Лабораторные работы по курсу «Операционные системы»

Дмитрий Тимофеев

11 сентября 2012 года

Содержание

1 Контактная информация

- Преподаватель: Дмитрий Андреевич Тимофеев, кафедра РВКС ФТК.
- Телефон: +7-921-915-95-32
- Электронная почта: `dtim@dcn.ftk.spbstu.ru`

2 Общие требования

Для получения зачета необходимо и достаточно выполнить 4 задания, предполагающих написание программ, и продемонстрировать умение выполнять основные пользовательские задачи в операционной системе Linux (основные операции с файлами и процессами, редактирование текста, компиляция и запуск программ).

Все программы должны работать в операционной системе Linux (в лаборатории используется дистрибутив Debian GNU/Linux 6.0). Чтобы задание было зачтено, необходимо продемонстрировать работу программы и ответить на вопросы, которые могут касаться используемых алгоритмов, структур данных, системных вызовов, библиотечных функций, а также результатов работы программы. Программа должна компилироваться без предупреждений и правильно работать. Все задания должны быть выполнены самостоятельно.

3 Задание 1. Скрипты на языке командного интерпретатора Bash

Напишите на языке командного интерпретатора Bash скрипт, решающий одну из следующих задач.

1. *Резервное копирование данных.* Скрипт получает в качестве аргументов один или несколько путей к файлам или каталогам, архивирует эти файлы (каталоги) и помещает архивы в каталог с резервными копиями. По умолчанию для хранения резервных копий должен использоваться каталог `.backup` в домашнем каталоге пользователя. Предусмотрите возможность задания иного каталога. Новые архивные копии не должны заменять уже существующие.

2. *Копирование структуры каталогов.* Скрипт получает в качестве параметра абсолютный или относительный путь к некоторому каталогу и создает в текущем каталоге иерархию каталогов, которая повторяет иерархию подкаталогов в заданном каталоге. Никакие файлы при этом не копируются.

3. *killall.* Скрипт получает в качестве параметров номер или имя сигнала (с префиксом "-") и список имен, и посылает всем процессам, имена которых совпадают с указанными, этот сигнал. Этот скрипт повторяет функциональность программы `killall`, поэтому ее использовать нельзя. Пример: `killall.sh -TERM xterm gedit` посылает сигнал `SIGTERM` всем выполняющимся экземплярам программ `xterm` и `gedit`.

При указании в командной строке параметров `-h` или `--help` скрипт должен напечатать краткую справку о своем назначении, способе использования и поддерживаемых параметрах командной строки. Необходимо проверять правильность параметров и сообщать пользователю об ошибке, если параметры заданы некорректно.

4 Задание 2. Командный интерпретатор

Напишите простой командный интерпретатор (shell), обладающий следующими базовыми возможностями:

1. Печать приглашения, в котором должен быть указан текущий каталог.
2. Выполнение вводимых пользователем команд.
3. Печать сообщения об ошибке, если введенная команда не может быть выполнена.
4. Возможность изменять текущий каталог (в Bash это можно сделать с помощью команды `cd`).

Кроме этого, интерпретатор должен поддерживать одну (или больше) из следующих дополнительных возможностей:

- Перенаправление ввода-вывода в файл и из файла (аналог операторов `>`, `>>`, `<` в Bash).
- Перенаправление стандартного вывода одного процесса на стандартный ввод другого процесса (аналог оператора `|` в Bash).
- Поддержка переменных окружения и возможность их использования в командах.
- Поддержка запуска программы в фоновом режиме, приостановки и возобновления выполняемой программы, перевода программы из фонового режима на передний план и наоборот.

5 Задание 3. Хранилище данных

Реализуйте простое нереляционное хранилище данных, позволяющее хранить пары *(ключ, значение)*. Программа должна принимать запросы от других процессов и отправлять ответы с помощью сокетов TCP/IP или Unix (Unix domain sockets). Необходимо реализовать операции добавления пары *(ключ, значение)* в хранилище, извлечения значения по ключу, удаления и модификации записи. Операции над записями должны быть атомарными.

Для диспетчеризации запросов используйте вызовы `poll`, `select` или `epoll`, можно также воспользоваться библиотеками *libevent* или *libev*. Для сохранения и чтения данных на диске может быть полезен механизм отображения файлов в память (`mmap`).

6 Задание 4. Параллельная обработка данных

С помощью библиотеки `pthreads` реализуйте параллельный алгоритм решения одной из следующих задач. Предусмотрите аргумент командной строки, позволяющий задать количество создаваемых программой потоков.

1. Составьте частотный словарь употребления различных слов в наборе текстовых файлов. Словом считается любая последовательность, состоящая из символов 'a'–'z' или 'A'–'Z', регистр символов следует игнорировать. Полученный список должен быть упорядочен по убыванию частоты появления слова в файлах. Пути к файлам передаются в качестве аргументов командной строки. Результаты работы программы должны быть напечатаны на стандартный вывод. Если аргумент командной строки соответствует несуществующему файлу или файлу, чтение которого невозможно, этот файл нужно пропустить.

2. Напишите программу для решения задачи коммивояжера. Необходимо прочитать из файла, путь к которому задан в качестве аргумента командной строки, описание задачи в формате TSPLIB[1] и напечатать длину гамильтонова цикла минимальной длины. Достаточно решить задачу только для одного из многих входящих в TSPLIB вариантов постановки задачи коммивояжера (например, для симметричной задачи коммивояжера с явно заданными весами).

7 Рекомендуемая литература

1. Керниган Б., Пайк Р. UNIX. Программное окружение. — М.: Символ-Плюс, 2003
2. Лав Р. Linux. Системное программирование. — СПб.: Питер, 2008
3. Робачевский А. М., Немнюгин С. А., Стесик О. Л. Операционная система UNIX. — 2-е изд. — СПб.: БХВ-Петербург, 2005
4. Стивенс У. Р., Раго С. А. UNIX. Профессиональное программирование. — 2-е изд. — М.: Символ-Плюс, 2007
5. Эндрюс Г. Р. Основы многопоточного, параллельного и распределенного программирования. — М.: Вильямс, 2003
6. Beej's Guide to Unix Interprocess Communication — `http://beej.us/guide/bgipc/`

[1] `http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/`