**Правительство Российской Федерации**

**Федеральное государственное автономное образовательное учреждение
высшего профессионального образования**

**Национальный исследовательский университет**

**«Высшая школа экономики»**

Факультет **КОМПЬЮТЕРНЫХ НАУК**

**Программа дисциплины**

**Архитектура компьютеров
и системное программирование**

для направления 01.03.02 «**Прикладная математика и информатика»** подготовки **бакалавров**

Авторы: А.А. Незнанов (aneznanov@hse.ru),
 А.В. Чернов (achernov@hse.ru)

|  |  |
| --- | --- |
| Рекомендована секцией УМС«Прикладная математика и информатика»Председатель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кузнецов С.О.«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г. | Одобрена на заседании департаментаанализа данных и искусственного интеллектаЗав. департамента\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кузнецов С.О.«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г. |
| Утверждена УС факультетакомпьютерных наукУченый секретарь\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ « \_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. |  |

Москва

# Пояснительная записка

## Авторы программы

кандидат технических наук, доцент А.А. Незнанов;

кандидат технических наук, доцент А.В. Чернов.

## Требования к студентам

Изучение дисциплины «Архитектура компьютеров и системное программирование» требует предварительных знаний в следующих областях: математическая логика, дискретная математика, основы информатики, программирование на языках высокого уровня, теория информации и кодирования. Желательно знакомство с основами современных телекоммуникаций.

## Аннотация

Дисциплина «Архитектура компьютеров и системное программирование» предназначена для подготовки бакалавров по направлению 01.03.02 – Прикладная математика и информатика.

Понимание устройства и принципов работы электронных вычислительных машин (ЭВМ), которые далее мы будем назвать уже привычным термином «компьютер», является незаменимым для инженеров в сфере информационных технологий при:

* оценке проектных решений, особенно комплексных и затрагивающих различные уровни абстракции, например, выборе систем виртуализации и платформ развёртывания;
* проектировании новых систем для достижения компромисса стоимость-производительность-надежность;
* выявлении узких мест в работе существующих информационных систем;
* диагностики нетривиальных неполадок и ошибок;
* повышении производительности вычислений за счет более эффективного использования программных и аппаратных средств;
* построении инновационных программно-аппаратных систем: систем интеллектуального анализа данных, роботов, инструментов телеприсутствия и дополненной реальности.

Цель дисциплины – помочь развитию незаменимого в профессиональной деятельности «инженерного чутья», которое рождается из понимания принципов работы компьютеров и деталей взаимодействия прикладного и системного программного обеспечения (ПО) с аппаратурой ЭВМ. Очевидно, что кроме теоретических оснований, дисциплина предусматривает изучение конкретных реализаций аппаратного обеспечения, операционных систем и т.п. Конкретно, будут рассмотрены следующие разделы.

1. Модели вычислений и вычислительные архитектуры. Платформы и парадигмы.
2. Кодирование информации и форматы данных. Передача сигналов и хранение данных в цифровых электронно-вычислительных машинах.
3. Аппаратура современных компьютеров. Интегральные схемы. Интерфейсы сопряжения: *RS-232*, *JTAG*, *USB*, *IEEE1394* и др.
4. Уровни абстракции – от микроархитектуры ядра центрального процессора до распределённых гетерогенных информационных систем. Роль базовых систем ввода/вывода и операционных систем.
5. Наиболее популярные архитектуры процессоров общего назначения. Подробное рассмотрение архитектур *x86* (от *8086/8088* до *AMD64/EM64T*), *ARM*, *AVR*.
6. Базовые проектные решения при создании операционных систем для достижения высокой производительности, надёжности, информационной безопасности.
7. Инструментарий *GNU* для инженера – системного программиста.
8. Основные сервисы операционных систем семейств *Linux* и *Microsoft Windows* и их правильное использование в соответствии с рекомендациями производителей.

## Учебные задачи дисциплины

Дисциплина должна сформировать представление о проблемах разработки ПО на разных уровнях абстракции при оптимальном использовании возможностей аппаратного обеспечения. В результате изучения дисциплины «Архитектура компьютеров и системное программирование» студенты должны:

1. знать наиболее важные проектные решения на всех уровнях абстракции, накопленные в процессе развития компьютерной техники, и основные параметры архитектур *x86* и *ARM*, строение операционных систем общего назначения и базовые сервисы операционных систем семейств *Linux* и *Microsoft Windows*;
2. понимать связь свойств аппаратного обеспечения современных цифровых компьютеров с проблемами системного программирования;
3. уметь применять язык ассемблера для платформ *x86* и *ARM* при создании системного ПО и использовать базовые сервисы операционных систем семейств *Linux* и *Microsoft Windows*.

## Развиваемые компетенции

Дисциплина формирует следующие компетенции.

1. ПК-8: способность решать задачи производственной и технологической деятельности на профессиональном уровне, включая разработку математических моделей, алгоритмических и программных решений.
2. ПК-9: способность применять в профессиональной деятельности современные языки программирования и языки баз данных, операционные системы, электронные библиотеки и пакеты программ, сетевые технологии и т.п.
	* ПК-9-СП1: способность сравнивать и давать рекомендации по использованию аппаратных и программных архитектур информационных систем.
	* ПК-9-СП2: способность разрабатывать программное обеспечение, взаимодействующее с аппаратурой ЭВМ и сервисами операционных систем.
	* ПК-9-СП3: способность использовать языки ассемблера для архитектур *x86* и *ARM*.

# Тематический план дисциплины «Архитектура компьютеров и системное программирование»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Название темы** | **Всего часов по дисциплине** | **Аудиторные часы** | **Самосто-ятельная работа** |
| **Лекции** | **Сем. и практика занятия** |
| 1 | Цифровые электронно-вычислительные машины. Основные концепции и история развития. | 26 | 6 | 4 | 16 |
| 2 | Аппаратура современного компьютера. Архитектура *x86*: история развития, аппаратная реализация и язык ассемблера. | 58 | 8 | 10 | 40 |
| 3 | Операционные системы: основы разработки и история развития. | 32 | 4 | 4 | 24 |
| 4 | Семейство операционных систем *Microsoft Windows*: базовые сервисы и использование *Windows API*. | 66 | 12 | 12 | 42 |
| 5 | Архитектура *ARM*. Низкоуровневое программирование на примере*Raspberry PI*. | 72 | 12 | 10 | 50 |
| 6 | Семейство операционных систем *Unix*. Инструментарий *GNU*. | 88 | 16 | 16 | 56 |
|  | Итого | 342 | 58 | 56 | 228 |

# Учебно-методическое и информационное обеспечение дисциплины

## Список литературы

### Базовые учебники

1. Харрис Д.М., Харрис С.Л. Цифровая схемотехника и архитектура компьютера, второе издание. – Издательство Morgan Kaufman, 2013. По согласованию с издательством Elsevier, *ISBN* 978-0-12-394424-5. – 1662 c.
2. Таненбаум Э., Остин Т. Архитектура компьютера, 6-е изд. – СПб: Питер, 2013. – 816 с.
3. Таненбаум Э., Вудхалл А. Операционные системы. Разработка и реализация. – СПб.: Питер, 2007. – 704 с.

### Основная литература

1. Руссинович М., Соломон Д. Внутреннее устройство *Microsoft Windows*. Часть 1. – СПб.: Питер, 2013. – 800 с.
2. Гордеев А.В., Молчанов А.Ю. Системное программное обеспечение. – СПб: Питер, 2001. – 736 с.
3. Рихтер Дж., Назар К. *Windows via C/C++*. Программирование на языке *Visual C++*. – СПб: Питер, 2008. – 896 с.
4. Рихтер Дж. *Windows* для профессионалов: создание эффективных Win32-приложений с учетом специфики 64-разрядной версии Windows. – СПб: Питер, 2001. – 752 с.
5. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 2.0 на языке C#. – СПб: Питер, 2007. – 656 с.
6. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы. 5-е изд. – СПб: Питер, 2011. – 944 с.
7. Лав Р. Разработка ядра Linux. – Вильямс, 2008. – 448 с.
8. Раго С.А., Стивенс У.Р. *UNIX*. Профессиональное программирование. 3-е издание. – М.: Символ-Плюс, 2014. – 1104 c.
9. Немет Э., Снайдер Г., Хейн Т.Р., Уэйли Б. *Unix* и *Linux*. Руководство системного администратора. – Вильямс, 2012. – 1312 с.
10. *Kusswurm D*. *Modern X86 Assembly Language Programming: 32-bit, 64-bit, SSE, and AVX*­. – *Apress*, 2014. – 685 p.
11. *Bloom J*. *Exploring Arduino: Tools and Techniques for Engineering Wizardry*. – *Wiley*, 2013. – 384 p.
12. *Arduino* (<http://www.arduino.cc>)
13. Аппаратная платформа *Arduino* | *Arduino.ru* (<http://arduino.ru>)
14. *Windows API Index*

([https://msdn.microsoft.com/en-us/library/windows/desktop/hh920508(vs.85).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/hh920508%28vs.85%29.aspx))

1. *Windows Sysinternals* (<http://technet.microsoft.com/ru-ru/sysinternals>)
2. *x86 architecture reference documents* (<http://www.sandpile.org/x86/ref_docs.htm>)
3. *Intel® 64 and IA-32 Architectures Software Developer Manuals* (<http://www.intel.in/content/www/in/en/processors/architectures-software-developer-manuals.html>)
4. *ARMv8-A Architecture – ARM* (<http://www.arm.com/products/processors/armv8-architecture.php>)

### Дополнительная литература

1. Причард Дж. COM и CORBA. Просто и доступно. – Лори, 2001. – 384 с.
2. ГОСТ Р ИСО/МЭК ТО 12182-2002 «Информационная технология. Классификация программных средств».
3. *Microsoft Learning center* ([www.microsoft.com/en-us/server-cloud/support/learning-center/learning-center.aspx](http://www.microsoft.com/en-us/server-cloud/support/learning-center/learning-center.aspx))
4. *Myers P.J. Inline Assembly Language Manual* (<http://www.pmzone.org/toc.html>)
5. *CIT Forum*: операционные системы (<http://citforum.ru/operating_systems>)
6. *INTUIT.RU*: Интернет-Университет Информационных Технологий: Операционные системы (<http://www.intuit.ru/studies/courses?service=0&option_id=6&service_path=1>)

# Формы контроля и структура итоговой оценки

Текущий контроль – 1 письменная контрольная работа (80 мин.),
5 домашних работ.

Промежуточный контроль – 1 экзамен (в конце второго модуля);

Итоговый контроль – 1 экзамен (в конце третьего модуля);

Итоговая оценка складывается из следующих элементов:

*Отекущий* = 0,1*·Од/з1 +* 0,1*·Од/з2 +* 0,15*·Од/з3 +* 0,2*·Од/з4 +* 0,2*·Од/з5 + 0,15·Ок/р+ 0,1·Оаудиторная*

*Одисциплина = 0,2·Оэкзамен2 + 0,15·О экзамен1 + 0,65·Отекущий*

Д/з сдаются (в виде предоставления отчёта в электронной форме) не позднее установленных преподавателем сроков (сроки зависят от графика учебного процесса и объявляются при выдаче заданий). Д/з защищается путём демонстрации отчёта и ответов на вопросы преподавателя по отчёту.

На пересдаче неудовлетворительной оценки за д/з студенту предоставляется возможность получить не более 3 дополнительных баллов для компенсации оценки за текущий контроль. Дата пересдачи определяются преподавателем (зависят от графика учебного процесса и объявляются после проверки отчётов по д/з). Пересдача д/з допускается только один раз.

### Таблица соответствия оценок по десятибалльной и системе зачет/незачет

|  |  |
| --- | --- |
| **Оценка по 10-балльной шкале** | **Оценка по 5-балльной шкале** |
| 1 | незачет |
| 2 |
| 3 |
| 4 | зачет |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| 10 |

### Таблица соответствия оценок по десятибалльной и пятибалльной системе

|  |  |
| --- | --- |
| **По десятибалльной шкале** | **По пятибалльной системе** |
| 1 – неудовлетворительно2 – очень плохо3 – плохо | неудовлетворительно – 2 |
| 4 – удовлетворительно5 – весьма удовлетворительно | удовлетворительно – 3 |
| 6 – хорошо7 – очень хорошо | хорошо – 4 |
| 8 – почти отлично9 – отлично10 – блестяще | отлично – 5 |

# Программа дисциплины «Архитектура компьютеров и системное программирование»

## Тема 1. Цифровые электронно-вычислительные машины. Основные концепции и история развития.

Вычисления. Модели вычислений. Системы счисления. Кодирование информации. Понятие компьютера. Машина Тьюринга и Равнодоступная адресная машина. Цифровые электронно-вычислительные машины.

Платформы и парадигмы построения компьютеров. История развития.

Дискретные элементы цифровых электронных схем. Логические элементы. Интегральные схемы. Передача сигналов и хранение информации. Информационные шины. Интерфейсы сопряжения: *RS-232*, *JTAG*, *USB*, *IEEE1394*.

Оборудование: абак, счеты, перфокарты, ЭВМ, микрокалькулятор, программируемый микрокалькулятор, микропроцессор/микроЭВМ, микропроцессорные системы *SoC*, сопроцессоры, специализированные вычислители, *RISC/CISC*, *x86*, *ARM*, наладонные устройства *PDA* (*Newton*, *Palm*, *iPaq* и др.), смартфоны, планшеты.

Общие методы повышения эффективности вычислений. Закон Мура. Конвейеризация и распараллеливание.

Программное обеспечение: ассемблеры, языки программирования высокого уровня, *Unix*, *BSD*, *DOS*, *Windows*, *MacOS*, *GNU*, *Minix*, *Linux*, *iOS*, *Android*, *Windows Mobile*.

Телекоммуникации. Хэш-функции. Электронная цифровая подпись. Системы виртуализации и эмуляторы: *VMWare*, *VirtualBox*, *Hyper-V* и др., эмуляторы *Android*, *iPhone* и др.

### Основная литература

1. Харрис Д.М., Харрис С.Л. Цифровая схемотехника и архитектура компьютера, второе издание. – Издательство Morgan Kaufman, 2013. По согласованию с издательством Elsevier, ISBN 978-0-12-394424-5. – 1662 c.
2. Таненбаум Э., Остин Т. Архитектура компьютера, 6-е изд. – СПб: Питер, 2013. – 816 с.

### Дополнительная литература

1. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы. 5-е изд. – СПб: Питер, 2011. – 944 с.
2. ГОСТ Р ИСО/МЭК ТО 12182-2002 «Информационная технология. Классификация программных средств»

## Тема 2. Аппаратура современного компьютера. Архитектура x86: история развития, аппаратная реализация и язык ассемблера.

Краткий практический обзор, без которого объяснение происходящего в современном цифровом компьютере было бы невозможно: использование логических элементов для построения двоичной арифметики и памяти, регистры, адресация памяти, стек, вызов подпрограмм, прерывания, защита памяти, виртуальная память. Иллюстрация на примере архитектур компании *Intel*.

Гонка производительности *RISC* и *CISC*, кеши, суперскалярность, предсказания ветвлений, многоядерность, микропрограммирование, альтернативы *Intel*.

Эволюция архитектур компании *Intel* от 8008 до *Intel* 64. Языки ассемблера для архитектур *IA*-32 и *Intel* 64. Инструментальные средства программиста. Организация программы. Наборы команд. Описание данных. Вызов подпрограмм. Использование сервисов *BIOS*.

### Основная литература

1. Харрис Д.М., Харрис С.Л. Цифровая схемотехника и архитектура компьютера, второе издание. – Издательство Morgan Kaufman, 2013. По согласованию с издательством Elsevier, *ISBN* 978-0-12-394424-5.
2. Таненбаум Э., Остин Т. Архитектура компьютера, 6-е изд. – СПб: Питер, 2013. – 816 с.

### Дополнительная литература

1. Kusswurm D. Modern X86 Assembly Language Programming: 32-bit, 64-bit, SSE, and AVX. – Apress, 2014. – 685 p.
2. *x86 architecture reference documents* (<http://www.sandpile.org/x86/ref_docs.htm>)
3. *Intel® 64 and IA-32 Architectures Software Developer Manuals* (<http://www.intel.in/content/www/in/en/processors/architectures-software-developer-manuals.html>)
4. *Myers P.J. Inline Assembly Language Manual* (<http://www.pmzone.org/toc.html>)

## Тема 3. Операционные системы: основы разработки и история развития

Цели создания и история развития операционных систем (ОС). Решаемые задачи и требования к ним. Классификация ОС. ОС семейства *Unix*. ОС фирм *Microsoft*, *IBM* и *Apple*.

Загрузка, ядро системы и процессы пользователя. Управление памятью, процессами, вводом-выводом, сетевым взаимодействием, устройствами.

Системные вызовы, интерфейсы программирования *API/ABI*, драйверы/модули ядра, динамическая компоновка программ, файловые системы, сетевые соединения.

Эффективность, надёжность и безопасность ОС. ОС для настольных компьютеров и серверов: требования и особенности. ОС реального времени. Особенности блокировки ресурсов, журналирования и восстановления после сбоев. Управление пользователями, аутентификация, авторизация, квотирование.

ОС и сеть Интернет. Распределённые ОС.

### Основная литература

1. Таненбаум Э., Вудхалл А. Операционные системы. Разработка и реализация. – СПб.: Питер, 2007. – 704 с.

### Дополнительная литература

1. Руссинович М., Соломон Д. Внутреннее устройство Microsoft Windows. Часть 1. – СПб.: Питер, 2013. – 800 с.
2. Лав Р. Разработка ядра Linux. – Вильямс, 2008. – 448 с.
3. Немет Э., Снайдер Г., Хейн Т.Р., Уэйли Б. Unix и Linux. Руководство системного администратора. – Вильямс, 2012. – 1312 с.
4. *CIT Forum*: операционные системы (<http://citforum.ru/operating_systems>)

## Тема 4. Семейство операционных систем *Microsoft Windows*: базовые сервисы и использование *Windows API*

История развития ОС семейства *Windows*. Разрядность (16, 32, 64 бита) и архитектуры процессоров. Развитие ядра ОС семейства *Windows* для настольных компьютеров и серверов. Реализация принципа *HAL*. *Windows API*: основные компоненты и версии.

Получение информации о системе. Основные утилиты системного программиста.

Объекты ядра ОС. Пользовательский режим и режим ядра. Основные инструменты для диагностики и мониторинга ОС *Windows*.

Процессы и потоки. Создание и жизненный цикл процесса. Переменные окружения процесса. Приоритет процесса. Основной поток процесса. Приоритет потока. Диспетчеризация потоков.

Управление памятью. Виртуальная память и логическое адресное пространство процесса. Структура логического адресного пространства. Работа с виртуальной памятью. Работа с кучами. Работа со стеком.

Системный реестр. *INI*-файлы и их ограничения. Работа с реестром: типы данных, структура, основные функции.

Локальные файловые системы. Системы *FAT16, FAT32* и *NTFS*. Атрибуты файлов. Создание и удаление файлов и директорий (папок в локальной файловой системе). Поиск файлов. Файловые потоки.

Синхронизация потоков и доступ к ресурсам. Критические сессии. Синхронизация с использованием объектов ядра. Семафоры и мьютексы. Семейство функций *WaitFor…*. Особенности синхронизации в многопроцессорных системах.

Межпроцессная коммуникация. История развития сервисов межпроцессной коммуникации. Буфер обмена (*Clipboard*). Динамический обмен данными (*DDE* – *Dynamic Data Exchange*). Проецируемые в память файлы (*File Mapping*), включая именованную общую память (*Named shared memory*). Прочие механизмы и варианты их использования.

Динамически загружаемые библиотеки (*DLL*). Импорт и экспорт функций из *DLL*.

### Основная литература

1. Таненбаум Э., Вудхалл А. Операционные системы. Разработка и реализация. – СПб.: Питер, 2007. – 704 с.

### Дополнительная литература

1. Руссинович М., Соломон Д. Внутреннее устройство Microsoft Windows. Часть 1. – СПб.: Питер, 2013. – 800 с.
2. Рихтер Дж., Назар К. Windows via C/C++. Программирование на языке Visual C++. – СПб: Питер, 2008. – 896 с.
3. Рихтер Дж. Windows для профессионалов: создание эффективных Win32-приложений с учетом специфики 64-разрядной версии Windows. – СПб: Питер, 2001. – 752 с.
4. *Windows API Index* ([https://msdn.microsoft.com/en-us/library/windows/desktop/hh920508(vs.85).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/hh920508%28vs.85%29.aspx))
5. *Windows Sysinternals* (<http://technet.microsoft.com/ru-ru/sysinternals>)

##  Тема 5. Архитектура *ARM*. Низкоуровневое программирование на примере

## *Raspberry PI*.

Инструментальные средства: отладчик,

Создание простейших программ для работы с портами ввода/вывода.

### Основная литература

1. Таненбаум Э., Остин Т. Архитектура компьютера, 6-е изд. – СПб: Питер, 2013. – 816 с.

### Дополнительная литература

1. Немет Э., Снайдер Г., Хейн Т.Р., Уэйли Б. Unix и Linux. Руководство системного администратора. – Вильямс, 2012. – 1312 с.

## Тема 5. Семейство операционных систем *Unux*. Инструментарий *GNU*.

Операционные системы типа Unix. История развития ОС семейства *Unix*. Современные варианты *Unix*: *Linux* (в том числе *Android*), *MacOS*, *iOS*. Архитектура ОС типа *Unix*, базовые интерфейсы программирования.

Запуск процессов в основном и фоновом режиме, перенаправления, конвейеры, переменные окружения, подстановки, for, if, &&, ||.

Инструментальные средства разработки в *Unix*. Традиционная идеология разработки приложений. Современные подходы к разработке приложений *Unix*. Makefile и CMake.

Файловые системы в ОС *Unix*, Журналирование ФС и квотирование ФС. ФС *ext2*, *ext3*, *ext4*.

Операционные системы типа Unix. История развития, современные варианты Unix: Linux (в том числе Android), MacOS, iOS. Архитектура ОС типа Unix, интерфейс программирования.

### Основная литература

1. Таненбаум Э., Остин Т. Архитектура компьютера, 6-е изд. – СПб: Питер, 2013. – 816 с.
2. Таненбаум Э., Вудхалл А. Операционные системы. Разработка и реализация. – СПб.: Питер, 2007. – 704 с.

### Дополнительная литература

1. Лав Р. Разработка ядра Linux. – Вильямс, 2008. – 448 с.
2. Раго С.А., Стивенс У.Р. *UNIX*. Профессиональное программирование. 3-е издание. – М.: Символ-Плюс, 2014. – 1104 c.
3. Немет Э., Снайдер Г., Хейн Т.Р., Уэйли Б. *Unix* и *Linux*. Руководство системного администратора. – Вильямс, 2012. – 1312 с.

## Тема 6. Платформа *Arduino*: опыт низкоуровневого программирования.

Обзор альтернативной массовой вычислительной платформы, которая применяется в большинстве мобильных устройств: смартфонах, планшетах. Система команд и язык ассемблера *ARM*.

Программирование *ARM* на примере *Arduino*. Общее описание платформы. История создания. Среда разработки.

Свойства конкретного оборудования, используемого при программировании. Доступ к постоянной и динамической памяти.

Создание простейших программ для работы с портами ввода/вывода.

### Основная литература

1. Таненбаум Э., Остин Т. Архитектура компьютера, 6-е изд. – СПб: Питер, 2013. – 816 с.

### Дополнительная литература

1. Bloom J. Exploring Arduino: Tools and Techniques for Engineering Wizardry. – Wiley, 2013. – 384 p.
2. *ARMv8-A Architecture – ARM* (<http://www.arm.com/products/processors/armv8-architecture.php>).

# Тематика заданий по формам текущего контроля

### Темы домашних работ

1. Домашнее задание 1. Обзор архитектуры какой-либо вычислительной системы или конкретного семейства процессоров. Примеры архитектур:
	1. *Intel* 8080/80286/80386…
	2. *Intel Pentium*
	3. *Intel* 64
	4. *Intel Itanium*
	5. *AMD K5/K6/K7/K8/K9/K10…*
	6. *IBM POWER* 1/2/PC/3/4/5…
	7. *ARM 7/9/11/Cortex A/Cortex M…*
	8. *Atmel AVR*
	9. *MIPS* I/II/III…
	10. *Sun SPARC/SuperSPARC/HyperSPARC/UltraSPARC/SPARC64…*
	11. *HP PA-RISC*
	12. *DEC Alpha EV4/EV5/EV6…*
	13. …
2. Домашнее задание 2. Исследование работы процессора с архитектурой *x86*.
3. Домашнее задание 3. Создание *Windows*-приложения с использованием языка ассемблера.
4. Домашнее задание 4. Создание *Windows*-приложений, демонстрирующих работу с различными сервисами операционной системы.
5. Домашнее задание 5. Создание *Unix*-программ, демонстрирующих работу с различными сервисами операционной системы

### Примеры задач, предлагаемых на контрольных работах

1. Составить подпрограмму на языке Ассемблера.
2. Дописать вызов функции *Windows API* на языке *С++*.

# Вопросы для оценки качества освоения дисциплины

1. Что такое «абстракция»?
2. Как Вы понимаете значение терминов «интерфейс» и «протокол»?
3. Сколько бит содержит байт? Почему?
4. Сколько бит содержит слово? Почему?
5. Что такое «обратный код» представления целых чисел со знаком?
6. Из чего состоит логическая схема?
7. Что такое прерывание?
8. Что такое шина? Какие они бывают?
9. Какие уровни виртуализации вы знаете?
10. Как может проявляться «закон дырявых абстракций»?
11. Что такое суперскалярность?
12. Как реализуется внеочередное выполнение инструкций центральным процессором?
13. Как связана длина конвейера центрального процессора с проблемой предсказания ветвлений?
14. Приведите примеры кэширования в аппаратуре вашего компьютера.
15. Как расшифровывается аббревиатура *BIOS*?
16. Когда появилась первая операционная система?
17. Как можно охарактеризовать строение ядра ОС *Windows*, начиная с версии *NT*?
18. Чем процесс отличается от потока?
19. Что такое *PID* в *Windows*?
20. Кому и как присваиваются приоритеты? Как их можно "посмотреть" и поменять?
21. Что такое логическое адресное пространство? Каков его размер и от чего он зависит?
22. Чем логический адрес отличается от физического?
23. Что такое «виртуальная память»?
24. Как реализуется страничная адресация?
25. Что такое «режим ядра»?
26. Что такое «окружение [*environment*]» процесса? Как классифицируются переменные окружения?
27. Как изменить переменные окружения? При каких условиях это возможно?
28. Что такое «дескриптор безопасности» объекта ядра?
29. Каковы основные особенности файловой системы *NTFS*?
30. Какие символы недопустимы в имени файла в файловой системе *NTFS*?
31. Перечислите основные атрибуты файла.
32. Что такое режим доступа к файлу?
33. При каких условиях файл может быть открыт двумя разными программами?
34. Что такое «маска файла»?
35. Чем папка [*folder*] отличается от директории [*directory*]? Когда эти различия становятся важны?
36. Как устроено пространство имен оболочки?
37. Как *PE*-файл загружается для выполнения?
38. Как компоновщик собирает *PE*-файл?
39. Чем статическая компоновка отличается от динамической?
40. Как происходи импорт/экспорт функций из *DLL*?
41. Какие способы межпроцессной коммуникации вы знаете?
42. Что такое «формат буфера обмена»?
43. Чем именованный проецируемый в память файл отличается от неименованного?
44. Как обеспечивается ограничение доступа пользователей к ресурсам в клиентских версиях ОС *Windows*?
45. Какие меры повышения надёжности применены в ОС *MINIX3*?

# Темы вопросов на экзамене 1

1. Цифровые компьютеры и модели вычислений. Машина Тьюринга и равнодоступная адресная машина. Архитектура Фон-Неймана и Гарвардская архитектура. Что такое алгоритм и программа.
2. Архитектура современных цифровых компьютеров общего назначения. Процессор и оперативная память. Внешние (периферийные) устройства. Методы повышения быстродействия компьютеров. Кэширование и его значение. Уровни доступа к памяти.
3. Уровни абстракции аппаратуры компьютера с точки зрения программиста. Языки программирования и трансляция. Машинные коды и язык ассемблера. Языки высокого и сверхвысокого уровня.
4. Системное и прикладное программное обеспечение (ПО). Свойства ПО. Версия ПО и её значение. Программные продукты и их распространение.
5. Архитектура x86. История развития. Формальные архитектуры *Intel IA-32* и *IA-64*. Основные особенности.
6. Архитектура x86. Регистры центрального процессора и их назначение. Основные исполнительные блоки. Доступ к памяти.
7. Архитектура x86. Расширения набора команд. *SIMD*-расширения *MMX*, *SSE* и др. Дополнительные регистры и их использование.
8. Операционная система. Её назначение и основные характеристики. Базовые сервисы.
9. Исполнение программы. Исполняемые файлы и их структура. Процесс загрузки программы в память. Что может содержаться в исполняемом файле, кроме машинного кода?
10. Многозадачность и её реализация с точки зрения программиста. Процессы и потоки. Диспетчеризация ресурсов.
11. Многозадачность и её реализация с точки зрения программиста. Изоляция процессов. Логическое адресное пространство. Виртуальная память и принципы её работы.
12. Многокомпонентное программирование. История развития многокомпонентного программирования. Основные стандарты и технологии взаимодействия компонентов.
13. Статическая компоновка. Объектные модули, таблицы точек входа и внешних ссылок. Компоновщики и их использование. Инкрементная компиляция.
14. Динамически компонуемые библиотеки (*DLL*) и их использование. Загрузка *DLL*, пути поиска по умолчанию. Экспорт и импорт функций из *DLL*. Именование функций, декорирование, индексация. Функции *API MS Windows* для работы с *DLL*.
15. Сервисы операционной системы для взаимодействия компонентов программного обеспечения. Уровни взаимодействия и его эффективность. Буфер обмена *MS Windows* (*Clipboard*). Форматы данных буфера обмена. Функции *Windows* *API* для работы с буфером обмена.
16. Проецируемые в память файлы в *MS Windows* (*File Mapping*). Создание и управление проекциями. Именованные проекции как инструмент взаимодействия потоков.
17. Телекоммуникации и сети передачи данных. Базовые понятия территориально-распределённых компьютерных систем. Узлы и каналы связи, их свойства. Принципы передачи данных. Классификация сетей передачи данных. Надёжность сетей передачи данных.
18. Семиуровневая модель *OSI* для формализации взаимодействия открытых систем. Сетевые интерфейсы и протоколы. Стек протоколов *TCP/IP*: базовые протоколы и их применение.

# Темы вопросов на экзамене 2

1. .
2. Платформа *Raspberry PI*. Центральные процессоры и их архитектуры. Взаимодействие с периферией. Среда разработки программ.
3. Платформа *Raspberry PI*. Работа с портами ввода/вывода. Примеры программ.

# Методические указания студентам

## Замечания по работе с …

Авторы программы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Незнанов А.А. /

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Чернов А.В. /

# Приложение 1. Методические рекомендации (материалы) преподавателю

Оформляются в виде отдельного приложения! Студентам они не нужны.