Расширенное программирование в UNIX

Работу выполнил:
Рыбалко Тимофей Александрович
НБИбд-02-24

Российский университет дружбы народов, Москва,
Россия

Введение

- Освоить работу с файловой системой Linux необходимо.
- Умение работать с git и GitHub важно для IT-специалистов.

Цель работы

- Изучить механизмы межпроцессного взаимодействия и управления ресурсами в UNIX.

- Освоить работу с системными справочными страницами и генерацию случайных данных.

Техническое обеспечение

- Виртуальная машина: VirtualBox или QEMU.
- Операционная система: Linux (дистрибутив Fedora).
- Требования: Intel Core i3-550, 4 ГБ ОЗУ, 80 ГБ свободного места.

Семафоры

- Процессы по очереди захватывают/освобождают ресурс:

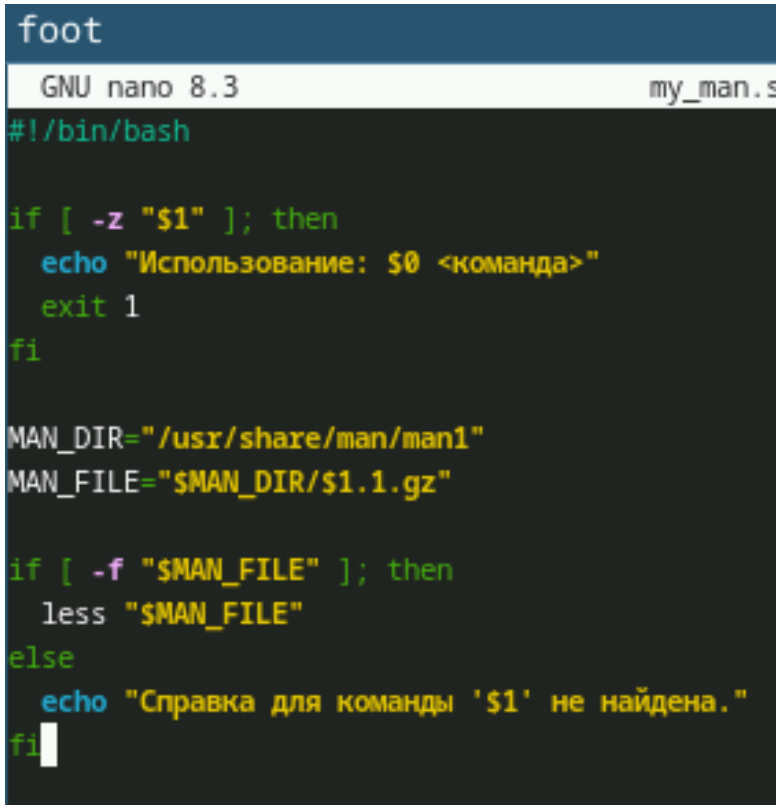
foot

GNU nano 8.3 semaphore.sh

```
#!/bin/bash

LOCK_FILE="/tmp/lab12.lock"
T1=5  # Время ожидания (сек)
T2=3  # Время использования (сек)

# Ожидание освобождения ресурса
echo "Процесс $$ ожидает ресурс..."
while [ -f "$LOCK_FILE" ]; do
  sleep 1
  echo "Ресурс занят, ожидание..."
done

# Захват ресурса
touch "$LOCK_FILE"
echo "Ресурс захвачен процессом $$"

# Использование ресурса
sleep "$T2"

# Освобождение ресурса
rm -f "$LOCK_FILE"
echo "Ресурс освобождён процессом $$"
```

Кастомный man

- Показывает справку или ошибку, если команды нет



Случайные буквы

- Генерирует строку из 15 случайных букв

foot

GNU nano 8.3 random_letters.sh

```
#!/bin/bash

LENGTH=${1:-10}  # Длина последовательности (по умолчанию 10)
ALPHABET="abcdefghijklmnopqrstuvwxyz"

for ((i=0; i<LENGTH; i++)); do
  RAND_INDEX=$((RANDOM % 26))
  echo -n "${ALPHABET:$RAND_INDEX:1}"
done
echo  # Перевод строки
```

Ответы на вопросы:

- 1.Ошибка: отсутствие пробелов внутри [ ] → while ["$1" != "exit"]
- 2.Конкатенация: result="${str1}${str2}"
- 3.seq альтернативы: {1..N} или for ((i=1; i<=N; i++))
- 4.$((10/3)): 3 (целочисленное деление)
- 5.Zsh vs Bash: Zsh - расширенное автодополнение, Bash - стандарт для скриптов
- 6.for синтаксис: верен, если LIMIT определен
- 7.Bash vs Python/C: Плюсы: интеграция с консолью, быстрое написание скриптов; Минусы: слабая типизация, медленные вычисления

Заключение

- Реализованы базовые механизмы синхронизации процессов через файловые семафоры

- Разработаны утилиты для работы с документацией и генерации случайных последовательностей