Министерство науки и высшего образования Российской Федерации
Алтайский государственный университет

П.Н. Уланов, И.А. Шмаков

Операционные системы

Методические указания к выполнению лабораторных работ

Барнаул
Издательство Алтайского государственного университета
2018

Барнаул 2018

Рецензент:
к.ф-.м.н. Рыкшин Алексей Юрьевич

П.Н. Уланов, И.А. Шмаков

Операционные системы [Текст] : Методические указания к выполнению лабораторных работ. — Барнаул : Изд-во Алт. ун-та, 2018. — 27 с.

Методические указания к выполнению лабораторных работ по курсу *«Операционные системы»*, для студентов обучающихся по направлению *«09.03.01 — Информатика и вычислительная техника»*

Оглавление

Введение

Операционная система, сокр. *ОС* (англ. operating system, OS) — комплекс взаимосвязанных программ, предназначенных для управления ресурсами компьютера и организации взаимодействия с пользователем.

UNIX — семейство переносимых, многозадачных и многопользовательских операционных систем.

Идеи, заложенные в основу **UNIX**, оказали огромное влияние на развитие компьютерных операционных систем. В настоящее время **UNIX**-системы признаны одними из самых исторически важных ОС.

Первая система **UNIX** была разработана в подразделении **Bell Labs** компании **AT&T**. С тех пор было создано большое количество различных **UNIX**-систем. Юридически право называться **«UNIX»** имеют лишь те операционные системы, которые прошли сертификацию (**Single UNIX Specification**) на соответствие стандарту. Остальные же, хотя и используют сходные концепции и технологии, называются **UNIX**-подобными операционными системами (англ. **UNIX**-like).

UNIX-подобная операционная система (иногда сокр. как *nix) — операционная система, которая образовалась под влиянием **UNIX**. Термин включает свободные/открытые операционные системы, образованные от **UNIX** компании **Bell Labs** или эмулирующие его возможности, коммерческие и запатентованные разработки, а также версии, основанные на исходном коде **UNIX**. Нет стандарта, определяющего термин, и допустимы различные точки зрения о том, считать определённый продукт **UNIX**-подобным или нет.

Деннис Ритчи, один из создателей **UNIX**, выразил своё мнение, что **UNIX**-подобные системы, такие, как **Linux**, являются де-факто **UNIX**-системами. **Эрик Рэймонд** предложил разделить **UNIX**-подобные системы на 3 типа:

1. Генетический **UNIX**: Системы, исторически связанные с кодовой базой **AT&T**. Большинство, но не все коммерческие дистрибутивы **UNIX**-систем попадают под эту категорию. Так же, как и **BSD**-системы, которые являются результатами работы **университета Беркли** в поздних 1970-х и ранних 1980-х. В некоторых из этих систем отсутствует код **AT&T**, но до сих пор прослеживается происхождение от разработки **AT&T**.

2. **UNIX** по товарному знаку, или бренду: эти системы, в основном коммерческого характера, были определены **The Open Group** как соответствующие **Единой спецификации UNIX**, и им разрешено носить имя **UNIX**. Большинство этих систем — коммерческие производные кодовой базы **UNIX System V** в той или иной форме (например, **Amiga UNIX**), хотя некоторые (например, **z/OS** компании **IBM**) заслужили торговую марку через слой совместимости с **POSIX**, не являясь, по сути, **UNIX**. Многие старые **UNIX**-системы не подходят под это определение.

3. **UNIX** по функциональности: В целом, любая система, поведение которой примерно соответствует спецификации **UNIX**. К таким системам можно отнести **Linux** и Minix, которые ведут себя подобно **UNIX**-системе, но не имеют генетических связей с кодовой базой **AT&T**. Большинство свободных/открытых реализаций **UNIX**, являясь генетическим **UNIX** или нет, подпадают под ограниченное определение этой категории в связи с дороговизной сертификации **The Open Group**, которая стоит несколько тысяч долларов.

Cygwin, не являясь операционной системой, предоставляет **UNIX**-подобную среду в **Microsoft Windows**; также существуют сервисы **Microsoft Windows** для **UNIX**.

MS Windows — семейство проприетарных *Операционных систем* корпорации **Microsoft**, ориентированных на применение графического интерфейса при управлении. **MS Windows** изначально была всего лишь графической надстройкой-программой для операционной системы 80-х и 90-х годов **MS-DOS**.

Ядро (kernel) — центральная часть *ОС*, обеспечивающая приложениям координированный доступ к ресурсам компьютера, таким как процессорное время, память, внешнее аппаратное обеспечение, внешнее устройство ввода и вывода информации.

Монолитное ядро — это такой тип ядра, при котором все его части в одном адресном пространстве. В такой схеме *ОС* все компоненты её ядра являются составными частями одной программы, используют общие структуры данных и взаимодействуют друг с другом путём непосредственного вызова процедур. **Монолитное ядро** — это старейший способ организации *ОС*. Примером систем с монолитным ядром является большинство **UNIX**-систем.

Достоинства: Скорость работы, упрощённая разработка модулей.

Недостатки: Поскольку всё ядро работает в одном адресном пространстве, сбой в одном из компонентов может нарушить работоспособность всей системы.

Примеры: Традиционные ядра **UNIX** (такие как **BSD**), **Linux**; ядро **MS-DOS**, ядро **KolibriOS**.

Модульное ядро — современная, усовершенствованная модификация архитектуры монолитных ядер *ОС*.

В отличие от «классических» монолитных ядер, модульные ядра, как правило, не требуют полной перекомпиляции ядра при изменении состава аппаратного обеспечения компьютера. Вместо этого модульные ядра предоставляют тот или иной механизм подгрузки модулей ядра, поддерживающих то или иное аппаратное обеспечение (например, драйверов). При этом подгрузка модулей может быть как динамической (выполняемой «на лету», без перезагрузки *ОС*, в работающей системе), так и статической (выполняемой при перезагрузке *ОС* после переконфигурирования системы на загрузку тех или иных модулей).

Микроядро — это ядро *ОС*, которое предоставляет только элементарные функции управления процессами и минимальный набор абстракций для работы с оборудованием. Большая часть работы осуществляется с помощью специальных пользовательских процессов, называемых сервисами. Решающим критерием «микроядерности» является размещение всех или почти всех драйверов и модулей в сервисных процессах, иногда с явной невозможностью загрузки любых модулей расширения в собственно микроядро, а также разработки таких расширений.

Достоинства: Устойчивость к сбоям оборудования, ошибкам в компонентах системы. Основное достоинство микроядерной архитектуры — высокая степень модульности ядра *ОС*. Это существенно упрощает добавление в него новых компонентов. В микроядерной операционной системе можно, не прерывая её работы, загружать и выгружать новые драйверы, файловые системы и т. д. Существенно упрощается процесс отладки компонентов ядра, так как новая версия драйвера может загружаться без перезапуска всей *ОС*. Компоненты ядра *ОС* ничем принципиально не отличаются от пользовательских программ, поэтому для их отладки можно применять обычные средства. Микроядерная архитектура повышает надежность системы, поскольку ошибка на уровне непривилегиро-

ванной программы менее опасна, чем отказ на уровне режима ядра.

Недостатки: Передача данных между процессами требует накладных расходов.

Примеры: Symbian OS, **Windows CE**, **OpenVMS**, **Mach**, используемый в **GNU/Hurd** и **Mac OS X**, **QNX**, **AIX**, **Minix**, **ChorusOS**, **AmigaOS**, **MorphOS**.

Экзоядро — ядро *ОС*, предоставляющее лишь функции для взаимодействия между процессами, безопасного выделения и освобождения ресурсов. Предполагается, что **API** для прикладных программ будут предоставляться внешними по отношению к ядру библиотеками (откуда и название архитектуры).

Наноядро — архитектура ядра *ОС*, в рамках которой крайне упрощённое и минималистичное ядро выполняет лишь одну задачу — обработку аппаратных прерываний, генерируемых устройствами компьютера. После обработки прерываний от аппаратуры наноядро, в свою очередь, посылает информацию о результатах обработки (например, полученные с клавиатуры символы) вышележащему программному обеспечению при помощи того же механизма прерываний. Примером является **KeyKOS** — самая первая ОС на наноядре. Первая версия вышла ещё в **1983** году.

Гибридные ядра — это модифицированные микроядра, позволяющие для ускорения работы запускать «несущественные» части в пространстве ядра.

Пример: ядра **ОС MS Windows** семейства **NT**.

1. Используемая форма отчёта

Форма отчёта: отчёт к лабораторной работе должен содержать следующие пункты:

1. Титульный лист:
   (a) указание университета, факультета и кафедры;
   (b) названия предмета;
   (c) указание названия лабораторной работы
   (d) указание исполнителя работы (группа, ФИО);
   (e) указание проверяющего преподавателя;
   (f) даты защиты лабораторной работы.
2. *Введение и постановка задачи:* во введении требуется описать постановку задачи, выбранные методы и инструменты решения поставленной задачи;
3. *Теоретическое описание задачи:* в данном пункте отчёта требуется описать теоретическую часть работы;
4. *Вывод по работе:* в данном пункте отчёта требуется сделать вывод по проделанной работе, сделать оценку полученной погрешности, если это возможно;
5. *Приложение:* в данном разделе содержатся рисунки и листинги программ, которые не вошли в другие разделы отчёта.

Отчеты по лабораторным работам должны быть сданы преподавателю в электронном виде и соответствовать последним требованиям к оформлению лабораторных работ и ВКР кафедры «Вычислительной техники и электроники». Требования к формату:

- Формат pdf;
- Размер страницы A4;
- Кегль шрифта 14pt;
- Отступы: сверху — 2 *см*, слева 3 *см*, снизу — 2 *см*, , справа — 1,5 *см*;

- Все иллюстрации, блок-схемы, формулы и специальные символы должны легко читаться и не содержать явных «артефактов» сжатия, не быть размытыми, должны содержать подписи;

- Формулы необхдимо оформлять в той издательской системе, в которой верстается отчет, вставка формул изображениями не допускается.

2. Лабораторная работа №0: «Базовые команды для работы к командной строке»

Честность в политике есть результат силы, лицемерие — результат слабости.

«Полемические заметки (март 1911 г.)».— ПСС, 5-е изд., т. 20, с. 210.
Владимир Ильич Ленин

Список команд:

1. *pwd* — выводит имя текущую директории;
2. *ls* — выдаёт информацию о *ФАЙЛАХ* и *ДИРЕКТОРИЯХ* (по умолчанию о текущем каталоге);
3. *cd* — изменяет текущею директории на заданную;
4. *cp* — копирует *ИСТОЧНИК* в *НАЗНАЧЕНИЕ*, или несколько *ИСТОЧНИКОВ* в *КАТАЛОГ*;
5. *mv* — переименовывает *ИСТОЧНИК* в *НАЗНАЧЕНИЕ*, или перемещает *ИСТОЧНИК(и)* в *КАТАЛОГ*;
6. *date* — выводит текущее время в заданном *ФОРМАТЕ*, или изменяет время в системе;
7. *uptime* — выводит информацию о том, как долго запущена система.

Задание: Ознакомиться с командами представленными выше. Привести краткое описание к каждой команде.

Особые пункты отчёта: Отчёт должен содержать следующие элементы: для каждой команды представить примеры использования команд. **Примечание:** показать практические навыки работы с данными командами преподавателю.

Контрольные вопросы:

1. Для чего нужны команды и программы, представленные выше?
2. Для чего был создан *POSIX*?

3. Лабораторная работа №1: «Вспомогательные программы и команды»

> Математику уже за то
> любить следует, что она ум
> в порядок приводит.
>
> *Михаил Васильевич*
> *Ломоносов*

Теоретическая справка: *Make* — утилита, автоматизирующая процесс преобразования файлов из одной формы в другую. Чаще всего это компиляция исходного кода в объектные файлы и последующая компоновка в исполняемые файлы или библиотеки.

Утилита использует специальные *make*-файлы, в которых указаны зависимости файлов друг от друга и правила для их удовлетворения. На основе информации о времени последнего изменения каждого файла *make* определяет и запускает необходимые программы. *Make* была создана *Стюартом Фельдманом (Stuart Feldman)* в **1977** году в **Bell Labs**.

```
1  # Makefile для сборки книги.
2  default: help
3  .PHONY: default
4  book:
5          pdflatex -shell-escape main
6          biber main
7          pdflatex -shell-escape main
8          cp main.pdf Book-name.pdf
9  .PHONY: book
10 help:
11         @echo "Следующие цели представлены в данном Makefile:"
12         @echo "... help (целью по умолчанию)"
13         @echo "... book (сборка книги)"
14 .PHONY: help
```

Листинг 1: Пример *Makefile* для сборки книги, в *GNU make*

Список команд:

	GNU/Linux	*MS Windows*
1	*man*	*Справка и поддержка*
2	*screen*	—
3	*ssh*	*putty*
4	*rsync*	*winSCP*
5	*cron*	*Планировщик заданий*
6	*make (GNU make)*	встроенное в *MS Visual Studio*

Задание: Привести краткое описание к каждой команде, представленной выше, и представить примеры использования данных команд и программ.

Особые пункты отчёта: Отчёт должен содержать следующие элементы: для пункта, связанного с использованием *GNU make*, требуется продемонстрировать сборку всех исходных файлов языка C/C++ в объектные файлы и использование всех объектных файлов для линковки исполнимого файла. **Примечание:** файлы .c и .h предоставляет преподаватель.

Контрольные вопросы:

1. Для чего нужны команды и программы, представленные выше?

2. Как и с помощью чего можно автоматизировать распределение и выполнение программ и команд в операционных системах? Приведите конкретный пример автоматизации.

4. Лабораторная работа №2: «Среды и средства разработки»

Я не должен бояться.
Страх — убийца разума.
Страх — это маленькая
смерть, влекущая за собой
полное уничтожение.
Я встречусь лицом к лицу
со своим страхом.
Я позволю ему пройти через
меня и сквозь меня.
И, когда он уйдет, я обращу
свой внутренний взор на его
путь.
Там, где был страх, не будет
ничего.
Останусь лишь я.

«Литания против страха — Дюна» Франклин Патрик Герберт-младший

Список компиляторов:

1. GCC;
2. minGW;
3. GCC (cygwin);
4. Clang.

Список сред разработки:

1. MS VisualStudio;
2. Code::Blocks / Code::Lite / Qt Creator.

Задание: Требуется написать программу, на вход которой поступает четыре значения (*A, B, C, D*), операции между операндами требуется получить у преподавателя. **Примечание:** сборка исполнимого файла должна осуществляться через *GNU make*.

Особые пункты отчёта: Отчёт должен содержать следующие элементы:

1. в приложении исходный код программы;
2. содержание *Makefile*.

Контрольные вопросы:

1. Что такое *«кроссплатформенность»*? Может ли исполнимый файл (программа) запуститься на вычислительных платформах с разной архитектурой (x86, SPARC, ARM).
2. В чём заключается отличие сборки программы через *GNU make* от сборки через скрипт (например используя *Bash — «GNU Bourne-Again SHell»*)? Каждый ли раз выполняются все команды в скрипте?
3. Одинаковой ли эффективностью обладают компиляторы? Может ли одна программа (исходный код), собранная разными компиляторами, выполняться за разное время? С чем может быть связана разница?

5. Лабораторная работа №3: «Оформление документов в издательской системе TeX»

Формально правильно, а по сути издевательство.

«Заключительное слово по докладу о продовольственном налоге, X Всероссийская конференция РКП(б), 27 мая 1921.» Владимир Ильич Ленин

Список программ:

1. TeX Live;
2. Texmaker / Texstudio / Kile;
3. Emacs.

Задание:

1. оформить предыдущий отчёт с помощью издательской системы LaTeX;
2. создать презентацию по теме, выбранной вами или полученной у преподавателя, с помощью издательской системы LaTeX, используя пакет Beamer.

Особые пункты отчёта: Отчёт должен содержать следующие элементы:

Документы должны быть оформлены строго с требуемыми параметрами оформления, описанные в данных методических указаниях.

Презентация должна быть составлена из не менее чем пяти слайдов и содержать следующие пункты:

- титульный слайд с автором и темой (выбор темы на ваше усмотрение);

- несколько рисунков;
- многоуровневые списки (нумерованный и ненумерованный);
- таблица (минимум 3x3);
- слайд с описанием выбранной темы презентации.

Оформление документов проводится в Libreoffice / MS Word с использованием стилей.

Контрольные вопросы:

1. Для чего создали и используют системы компьютерной вёрстки *TeX*?
2. Какие системы и программы используются для создания научных журналов и книг?
3. Какие системы компьютерной верстки библиографии существуют?
4. Какие способы сортировки библиографических списков существуют?

6. Лабораторная работа №4: «Система управления пакетами ОС Debian GNU/Linux»

Ни один человек не борется против свободы, — борется человек, самое большее, против свободы других.

«Дебаты о свободе печати и об опубликовании протоколов сословного собрания» (апрель 1842 г.).— К. Маркс и Ф. Энгельсъ. Сочинения. Изд. 2-е. Т. 1, с. 55.»

Теоретическая справка: *Репозиторий, хранилище* — место, где хранятся и поддерживаются какие-либо данные. Чаще всего данные в *репозитории* хранятся в виде файлов, доступных для дальнейшего распространения по сети.

Существуют репозитории для хранения программ, написанных на одном языке (например, *CPAN* для *Perl*) или предназначенных для одной платформы. Многие современные *ОС*, такие как *GNU/Linux*, *OpenSolaris*, имеют официальные *репозитории*, но также позволяют устанавливать пакеты из других мест.

Команды *apt*:

1. *list* — показать список пакетов из указанных имён пакетов;
2. *search* — искать в описаниях пакетов;
3. *show* — показать дополнительные данные о пакете;
4. *install* — установить пакеты;
5. *remove* — удалить пакеты;
6. *autoremove* — автоматически удалить все неиспользуемые пакеты;
7. *update* — обновить список доступных пакетов;

8. *upgrade* — обновить систему, устанавливая/обновляя пакеты;
9. *full-upgrade* — обновить систему, удаляя/устанавливая/обновляя пакеты;
10. *edit-sources* — редактировать файл с источниками пакетов.

Задание:

1. изучить интерфейс командной строки *apt*;
2. изучить базовые возможности пакетного менеджера *dpkg*;
3. настроить *ОС Debian GNU/Linux* на получение пакетов из локального сервера обновления (*http://linuxupdate.asu.ru*) *Алтайского государственного университета*, путём внесением правки в файл *sources.list*.

.

Особые пункты отчёта: Отчёт должен содержать следующие элементы:

1. пример использования команд и их описание;
2. изменённый файл *sources.list*.

Контрольные вопросы:

1. Как осуществляется добавление пакета в репозитарий?
   - Как сделать собственный дистрибутив (набор пакетов)?
   - Как добавить свой пакет (программе) в существующий дистрибутив (например в **Debian GNU/Linux**)?
2. Как осуществляется проверка подлинности пакета?

7. Лабораторная работа №5: «Сеть в ОС GNU/Linux, MS Windows»

Стояли звери
Около двери,
В них стреляли,
Они умирали.

«Жук в муравейнике»
Аркадий и Борис
Стругацкие

Список используемых команд:

	GNU/Linux	*MS Windows*
1	*ip*	*ipconfig*
2	*dig*	—
3	*nslookup*	*nslookup*
4	*ping / ping6*	*ping*
5	*traceroute / traceroute6*	*tracert*
6	*nmap*	—

Задание: познакомиться с командами, представленными выше, в операционных системах: **Debian GNU/Linux** и **MS Windows**. Выполнить следующие пункты задания:

1. ознакомиться с документацией на данные команды с помощью команды *man*;
2. определить *ip*-адрес используемой машины через команду *ip*;
3. с помощью команды *dig* найти *ip*-адреса нескольких сайтов внутри университетской сети;
4. воспользуйтесь командой *nslookup*;
5. провести *ping* до найденных адресов;
6. с помощью *traceroute* осуществить трассировку до найденных *ip*-адресов;
7. изучить основные возможности пакета *nmap*.

```
~$ man COMMAND_NAME
...
~$ ip a
...
~$ dig @10.0.0.33 www.asu.ru
...
~$ nslookup www.asu.ru
...
~$ ping IP_ADDRESS_OF_WWW.ASU.RU
...
~$ traceroute IP_ADDRESS_OF_WWW.ASU.RU
...
~$ nmap -A -T4 IP_ADDRESS_OF_WWW.ASU.RU
...
```

Листинг 2: Пример команд

Особые пункты отчёта: составить краткую схему университетской сети.

Контрольные вопросы:

1. Что такое *ip-адрес* и чем он отличается от *«domain name»*?
2. В чём заключаются отличия IPv4 от IPv6?
3. Как работают команды, представленные в данной лабораторной работе?

8. Лабораторная работа №6: «Виртуализация ОС»

> Замечательное чувство — знать, что ты сам строишь мир.
>
> *Айзек Азимов*

Теоретическая справка: *Гипервизор* (англ. Hypervisor; от др.-греч. ὑπέρ «над, выше, сверх» + лат. vīsio «зрение; видение») или *монитор виртуальных машин* (в компьютерах) — программа или аппаратная схема, обеспечивающая или позволяющая одновременное, параллельное выполнение нескольких операционных систем на одном и том же хост-компьютере.

Список гипервизоров:

1. *Hyper-V*;
2. *KVM*;
3. *VirtualBox*;
4. *Xen*.

Для работы с *KVM* вам могут потребоваться следующие команды:

1. virt-install [OPTIONS]... — команда для создания виртуальной машины;
2. virsh list — вывести на экран список всех машин;
3. virsh start MACHINE_NAME — включить виртуальную машину;
4. virsh shutdown MACHINE_NAME — выключить виртуальную машину;
5. virsh destroy MACHINE_NAME — уничтожить виртуальную машину;
6. virsh edit MACHINE_NAME — редактирование конфигурационного фала виртуальной машины;

7. virsh undefine MACHINE_NAME — убрать из списка существующих машин;

8. virsh vncdisplay MACHINE_NAME — показать, какой порт использует VNC;

9. virt-viewer MACHINE_NAME — подключиться к виртуальной машине через VNC.

```
virt-install --name MACHINE\_NAME \
             --memory 512 \
             --arch x86_64 \
             --vcpus 1 \
             --cpu host-model-only \
             --cdrom /path_to_iso/debian-netinst.iso \
             --boot=cdrom,hd \
             --disk=path=/path_to_img/example2018.img,\
             size=5,bus=virtio,format=qcow2 \
             --graphics vnc,listen=0.0.0.0,password=12345
```

Листинг 3: Пример команды для создания виртуальной машины с 512 Мб памяти и 64-битной архитектурой

Другой способ создания виртуальной машины — это воспользоваться командой *virt-manager*.

Задание:

1. познакомиться с представленными выше гипервизорами;

2. создать виртуальную машину и произвести установку *«гостевой»* системы;

3. произвести настройку *«гостевой»* системы.

Особые пункты отчёта: Отчёт должен содержать следующие элементы:

Контрольные вопросы:

1. Что такое *«эмуляция»* и что такое *«виртуализация»*?
2. Отличия *«программной виртуализации»* от *«аппаратной виртуализации»*?

Библиографический список

В данном разделе представлен краткий список литературы, которую рекомендуется использовать при выполнении лабораторных работ:

1. Калачев А.В. Многоядерные процессоры: учебное пособие / А.В. Калачев. — М.: Интернет-Университет Информационных Технологий: БИНОМ. Лаборатория знаний, 2010. — 247 с.: ил., табл. — (Основы информационных технологий). ISBN 978-5-9963-0349-6

2. Левин М.П. Параллельное программирование с использованием OpenMP: учебное пособие / М.П. Левин. — М.: Интернет-Университет Информационных Технологий: БИНОМ. Лаборатория знаний, 2010. — 118 с.: ил., табл. — (Серия «Основы информационных технологий»). ISBN 978-5-94774-857-4

3. Дональд Эрвин Кнут. Искусство программирования, том 1. Основные алгоритмы, 3-е изд.: Пер. с англ.: Уч. пос. — М.: Издательский дом «Вильямс», 2000. — 720 с.: ил. — Парал. тит. англ. ISBN 5-8459-0080-8 (рус.)

4. Орлов С.А. Технологии разработки программного обеспечения. Учебное пособие. 2-е изд. / А.С. Орлов. — СПб.: Питер, 2003. — 480 с.: ил. ISBN 5-94723-145-X

5. Немнюгин С.А., Стесик О.Л. Современный Фортран. Самоучитель. — СПб.: БХВ-Петербург, 2004. — 496 с.: ил. ISBN 5-94157-302-2

6. Немнюгин С.А., Стесик О.Л. Параллельное программирование для многопроцессорных вычислительных систем. — СПб.: БХВ-Петербург, 2002. — 400 с.: ил. ISBN 5-94157-188-7

7. Абрамовиц М. и др. Справочник по специальным функциям. — Наука, 1979.

8. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. — СПб.: Питер, 2015. — 1120 с.: ил. — (Серия «Классика computer science»). ISBN 978-5-496-01395-6

9. Робачевский А.М. Операционная система UNIX®. — СПб.: 2002. — 528 с.: ил. ISBN 5-8206-0030-4

10. Г. В .Курячий, К. А. Маслинский. Операционная система Linux: Курс лекций. Учебное пособие. — М. : ALT Linux; Издательство ДМК Пресс, 2016. — 348 с.: ил.; 2-е изд., исправленное.— (Библиотека ALT Linux). ISBN 978-5-97060-390-1

11. Моргунова, О.Н., Тынченко В.В. Операционная система FreeBSD. Вводный курс: учеб. пособие; Сиб. гос. аэрокосмич. ун-т. — Красноярск, 2011. — 132 с.

12. Руссинович М., Соломон Д. Внутреннее устройство Microsoft Windows. 6-е изд. — СПб.: Питер, 2013. — 800 с.: ил. — (Серия «Мастер-класс»). ISBN 978-5-459-01730-4

Подписано в печать 04.06.2018. Формат 60x84/16
Усл.-печ. л. 1,08 Тираж 50 экз. Заказ № 299.
Типография Алтайского государственного университета:
656099, Барнаул, ул. Димитрова, 66