

**Федеральное государственное автономное образовательное
учреждение высшего образования
«Московский физико-технический институт
(национальный исследовательский университет)»**

УТВЕРЖДЕНО

**Проректор по учебной работе и
довузовской подготовке**

А.А. Воронов

	Рабочая программа дисциплины (модуля)
по дисциплине:	Архитектура вычислительных систем и языки ассемблера
по направлению:	Информатика и вычислительная техника
профиль подготовки:	Системное программирование и прикладная математика Физтех-школа Прикладной Математики и Информатики кафедра алгоритмов и технологий программирования
курс:	1
квалификация:	бакалавр

Семестр, формы промежуточной аттестации: 2 (весенний) - Дифференцированный зачет

Аудиторных часов: 120 всего, в том числе:

лекции: 0 час.

семинары: 60 час.

лабораторные занятия: 60 час.

Самостоятельная работа: 105 час.

Всего часов: 225, всего зач. ед.: 5

Программу составили:

И.Р. Дединский, старший преподаватель

В.В. Яковлев, канд. физ.-мат. наук, заведующий кафедрой

Программа обсуждена на заседании кафедры алгоритмов и технологий программирования 04.06.2020

Аннотация

Курс посвящен низкоуровневым аспектам разработки программного обеспечения для UNIX-подобных операционных систем, а также отработки навыков написания программ и их тестирования в предельных ситуациях.

В рамках данной дисциплины будут немного затронуто программирование на языках ассемблера под архитектуры компьютеров ARM (32 бит) и x86, - в объеме, минимально необходимом для понимания таких аспектов, как работа с памятью, соглашения о вызовах, и способы системных вызовов.

После прохождения тем про язык ассемблера, оставшаяся часть курса будет посвящена изучению системных вызовов для работы с памятью, файлами, процессами. Особое внимание будет уделено механизмам межпроцессных взаимодействий: сигнала, каналам, разделяемой памяти, и сетевому взаимодействию.

1. Цели и задачи

Цель дисциплины

Познакомить студентов с базовыми принципами организации внутренней организации компьютерных систем, с базовыми принципами организации операционных систем, а также абстракций и интерфейсов, которые предоставляются программисту для взаимодействия с операционной системой.

Задачи дисциплины

Заключается в демонстрации базовых принципов низкоуровневого программирования.

2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
ОПК-1 Способен применять фундаментальные знания, полученные в области физико-математических и (или) естественных наук, и использовать их в профессиональной деятельности	ОПК-1.1 Способен анализировать поставленную задачу, намечать пути ее решения
ОПК-2 Способен использовать современные информационные технологии и программные средства при решении задач профессиональной деятельности, соблюдая требования информационной безопасности	ОПК-2.1 Способен применять современные вычислительную технику и сервисы сети Интернет в области (сфере) профессиональной деятельности

3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

- Основы работы в UNIX- подобных системах;
- основы низкоуровневого программирования;
- основы машинного кода, языков ассемблера;
- различные пути повышения производительности программы.

уметь:

- Создавать программы на языках Си и Ассемблер;
- работать в UNIX- подобных средах;
- создавать программы на языках Си и Ассемблер без использования высокоуровневых библиотек.

владеть:

- Навыками ведения простейших программных проектов в системах контроля версий.

4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

№	Тема (раздел) дисциплины	Трудоемкость по видам учебных занятий, включая самостоятельную работу, час.			
		Лекции	Семинары	Лаборат. работы	Самост. работа
1	Введение. Структура ЭВМ		8	6	12
2	Архитектура процессора		8	6	12
3	Представление информации в памяти ЭВМ		6	6	12
4	Программные сегменты		6	6	12
5	Машинное представление программ		4	6	9
6	Ввод-вывод. Прерывания		6	6	12
7	Иерархия памяти		6	8	12
8	Многомодульные программы		8	8	12
9	Оптимизация программ		8	8	12
Итого часов			60	60	105
Подготовка к экзамену		0 час.			
Общая трудоёмкость		225 час., 5 зач.ед.			

4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

Семестр: 2 (Весенний)

1. Введение. Структура ЭВМ

Уровни абстрактного представления ЭВМ, язык ассемблера. Трансляция и интерпретация программ и команд. Краткое описание устройств ЭВМ и схема их взаимодействия. Структура центрального процессора (ЦП). Регистры, арифметико-логическое устройство, устройство управления. Схема работы ЭВМ.

Оперативная память ЭВМ. Ячейки, адреса, машинные слова, разряды, биты. Двоичное представление информации в ЭВМ, причины выбора такого представления.

2. Архитектура процессора

Иллюстративная архитектура системы команд X86-64. Способы задания операндов. Система команд как важнейшая характеристика ЭВМ. Последовательная реализация X86-64. Основные принципы конвейеризации.

Конвейерная реализация X86-64.

3. Представление информации в памяти ЭВМ

Двоичная система счисления. Шестнадцатеричная нотация. Слова и размеры данных. Представления целых чисел в форме с фиксированной точкой (представление беззнаковых чисел, представление знаковых чисел в дополнительном коде). Особенности сложения и вычитания целых чисел.

4. Программные сегменты

Особенности сегментирования (базирования) адресов в ПК. Префиксы сегментных регистров. Соглашения о выборе сегментных регистров по умолчанию. Описание программных сегментов. Модели памяти.

5. Машинное представление программ

Команды CPU. Основные приемы программирования с использованием CPU. Кодирование программ. Форматы данных. Обращение к данным. Арифметические и битовые операции. Команды управления. Процедуры.

Массивы. Указатели. Строковые команды. Префиксы повторения. использование отладчика. Некорректные ссылки и переполнение буфера. 64-битное расширение IA-32.

6. Ввод-вывод. Прерывания

Использование механизма прерываний для контроля за событиями в ЭВМ, команда прерывания INT.

Системные вызовы ОС для ввода-вывода и завершения программы.

7. Иерархия памяти

Технологии хранения данных. Локальность. Иерархия видов памяти и принцип кэширования. Кэш-память.

Создание кэш-ориентированных программ. Влияние кэш-памяти на производительность.

8. Многомодульные программы

Понятие модульного программирования, независимая трансляция модулей. Структура модуля. Внешние и общие имена, доступ к ним. "Разноязычные" модули, соглашения о связях. Ассемблерные вставки.

9. Оптимизация программ

Возможности и ограничения оптимизирующих компиляторов. Измерение производительности программ.

Уменьшение количества вызовов процедур. Исключение ненужных ссылок в память. Понятие о современном процессоре. Разворачивание циклов. Увеличение степени параллелизма. Результат оптимизации кода.

Ограничители производительности. Производительность памяти. Обнаружение и исключение мест потери производительности.

5. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

Учебная аудитория, оснащенная компьютерами с OS Linux для каждого студента, либо с предустановленной системой виртуализации.

6. Перечень рекомендуемой литературы

Основная литература

1. Современные операционные системы [Текст] / Э. Таненбаум, Х. Бос ; [пер. с англ.] - СПб. Питер, 2016
2. Assembler / В. И. Юров - СПб. Питер, 2007, 2008, 2010
3. Архитектура компьютера [Текст] : [учеб. пособие для вузов] / Э. Таненбаум, Т. Остин ; [пер. с англ. Е. Матвеев] .— 6-е изд. — СПб. : Питер, 2014 .— 816 с

Дополнительная литература

1. Операционная система UNIX [Текст] : учеб. пособие для вузов / А. М. Робачевский .— СПб. : БХВ-Петербург, 2002, 2003 .— 528 с.

7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)

Не используются

8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)

Стандартные средства разработки, входящие в состав ОС Linux.

9. Методические указания для обучающихся по освоению дисциплины (модуля)

Успешное освоение курса требует напряжённой самостоятельной работы студента. В программе курса приведено минимально необходимое время для работы студента над темой. Самостоятельная работа включает в себя проработку учебного материала (по конспектам лекций, учебной и научной литературе), подготовку ответов на вопросы, предназначенных для самостоятельного изучения. Промежуточный контроль знаний проводится в виде письменных опросов.

Экзамен проставляется на основе работы на семинаре и выполнения домашних работ.

Оценка выставляется за выполнение семинарских и домашних заданий, с учетом сроков сдачи.

Внимание: выполнение и сдача задач, разбираемых на семинарских занятиях, и задач домашнего задания, помеченных как “обязательные” (как правило, по одной задаче в неделю) является обязательным условием получения положительной оценки.

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

по направлению:	Информатика и вычислительная техника
профиль подготовки:	Системное программирование и прикладная математика Физтех-школа Прикладной Математики и Информатики кафедра алгоритмов и технологий программирования
курс:	1
квалификация:	бакалавр

Семестр, формы промежуточной аттестации: 2 (весенний) - Дифференцированный зачет

Разработчики:

И.Р. Дединский, старший преподаватель

В.В. Яковлев, канд. физ.-мат. наук, заведующий кафедрой

1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
ОПК-1 Способен применять фундаментальные знания, полученные в области физико-математических и (или) естественных наук, и использовать их в профессиональной деятельности	ОПК-1.1 Способен анализировать поставленную задачу, намечать пути ее решения
ОПК-2 Способен использовать современные информационные технологии и программные средства при решении задач профессиональной деятельности, соблюдая требования информационной безопасности	ОПК-2.1 Способен применять современные вычислительную технику и сервисы сети Интернет в области (сфере) профессиональной деятельности

2. Показатели оценивания компетенций

В результате изучения дисциплины «Архитектура вычислительных систем и языки ассемблера» обучающийся должен:

знать:

- Основы работы в UNIX- подобных системах;
- основы низкоуровневого программирования;
- основы машинного кода, языков ассемблера;
- различные пути повышения производительности программы.

уметь:

- Создавать программы на языках Си и Ассемблер;
- работать в UNIX- подобных средах;
- создавать программы на языках Си и Ассемблер без использования высокоуровневых библиотек.

владеть:

- Навыками ведения простейших программных проектов в системах контроля версий.

3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

1. Реализуйте на языке ассемблера x86 (IA-32) или x86-64 функцию с сигнатурой:

```
extern double my_sin(double x)
```

которая вычисляет значение $\sin(x)$.

Запрещено использовать встроенные тригонометрические инструкции.

Для вычислений используйте известный вам из курса Математического анализа способ разложения функции в ряд. Точность результата должна быть максимально возможной для типа данных double.

2. Программе в аргументе командной строки передается имя файла с бинарными данными в Little-Endian.

Файл хранит внутри себя односвязный список элементов:

```
struct Item {  
    int value;  
    uint32_t next_pointer;  
};
```

Поле value хранит значение элемента списка, поле next_pointer - позицию в файле (в байтах), указывающую на следующий элемент. Признаком последнего элемента является значение next_pointer, равное 0.

Расположение первого элемента списка (если он существует) - строго в нулевой позиции в файле, расположение остальных - случайным образом.

Выведите на экран значения элементов в списке в текстовом представлении.

Используйте отображение содержимого файла на память.

3.Программе задается единственный аргумент - номер TCP-порта.

Необходимо принимать входящие соединения на TCP/IPv4 для сервера localhost, читать данные от клиентов в текстовом виде, и отправлять их обратно в текстовом виде, заменяя все строчные буквы на заглавные. Все обрабатываемые символы - из кодировки ASCII.

Одновременных подключений может быть много. Использовать несколько потоков или процессов запрещено.

Сервер должен корректно завершать работу при получении сигнала SIGTERM.

Указание: используйте неблокирующий ввод-вывод.

4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся

Перечень контрольных вопросов:

1. POSIX: Обычные (тяжелые) процессы (порождение, завершение, ожидание завершения)
2. POSIX: Память: получение, освобождение, malloc/jmalloc
3. POSIX: Легкие процессы (порождение, завершение, ожидание завершения)
4. POSIX: Работа с файлами, дескрипторы, dup/dup2/fork/pthread_create
5. POSIX: IPC: Сигналы. Системные сигналы.
6. POSIX: IPC: Семафоры/мьютексы (mutex)
7. POSIX: IPC: Разделяемая память
8. POSIX: IPC: Очереди сообщений
9. POSIX: обход множества дескрипторов, poll/select, epoll/kqueue
10. POSIX: TCP-сервер, клиент, вызовы установления связи, передачи данных
11. POSIX: UDP-сервер, клиент, вызовы установления связи, передачи данных
12. Устройство ФС и ХД: типы размещения данных: потоком/разделом, примеры
13. Устройство ФС и ХД: типы размещения данных: списком, сжатым списком, примеры
14. Устройство ФС и ХД: типы размещения данных: деревом, примеры
15. Устройство ФС и ХД: Flash-диски, TRIM, ZFS/BTRFS.
16. Устройство ФС и ХД: защита от сбоев (пропажа питания) сравнение обычной, журналируемой и транзакционной ФС
17. Устройство ФС и ХД: RAID (0,1,4,5,6, дополнительный плюс: 2,3), NAS
18. Безопасность в ОС: защита процесса от влияния других процессов:
 - что требуется от архитектуры для реализации защиты?
19. Безопасность в ОС: атака на переполнение буфера, механизм, способы защиты: (W^X-стратегия, с помощью компилятора, рандомизация библиотек, памяти)
20. Аппаратная часть: MBR и BIOS
21. Аппаратная часть: GPT и UEFI
22. Аппаратная часть: Процессор, память, L1/L2 кеш, архитектура фон Неймана, гарвардская
23. Аппаратная часть: устройство (машинный код, шины данных, энергозависимая память)
24. Аппаратная часть: энергонезависимая память, внешние устройства хранения данных (примеры).
25. Аппаратная часть: Механизм прерываний, системные вызовы в i386 и amd64
 - связь прерываний и POSIX-сигналов
26. Аппаратная часть: Виртуальная память: страницы/суперстраницы (hugepages), взаимодействие TLB, L1, L2, памяти, swap/mmap
27. Устройство ОС: Ядро и пользовательское окружение, примеры.
28. Устройство ОС: Задачи ядра, жизненный цикл процесса
29. Устройство ОС: типы ядер, виртуализация, типы операционных систем
30. Устройство ОС: Этапы загрузки операционной системы, loader (диск, сеть), kernel, пользовательские процессы, init
31. Сети: уровни ISO/OSI, TCP/IP
 - Примеры протоколов
32. Сети: TCP/IP: протокол Ethernet, передача данных в локальной сети, MAC, ARP
33. Сети: TCP/IP: протокол IP, передача данных в межсетевом общении, IP, NAT, NETMASK

34. Производительность:

◦ Из каких множителей состоит показатель Время/Программа, на какие части влияет программист? Компилятор? Процессор?

35. Производительность процессора: кеширование

36. Производительность процессора: конвейер

37. Производительность процессора: скалярная/суперскалярная архитектура

38. Производительность процессора: регистровые окна

Примеры билетов:

1. Программа запускается с двумя параметрами: `arg1` — «ключевая фраза», `arg2` — «каталог с файлами».

Программа должна пройти рекурсивно по каталогу `arg2` и зашифровать содержимое всех файлов через XOR с `arg1`

Ссылки тоже должны обрабатываться.

2. Программа запускается с двумя параметрами: `arg1` — «имя программы», `arg2` — «порт». Программа должна открыть соединение на заданный порт и на каждое подключение выдавать клиенту результат работы программы `arg1`

Клиенты должны обрабатываться параллельно.

Критерии оценивания

отлично

10: всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений;

9: систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, правильное обоснование принятых решений;

8: глубокие знания учебной программы дисциплины и умение применять их на практике при решении конкретных задач, правильное обоснование принятых решений;

хорошо

7: твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;

6: знает материал, грамотно излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;

5: знает основной материал, грамотно излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач неточности;

удовлетворительно

4: фрагментарный, разрозненный характер знаний, недостаточно правильные формулировки базовых понятий, нарушения логической последовательности в изложении программного материала, но при этом он владеет основными разделами учебной программы, необходимыми для дальнейшего обучения и может применять полученные знания по образцу в стандартной ситуации;

3: характер знаний достаточен для дальнейшего обучения и может применять полученные знания по образцу в стандартной ситуации;

неудовлетворительно

2: не знает большей части основного содержания учебной программы дисциплины, допускает грубые ошибки в формулировках основных понятий дисциплины и не умеет правильно использовать полученные знания при решении типовых практических задач.

1: не знает формулировок основных понятий дисциплины и не умеет использовать полученные знания при решении типовых практических задач.

5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

При проведении дифференцированного зачета обучающемуся предоставляется 30 минут на подготовку. Обучающиеся могут пользоваться программой дисциплины,