Министерство науки и высшего образования Российской Федерации
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)

Институт прикладной математики и компьютерных наук

УТВЕРЖДЕНО:
Директор
А. В. Замятин

Оценочные материалы по дисциплине

Операционная система UNIX

по направлению подготовки

01.03.02 Прикладная математика и информатика

Направленность (профиль) подготовки:
Математические методы в цифровой экономике

Форма обучения
Очная

Квалификация
Бакалавр

Год приема
2024

СОГЛАСОВАНО:
Руководитель ОП
К.И. Лившиц

Председатель УМК
С.П. Сущенко

Томск – 2024

1. Компетенции и индикаторы их достижения, проверяемые данными оценочными материалами

Целью освоения дисциплины является формирование следующих компетенций:

ОПК-2. Способен использовать и адаптировать существующие математические методы и системы программирования для разработки и реализации алгоритмов решения прикладных задач.

Результатами освоения дисциплины являются следующие индикаторы достижения компетенций:

ИОПК-2.2. Проявляет навыки использования основных языков программирования, основных методов разработки программ, стандартов оформления программной документации.

ИОПК-2.3. Демонстрирует умение отбора среди существующих математических методов, наиболее подходящих для решения конкретной прикладной задачи.

2. Оценочные материалы текущего контроля и критерии оценивания

Элементы текущего контроля:

– лабораторные работы.

Задания для лабораторных работ

1. Лабораторная работа «Скрипт для сборки»

Требуется написать скрипт, совместимый с POSIX Shell, который производит сборку некоторого исходного файла. Таким файлом может быть как программа на языке C/C++, так и документ TeX. Скрипт должен:

1. Возвращать информативный код ошибки (скажем, если исходный файл вообще невозможно откомпилировать).

2. Анализировать текст и находить заготовленный комментарий с именем конечного файла. Комментарий должен быть некоторым ключевым словом, наиболее логично использовать, скажем, &Output:.

3. Сборка должна производиться в временном каталоге, который должен быть создан при помощи утилиты mktemp.

4. Каталог должен быть удалён при любом исходе работы скрипта, включая обработку сигналов, которые требуют немедленного прекращения работы.

5. Рядом с исходным файлом после завершения работы должен появиться конечный файл с именем, как в распознанном комментарии. Все попутные файлы компиляции должны быть удалены вместе со временным каталогом (что очевидно, потому что они не должны покидать пределы временного каталога).

2. Лабораторная работа «Конкурентный контейнер»

Для выполнения работы следует:

- завести некоторый разделяемый том в Docker, который можно подключать к нескольким контейнерам;

- подготовить Dockerfile и скрипт на POSIX Shell, с помощью которых можно создать образ контейнера для конкурентного управления файлами.

Под конкурентным управлением файлами мы подразумеваем следующие действия:

1. Выполняются поочерёдно (бесконечно в цикле) две операции, между которами всегда осуществляется задержка в одну секунду: создание файла в каталоге общего тома и его последующее удаление.

2. Для определения имени файла следует выбрать первое незанятое имя из последовательности 001, 002, 003 и т. д.

3. В файл нужно записать некоторый случайно сгенерированный идентификатор контейнера (контейнер во все свои файлы записывает всегда один и тот же

идентификатор) и порядковый номер созданного файла для этого контейнера (в первый записывается 1, во второй 2 и т. д.).

5. Определение незанятого имени и создание файла с ним должно проводиться как атомарная операция (т. е. этот фрагмент должен быть внутри блока с вызовом flock на некотором служебном файле синхронизации). Атомарная операция должна длиться как можно короче.

Контейнер будет проверяться с запуском в единственном экземпляре, с запуском десяти экземпляров и с запуском 50 экземпляров. Во всех случаях работа должна быть стабильной и исключающей race condition.

3. Лабораторная работа «Оконное приложение на QT#187;

Необходимо разработать любое оконное приложение с использованием библиотеки QT, которое должно удовлетворять следующим требованиям:

- приложение должно содержать как минимум два любых элемента управления;
- приложение должно обрабатывать как минимум одно событие от пользователя;
- функционал приложения не должен встречаться в работах других студентов.

4. Лабораторная работа «Горизонтально масштабируемый сервис»

В рамках работы необходимо разработать сервис по микросервисной архитектуре, отвечающий следующим требованиям:

1. Сервис должен содержать контейнеры не менее двух различных типов и один брокер. В общем случае допускается реализация одного управляющего контейнера, одного брокера и множество тиражируемых контейнеров-работников, запускаемых из общего образа.

2. Сервис должен быть горизонтально масштабируемым. Другими словами, как минимум контейнер одного типа должен быть реплицируемым с балансировки нагрузки на все запущенные реплики.

3. Функциональность сервиса любая под интерес авторов.

Критерии оценивания

Зачёт за лабораторную работу ставится, если программа функционирует в соответствии с требованиями, и автор способен пояснить написанный код.

3. Оценочные материалы итогового контроля (промежуточной аттестации) и критерии оценивания

Экзаменационный билет состоит из трех устных вопросов, необходимо дать на них развёрнутый ответ.

Перечень теоретических вопросов:

1. Введение, история и разновидности.
2. История развития UNIX.
3. Сильные и слабые стороны UNIX.
4. Разновидности UNIX и дистрибутивы GNU/Linux.
5. Пакеты, процессы и файловая система.
6. Понятие пакета, пакетные менеджеры, типы зависимостей между пакетами.
7. Обработка зависимостей между пакетами и её вычислительная сложность.
8. Процессы, зомби, демоны, группы процессов, сигналы, идентификатор процесса pid_t.
9. Управление процессами, функции fork(), exec*(), wait(); команды ps, kill, killall.
10. Структура файловой системы, типы файловых систем, монтирование файловых систем, команда chroot.
11. Типы файлов и права доступа.
12. Управление сетью и инструменты обеспечения безопасности.

13. Сетевые интерфейсы, таблица маршрутизации и беспроводные подключения.
14. Инструменты отладки nmap, ping и traceroute.
15. Установление сетевого подключения.
16. Утилиты подсчёта хэш-сумм.
17. Утилиты gpg и cryptsetup.
18. Контейнерная виртуализация.
19. Технологии контейнерной виртуализации, плюсы и минусы.
20. Docker и его понятия.
21. Сервисы в Docker Compose и Docker Stack.
22. Инструменты оркестрации.
23. «Хитрости» микросервисной архитектуры.
24. Порядок «горячего» обновления контейнеров.
25. Файловые системы для сетевого доступа.
26. Назначение брокеров и их примеры.
27. Сигналы и инструменты межпроцессного взаимодействия.
28. Обработка и блокирование сигналов, функция signalfd().
29. Трубы для потоковой передачи данных.
30. Семафоры, разделяемая память и очереди сообщений.
31. Сервис D-Bus.
32. Оконная система, терминалы и TeX.
33. X Window system и Wayland.
34. Оконные менеджеры, GTK+ и QT.
35. Терминалы и псевдотерминалы.
36. TeX и LaTeX.
37. Структура документа и форматирование текста.
38. Работа с оборудованием в Linux.
39. Каталоги /dev, /proc и /sys.
40. Получение информации о блочных устройствах.
41. Файловая система ext4.
42. Команды lsmod, lsusb и lspci.
43. Протоколирование системных событий.

Критерии оценивания

Результаты определяются оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Оценка «отлично» выставляется, если сданы все лабораторные работы, на все теоретические вопросы даны правильные развернутые ответы.

Оценка «хорошо» выставляется, если сданы все лабораторные работы, но даны ответы не на все теоретические вопросы.

Оценка «удовлетворительно» выставляется, если сданы 2 лабораторные работы, даны неверные или неполные ответы на теоретические вопросы.

Оценка «неудовлетворительно» выставляется, если лабораторные работы не были сданы.

4. Оценочные материалы для проверки остаточных знаний (сформированности компетенций)

Теоретические вопросы:

1. Введение, история и разновидности.
2. История развития UNIX.
3. Сильные и слабые стороны UNIX.
4. Разновидности UNIX и дистрибутивы GNU/Linux.
5. Пакеты, процессы и файловая система.

6. Понятие пакета, пакетные менеджеры, типы зависимостей между пакетами.

7. Обработка зависимостей между пакетами и её вычислительная сложность.

8. Процессы, зомби, демоны, группы процессов, сигналы, идентификатор процесса pid_t.

Ответ должен содержать развернутый ответ на один теоретический вопрос.

Информация о разработчиках

Пожидаев Михаил Сергеевич, канд. техн. наук, доцент кафедры теоретических основ информатики НИ ТГУ.