**Отчет по лабораторной работе №26** по курсу Алгоритмы и структуры данных

Студент группы М8О-107Б-2022 Калуцкий Максим Витальевич, № по списку 13

Контакты e-mail max.kalutskij17@gmail.com

Работа выполнена: «12» .02.2023г.

Преподаватель: Аносова Наталья Павловна каф. 806

Входной контроль знаний с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчет сдан « » \_\_\_\_\_\_\_\_\_202 \_\_ г., итоговая оценка \_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* 1. **Тема:** Абстрактные типы данных?

1. **Цель работы:**  Составить программу выполнения заданных действий над файлами на интерпретируемом командном языке Bash
2. **Задание: Конкатенация файлов наименьшей длины до тех пор, пока размер результирующего файла не превысит размера блока заданного устройства(файловой системы)**
3. **Оборудование** (лабораторное):

*Оборудование ПЭВМ студента, если использовалось:*

Процессор Intel Core i3-4005U CPU с ОП 8096 Мб, НМД \_ 131072\_ Мб. Монитор lenovo

Другие устройства \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Программное обеспечение (лабораторное):**

*Программное обеспечение ЭВМ студента, если использовалось:*

Операционная система семейства \_\_\_Unix\_\_\_\_, наименование \_\_ Ubuntu \_\_\_\_\_\_\_\_\_\_\_\_ версия 4.4.2

интерпретатор команд \_\_\_\_bash\_\_\_\_\_\_\_\_ версия \_\_5.0.17\_\_\_\_.

Система программирования \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_версия \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Редактор текстов \_\_\_\_\_nano\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_25.2.2 \_\_\_\_\_\_\_\_\_

Утилиты операционной системы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Прикладные системы и программы **\_\_\_**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Местонахождение и имена файлов программ и данных на домашнем компьютере \_\_\_\_\_home/jekich228\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Идея, метод, алгоритм** решения задачи(в формах:словесной,псевдокода,графической[блок-схема,диаграмма,рисунок,таблица] или формальные спецификации с пред- и постусловиями)
2. Выбираем два файла с наименьшим размером из всех файлов в директории.
3. Конкатенируем эти два файла в новый файл.
4. Если размер нового файла превышает размер блока файловой системы, то заканчиваем процесс конкатенации файлов и выводим сообщение об окончании.
5. Если размер нового файла не превышает размер блока файловой системы, то повторяем процесс выбора двух файлов с наименьшим размером и конкатенации, пока размер результирующего файла не превысит размер блока файловой системы.
6. **Сценарий выполнения работы** [план работы,первоначальный текст программы в черновике(можно на отдельном листе)итесты либо соображения по тестированию].

Создам исполняемый файл”generator”, который генерирует файлы и заполняет их.

#!/bin/bash

count=$(shuf -i 1-40 -n 1)

for i in $(seq 1 "$count"); do

k=$(shuf -i 80-200 -n 1)

filename="file$i.txt"

tr -dc '[:alnum:]' < /dev/urandom | fold -w "$k" | head -n 1 > "$filename"

done

Этот скрипт на Bash создает случайное количество файлов и заполняет каждый файл случайным набором символов. Давайте подробно рассмотрим каждую строку:

1. **#!/bin/bash**: это шебанг, он сообщает операционной системе, что этот файл является скриптом на языке Bash.
2. **count=$(shuf -i 1-40 -n 1)**: создает переменную **count**, которая содержит случайное число от 1 до 40 с помощью команды **shuf**.
3. **for i in $(seq 1 "$count"); do**: начинает цикл **for**, который будет выполняться **$count** раз.
4. **k=$(shuf -i 80-200 -n 1)**: создает переменную **k**, которая содержит случайное число от 80 до 200 с помощью команды **shuf**.
5. **filename="file$i.txt"**: создает переменную **filename**, которая содержит имя файла, сгенерированного на основе значения переменной **i**.
6. **tr -dc '[:alnum:]' < /dev/urandom | fold -w "$k" | head -n 1 > "$filename"**: использует команду **tr**, чтобы удалить все не буквенно-цифровые символы из потока случайных байтов, генерируемых **/dev/urandom**. Затем используется команда **fold**, чтобы перенести каждый символ на новую строку и создать несколько строк шириной **$k**. И, наконец, используется команда **head**, чтобы выбрать первую из этих строк и сохранить ее в файл **$filename**.
7. **done**: завершает цикл **for**.

Далее пишем сам скрипт, который решает задачу “kek.sh”.

#!/bin/bash

if [ $(ls -S | grep -v "$(basename "$0")"|grep -v "merged\_file" |grep -v "h.sh"|grep -v "generator.sh" | wc -l) -lt 2 ]> echo "Not enough files to merge."

exit 1

fi

while [ $(wc -c merged\_file | awk '{print $1}') -le 4096 ]

do

if [ $(ls -S | grep -v "$(basename "$0")"|grep -v "merged\_file" |grep -v "h.sh"|grep -v "generator.sh" | wc -l) -lt 2> echo "Not enough files to merge."

exit 1

fi

file1=$(ls -S| grep -v "$(basename "$0")"|grep -v "merged\_file"|grep -v "generator.sh"|grep -v "h.sh"| tac | head -n > file2=$(ls -S | grep -v "$(basename "$0")"|grep -v "merged\_file" |grep -v "h.sh"|grep -v "generator.sh" | tac | head >

# Конкатенируем два файла в один

cat "$file1" "$file2" >> merged\_file

# Удаляем исходные два файла

rm "$file1" "$file2"

done

Данный скрипт на bash предназначен для объединения нескольких файлов в один файл до тех пор, пока размер конечного файла не превысит 4 килобайта.

Первые несколько строк предназначены для проверки наличия достаточного количества файлов для объединения. Эти строки используют команды ls, grep и wc, чтобы отфильтровать файлы в текущей директории и определить, сколько файлов необходимо объединить. Если количество файлов меньше двух, выводится сообщение "Not enough files to merge." и скрипт завершается с кодом 1.

Далее следует бесконечный цикл while, который продолжается до тех пор, пока размер конечного файла не превысит 4 килобайта. Цикл содержит условный оператор if, который проверяет, есть ли достаточное количество файлов для объединения. Если количество файлов меньше двух, выводится сообщение "Not enough files to merge." и скрипт завершается с кодом 1.

Далее в цикле определяются два файловых объекта, file1 и file2, которые содержат имена двух наибольших файлов в текущей директории, за исключением некоторых файлов, указанных в grep -v.

Затем команда cat используется для объединения содержимого этих двух файлов в один файл с именем merged\_file, исходные два файла удаляются командой rm.

Цикл продолжается до тех пор, пока размер конечного файла не превысит 4 килобайта.

1. **Распечатка протокола** (подклеить листинг окончательного варианта программы с тестовыми примерами,подписанныйпреподавателем).
2. **Дневник отладки** должен содержать дату и время сеансов отладки, и основные события(ошибки в сценарии и программе,нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  | или |  |  |  |  |  |
|  | дом. |  |  |  |  |  |
|  |  |  |  |  |  |  |

1. **Замечания автора** по существу работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Выводы**: Проделав данную работу, я научился составлять программы выполнения заданных действий над файлами на интерпретируемом командном языке Bash

Недочёты при выполнении задания могут быть устранены следующим образом: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_