**Отчет по лабораторной работе №** 21 по курсу 1

Студент группы М80-108Б-18, Коростелев Д.В. № по списку 12

E-mail [dmitry.k48@yandex.ru](mailto:dmitry.k48@yandex.ru)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Работа выполнена: «28 » февраля 2019 г.

Преподаватель: Поповкин А.В. каф.806 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Отчет сдан « » \_\_\_\_\_\_\_\_\_ 201\_\_ г., итоговая оценка \_\_\_\_\_\_

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

1. **Тема:** Программирование на интерпретируемых командных языках
2. **Цель работы:** составить программу выполнения заданных действий над файлами на одном из интерпретируемых командных языков ОС UNIX
3. **Задание (***вариант №* 12 **):** генерация заданного числа копий указанного файла, Имена копий генерировать добавлением к имени исходного файла следующей по порядку буквы или цифры, начиная с заданной буквы или цифры
4. **Оборудование (лабораторное):**

ЭВМ Cameron, процессор Intel Core2 Duo CPU E8500 3.16ГГц, имя узла сити cameron с ОП 16029 Мб, НДМ 50 Гб. Терминал gnome, адрес 172.16.80.213. Принтер \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

*Оборудование ПЭВМ студента, если использовалось:*

Процессор Intel Core i7-7700HQ 4x 2.808ГГц с ОП 3945 Мб, НДМ 48.9 Гб Монитор \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

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

Операционная система семейства UNIX , наименование UBUNTU версия \_16.04\_\_\_\_\_\_\_\_\_\_\_\_, интерпретатор команд bash версия 4.3.48 , Система программирования \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Редактор текстов \_\_\_\_nano\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_2.9.3\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Утилиты операционной системы: cd, ls, cat, df, dd, du, touch, split, iconv, cmp, comm, wc, diff, grep, sort, tail, tr, uniq, sum, cut, tar, head, file, find, mktemp.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Прикладные системы и программы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Местонахождение и имена файлов программ и данных: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *Программное обеспечение ЭВМ студента, если использовалось:*

Операционная система семейства UNIX, наименование UBUNTU\_\_\_ версия \_\_\_\_\_\_18.04\_\_\_\_\_\_\_, интерпретатор команд \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия\_\_\_\_\_\_4.3.48\_\_\_\_\_\_\_

Система программирования \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Редактор текстов \_nano\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ версия \_2.9.3\_\_\_\_\_\_\_\_\_\_\_\_\_

Утилиты операционной системы: cd, ls, cat, df, dd, du, touch, split, iconv, cmp, comm, wc, diff, grep, sort, tail, tr, uniq, sum, cut, tar, head, file, find, mktemp.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Прикладные системы и программы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Местонахождение и имена файлов программ и данных на домашнем компьютере: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

1. **Идея, метод, алгоритм** решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальные спецификации с пред- и постусловием)

Суть данного задания заключается в том, чтобы нужное количество раз применить команду cp, скопировать файл, однако стоит также задуматься над тем куда приписывать символ, например если файл не имеет расширения в названии то приписывать номер будем в конце файла, если же у файла есть расширение то логичным будет, приписать номер до расширения, в следствии необходимо каким-то образом разбить название файла, на само имя файла и его расширение. Также логичным будет сделать как параметрический ввод данных, таки запрашиваемый, то есть если пользователь ввел нужные параметры после вызова скрипта, программа должна прочитать данные и выполнить свою работу, если же данных нет, то программа сама их запросит ввести. Стоит также задуматься над тем, что при параметрическом вводе данных, пользователь может забыть (специально не написать) написать все необходимые параметры, для этого будут предусмотрены стандартные значения переменных.

1. **Сценарий выполнения работы** (план работы, первоначальный текст программы на черновике или отдельном листе, тесты либо соображения по тестированию)

Сначала программа должна понять были ли введены параметры, если нет, запросить их, также если был введен один параметр, проверить введен ли второй, если нет, то записать в соответствующую переменную ранее предусмотренное значение (речь, о количестве копий и о начальной букве, так как сам скрипт непосредственно работает с файлами). Далее идет главная часть алгоритма, в ней мы с помощью утилиты cut проверяем из чего состоит файл, есть ли в нем расширение, передавая в cut название файла, делить (точку) по которому будем смотреть поля названия файла, получим две переменные. После разделения возможно два варианта: 1) переменные разные, значит в файле есть расширение, делаем копии, вставляя номер копии перед расширением. 2) переменные одинаковые (может произойти, так как cut считает, что если в слове n полей меньше запрашиваемого кол-ва, то утилита вернет последнее поле), значить просто добавляем букву в конец имени файла.

*Пункты 1-7 отчета составляются* ***строго до*** *начала лабораторной работы.*

*Допущен к выполнению работы.* Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Распечатка протокола** (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем)

|  |
| --- |
| dmitry@dmitry-VirtualBox:~/lubs/21$ cat main.sh  #!/bin/bash  if [[ -z $1 ]]; then  echo -n "Введите название файла : "  read name\_of\_file  echo -n "Введите кол-во копий : "  read number\_of\_copy  echo -n "Введите цифру, с которой будут начианться номера копий : "  read first\_number\_of\_copy  else  name\_of\_file=$1  number\_of\_copy=$2  first\_number\_of\_copy=$3  if [[ -z $2 ]]; then  number\_of\_copy=1  fi  if [[ -z $3 ]]; then  first\_number\_of\_copy=1  fi  fi  let "first\_number\_of\_copy = first\_number\_of\_copy - 1"  let "number\_of\_copy = number\_of\_copy + first\_number\_of\_copy"  end\_of\_file=`echo $name\_of\_file | cut -f 2 -d '.'`  start\_of\_file=`echo $name\_of\_file | cut -f 1 -d '.'`  if [[ "$start\_of\_file" == "$end\_of\_file" ]]; then  while [[ $number\_of\_copy > $first\_number\_of\_copy ]]  do  cp $name\_of\_file $start\_of\_file"\_"$number\_of\_copy  let "number\_of\_copy = number\_of\_copy - 1"  done  else  while [[ $number\_of\_copy > $first\_number\_of\_copy ]]  do  cp $name\_of\_file $start\_of\_file"\_"$number\_of\_copy"."$end\_of\_file  let "number\_of\_copy = number\_of\_copy - 1"  done  fi  exit 0  dmitry@dmitry-VirtualBox:~/lubs/21$ cat JustFile  It's just a file with a very simple text  It's gonna be used to test utlite  Let's go!  dmitry@dmitry-VirtualBox:~/lubs/21$ cat TextScript.sh  #!/bin/bash  echo "Hello world!"  dmitry@dmitry-VirtualBox:~/lubs/21$ cat Dir/FILE2  dmitry@dmitry-VirtualBox:~/lubs/21$ ./main.sh  Введите название файла : JustFile  Введите кол-во копий : 2  Введите цифру, с которой будут начианться номера копий : 2  dmitry@dmitry-VirtualBox:~/lubs/21$ ls  1\_3.txt 1\_4.txt 1.txt Dir JustFile JustFile\_2 JustFile\_3 JustFile\_4 main.sh prot21.txt TextScript.sh  dmitry@dmitry-VirtualBox:~/lubs/21$ rm \*\_\*  dmitry@dmitry-VirtualBox:~/lubs/21$ ls  1.txt Dir JustFile main.sh prot21.txt TextScript.sh  dmitry@dmitry-VirtualBox:~/lubs/21$ ./main.sh  Введите название файла : ^Z  [3]+ Stopped ./main.sh  dmitry@dmitry-VirtualBox:~/lubs/21$ rm 1.txt  dmitry@dmitry-VirtualBox:~/lubs/21$  dmitry@dmitry-VirtualBox:~/lubs/21$ ./main.sh  Введите название файла : JustFile  Введите кол-во копий : 2  Введите цифру, с которой будут начианться номера копий : 2  dmitry@dmitry-VirtualBox:~/lubs/21$ ls  Dir JustFile JustFile\_2 JustFile\_3 main.sh prot21.txt TextScript.sh  dmitry@dmitry-VirtualBox:~/lubs/21$ ./main.sh TextScript.sh 3 4  dmitry@dmitry-VirtualBox:~/lubs/21$ ls  Dir JustFile JustFile\_2 JustFile\_3 main.sh prot21.txt TextScript\_4.sh TextScript\_5.sh TextScript\_6.sh TextScript.sh  dmitry@dmitry-VirtualBox:~/lubs/21$ ./main.sh Dir/FILE2 1 1  dmitry@dmitry-VirtualBox:~/lubs/21$ ls Dir/  FILE2 FILE2\_1  dmitry@dmitry-VirtualBox:~/lubs/21$ ./main.sh JustFile\_2 2  dmitry@dmitry-VirtualBox:~/lubs/21$ ls  Dir JustFile JustFile\_2 JustFile\_2\_1 JustFile\_2\_2 JustFile\_3 main.sh prot21.txt TextScript\_4.sh TextScript\_5.sh TextScript\_6.sh TextScript.sh  dmitry@dmitry-VirtualBox:~/lubs/21$ ./main.sh TextScript\_5.sh  dmitry@dmitry-VirtualBox:~/lubs/21$ ls  Dir JustFile JustFile\_2 JustFile\_2\_1 JustFile\_2\_2 JustFile\_3 main.sh prot21.txt TextScript\_4.sh TextScript\_5\_1.sh TextScript\_5.sh TextScript\_6.sh TextScript.sh |

1. **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. или дом. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  |  |  |  |  |  |  |

1. **Замечания автора по существу работы:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. **Выводы:** в итоге смогли составить программу выполнения заданных действий над файлами на одном из интерпретируемых языков ОС UNIX, разобрали структуру интерпретируемых языков, способы программирования скриптов.

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

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