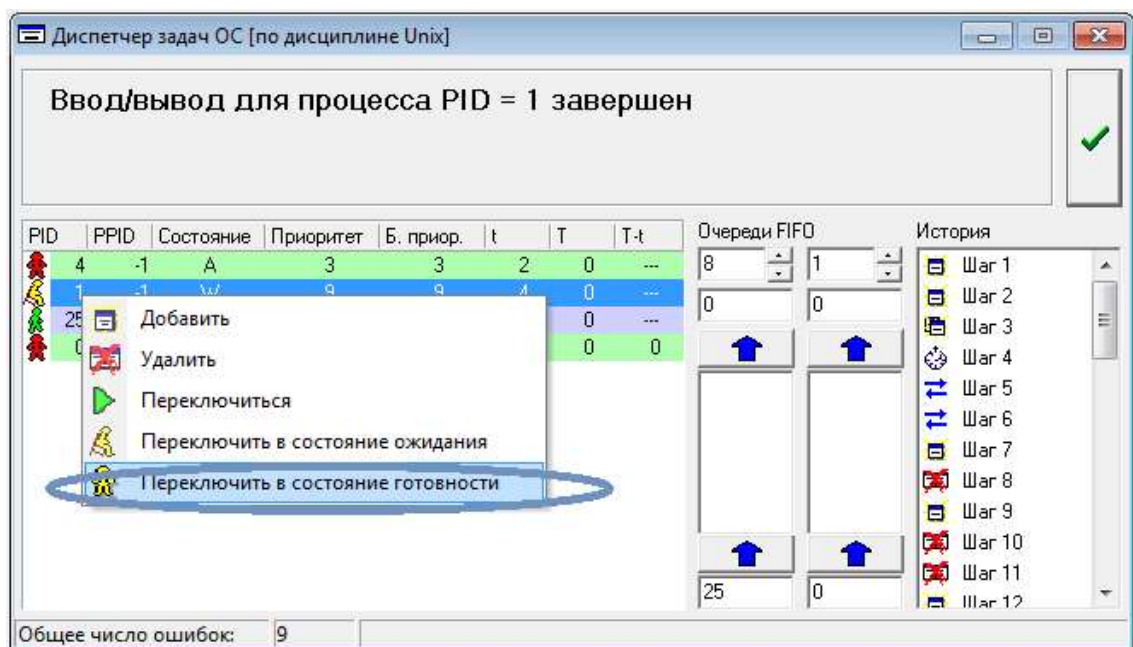


Рис 2.16 Завершение ввода/вывода для Unix



Рис 2.17 Завершение ввода/вывода для Unix

2.4.3 Планирование процессов в ОС Windows NT

В лабораторной установке реализована упрощенная модель диспетчеризации процессов с динамическими приоритетами.

Используются 15 очередей. Текущий приоритет процесса соответствует номеру очереди, в которой он находится. При помещении в очередь в результате передачи управления процессом операционной системе или истечения отведенного ему кванта времени текущий приоритет процесса уменьшается на единицу (то есть, он помещается в очередь с меньшим номером), как представлено на рисунке 2.18.

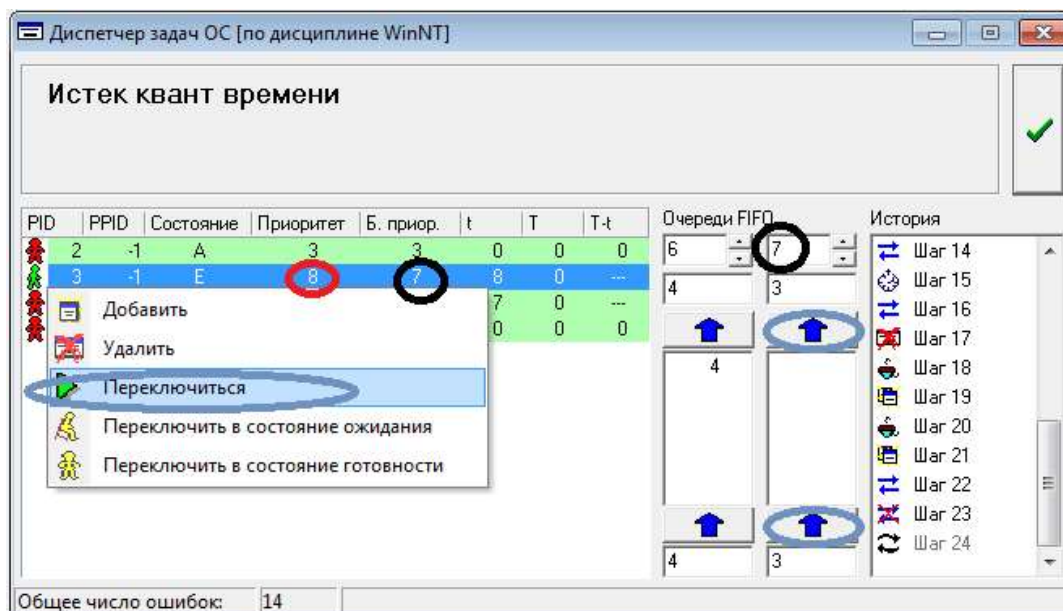


Рис 2.18 Уменьшение приоритета для WinNT

По завершении операций ввода/вывода ожидающему процессу назначается некоторая добавка (он помещается в более приоритетную очередь).

Выполнение данного действия представлено на рисунках 2.19-2.10

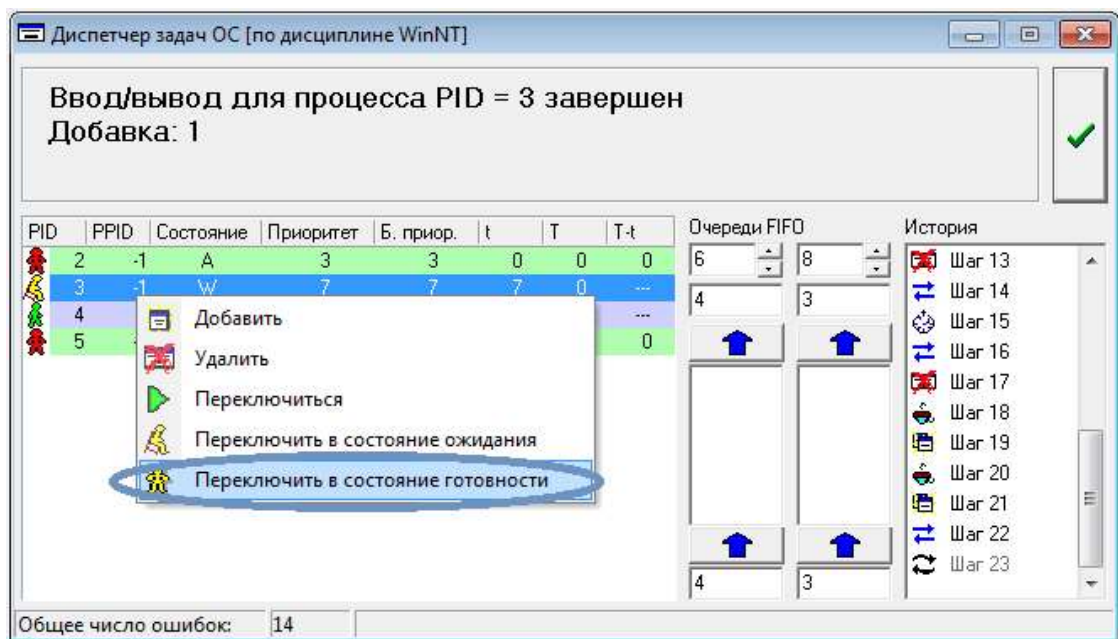


Рис 2.19 Добавка приоритета для WinNT

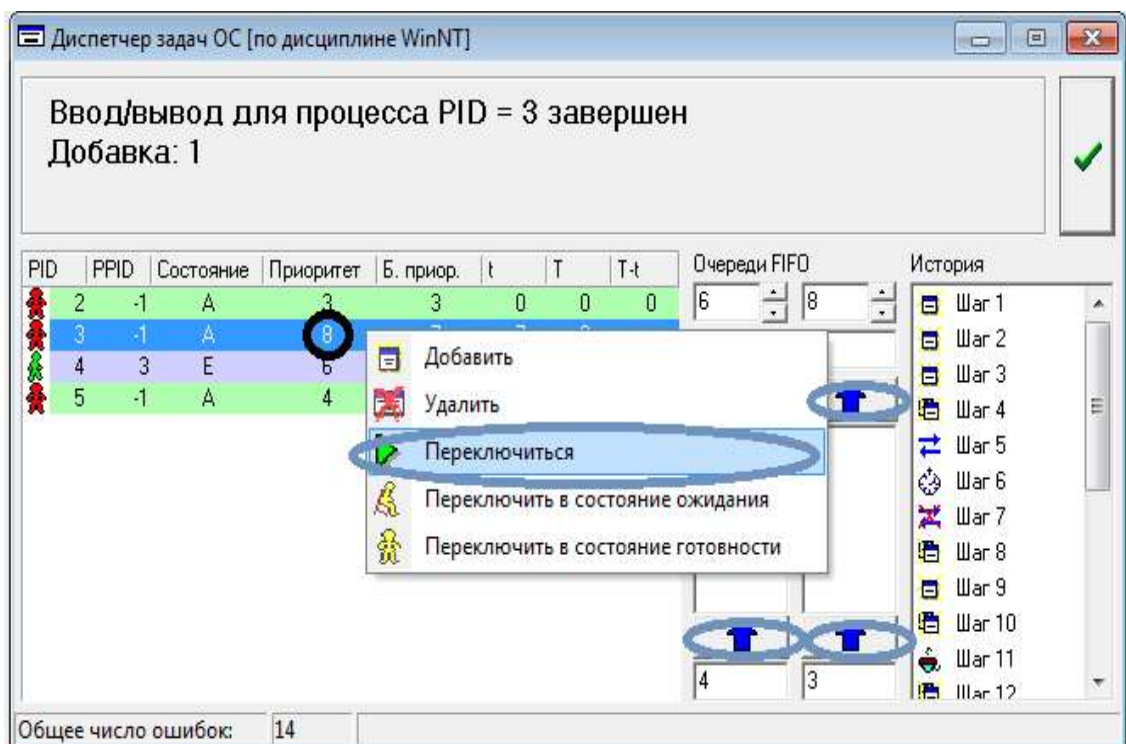


Рис 2.20 Добавка приоритета для WinNT

Внимание! В любом случае значение текущего приоритета процесса не может опуститься ниже базового приоритета этого процесса или достигнуть 16.

Для выбора процесса, которому следует передать управление, просматривают очереди, начиная с самой приоритетной. Если очередь не

пустая, из нее извлекается PID процесса, на который затем осуществляется переключение процессора (рис. 2.22).

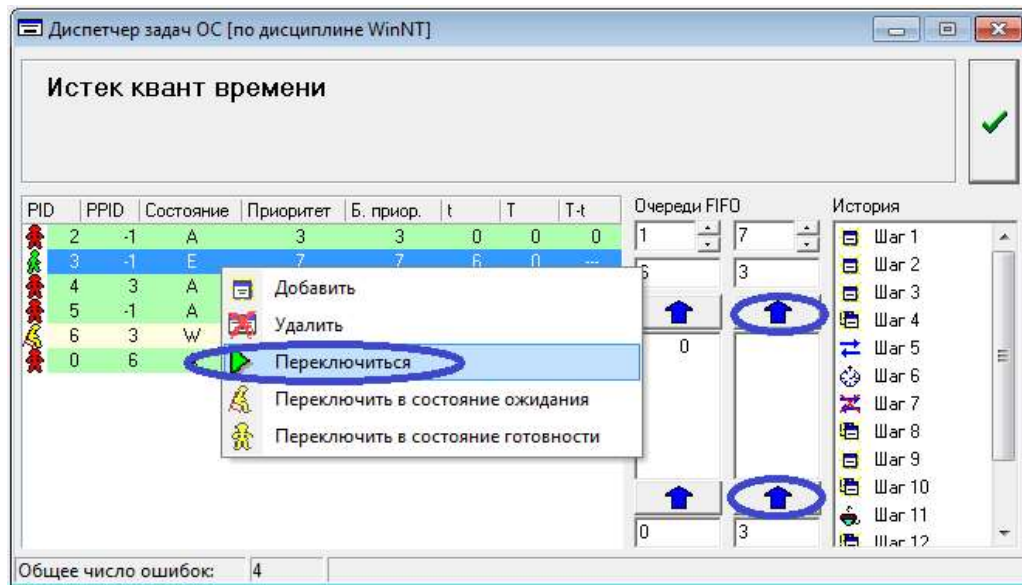


Рис 2.22 Истечение кванта времени для WinNT

При создании процесса или при завершении ввода/вывода процесса с более высоким приоритетом, чем у активного, происходит переключение на данный процесс (рис 2.23).

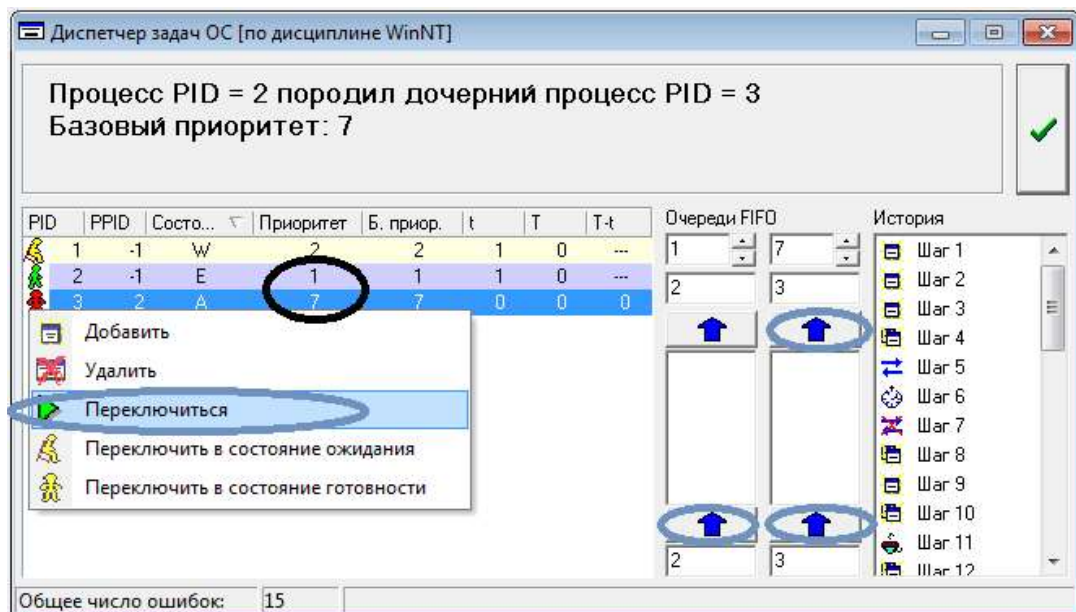


Рис 2.23 Создание процесса с большим приоритетом для WinNT

2.4.4 Планирование процессов в OS/2

В лабораторной установке реализована упрощенная модель диспетчеризации процессов с динамическими приоритетами. Классу

критических задач принадлежит группа приоритетов с интервалом значений от 13 до 15, приоритетному классу – от 9 до 12, к стандартному классу – от 5 до 8, к остаточному классу – от 1 до 4.

Операционная система изменяет приоритет задачи в следующих случаях:

- приоритет задачи увеличивается, когда она становится активной;
- по завершении операции ввода/вывода задача получает самый высокий уровень приоритета ее класса.

Ситуация добавки приоритета после ввода/вывода представлена на рисунке 2.24.



Рис 2.24 Добавка приоритета после ввода/вывода для OS/2

При создании процесса более высоким приоритетом, чем у активного, происходит переключение на вновь созданный процесс. Если при завершении ввода/вывода процесса его приоритет оказался выше приоритета активной задачи, происходит то же самое (рис 2.25).

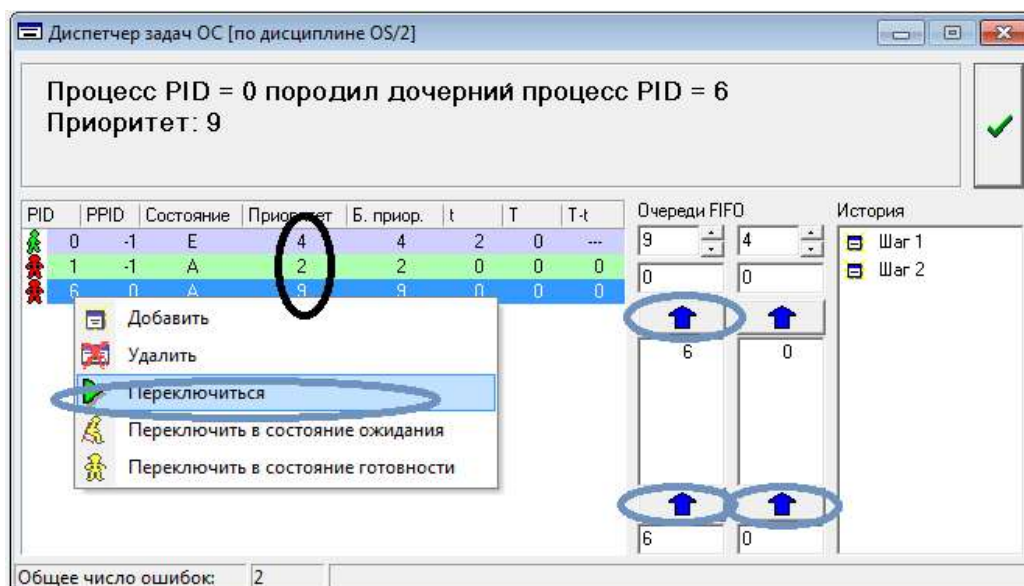


Рис 2.25 Создание процесса с высшим приоритетом для OS/2

Для выбора процесса, которому следует передать управление, просматривают очереди, начиная с самой приоритетной. Если очередь не пустая, из нее извлекается PID процесса, на который затем осуществляется переключение процессора (рис. 2.26).

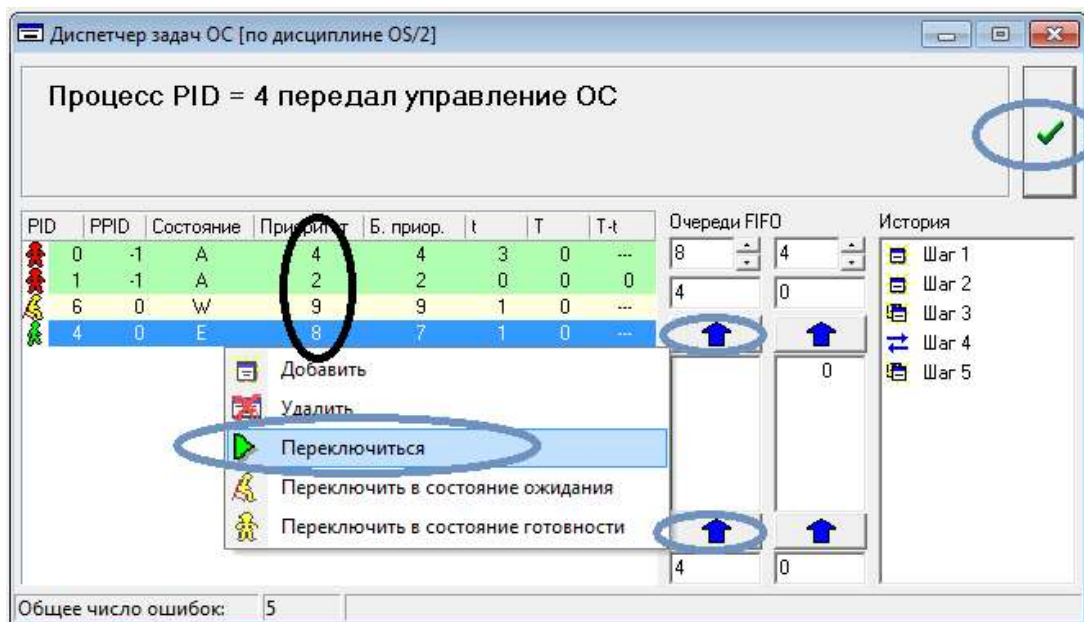


Рис 2.26 Передача управления для OS/2

Заключение

Управление ресурсами составляет важную часть функций любой операционной системы, в особенности мультипрограммной. Операционная система должна управлять всеми ресурсами вычислительной машины таким образом, чтобы обеспечить максимальную эффективность ее функционирования.

Знания, полученные в ходе выполнения данной лабораторной работы, включают в себя понимание алгоритмов управления памятью и алгоритмов планирования и диспетчеризации процессов. Данные знания могут быть применены во множестве сфер, связанных с работой с операционными системами.

Библиографический список

1. Назаров, С.В. Современные операционные системы [Текст]: учеб. пособие / С. В. Назаров, А. И. Широков. - М. : Интернет-Университет Информационных Технологий, 2011. - 279 с.

2. Управление процессами. Планирование и диспетчеризация процессов [Электронный ресурс] – режим доступа: <http://www.intuit.ru/studies/courses/641/497/lecture/11280?page=1> – 24.11.2016