Лабораторная работа №12

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Ким Михаил Алексеевич

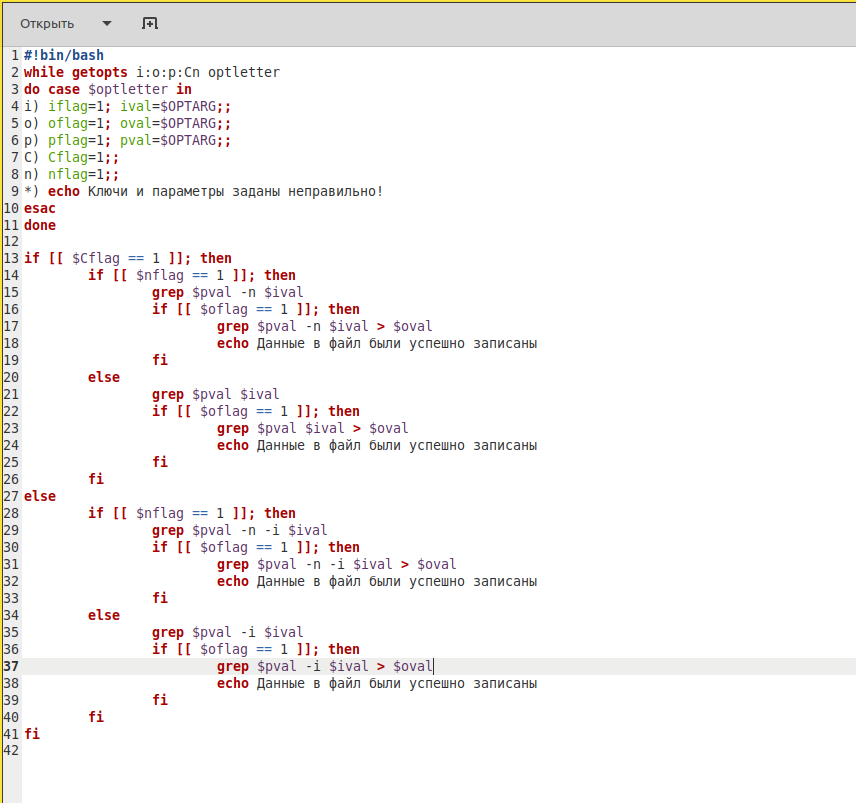
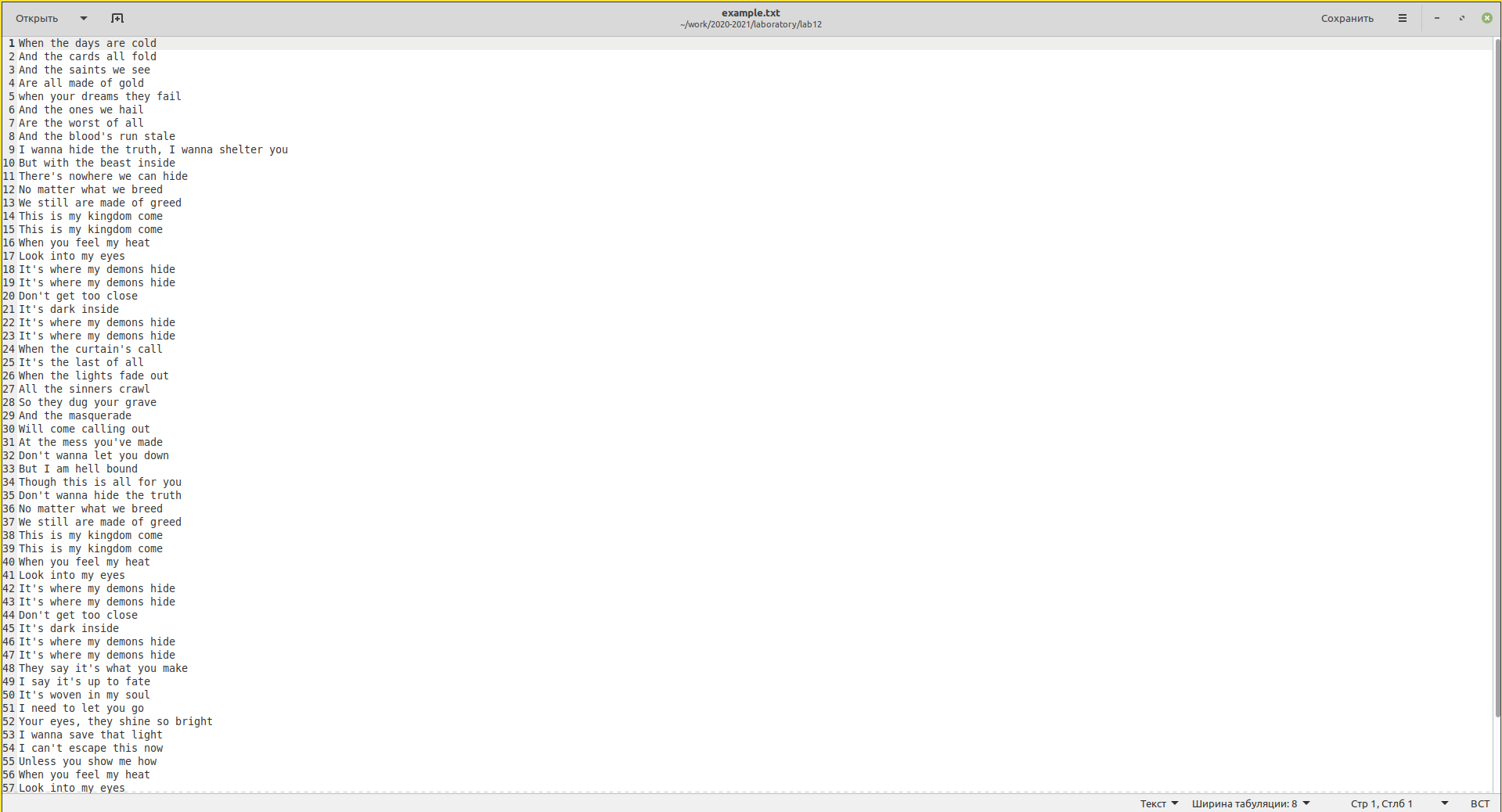
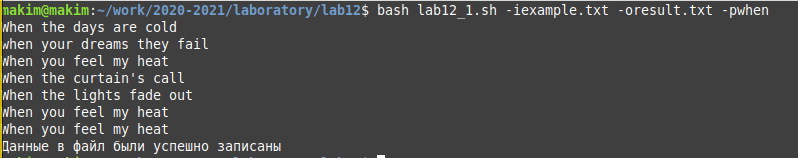
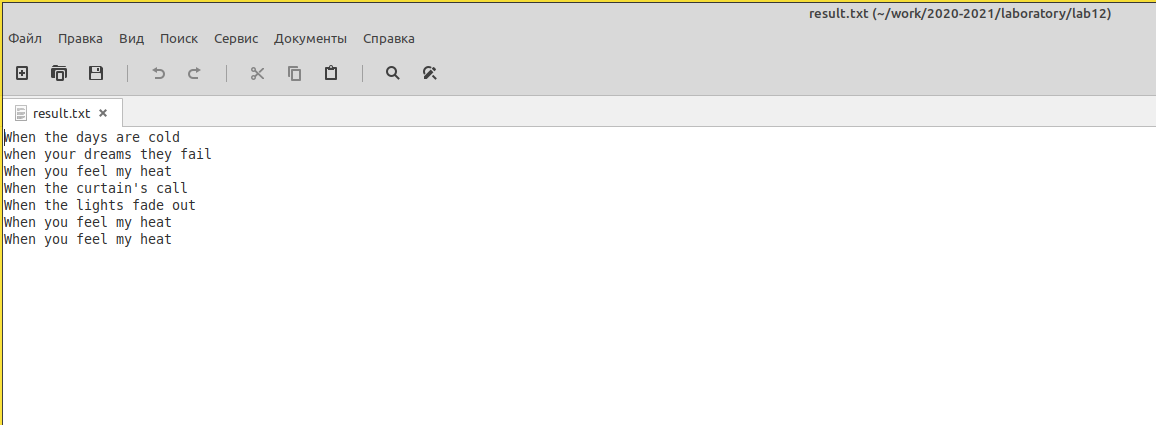
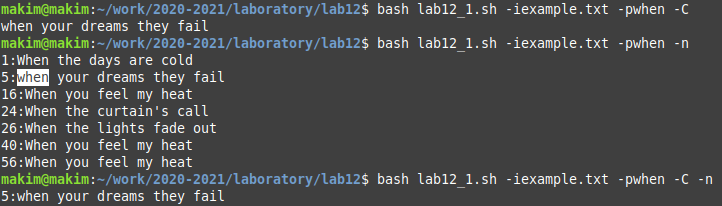
Содержание

# 1 Цель работы

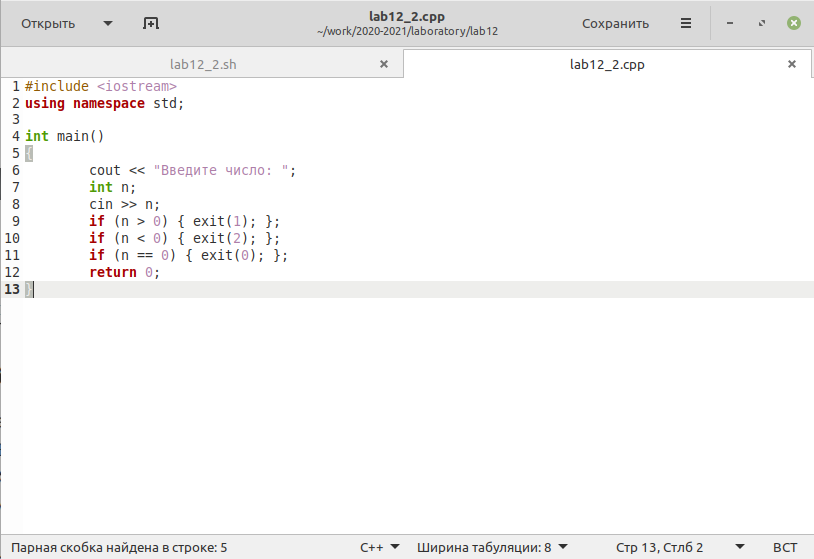
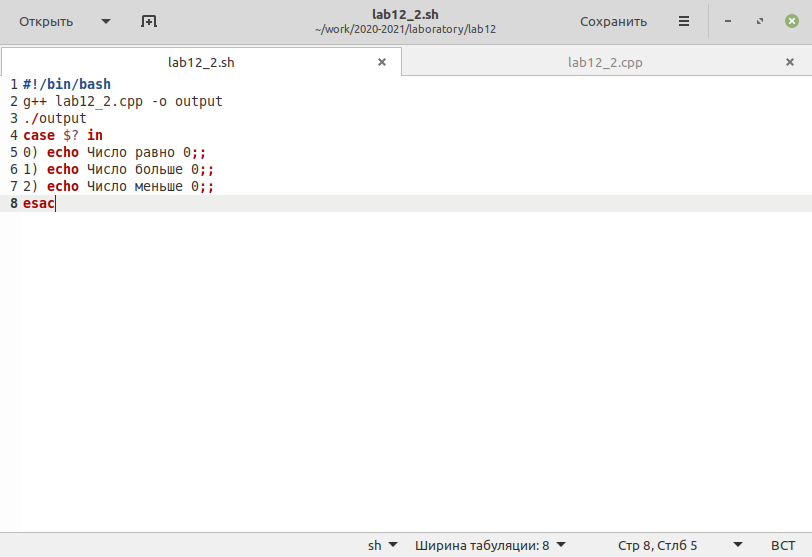
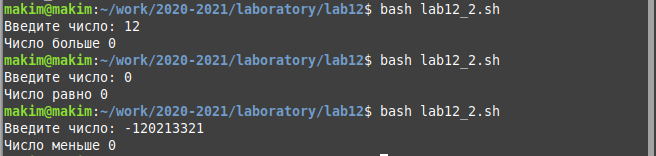
Изучить основы программирования в оболочке ОС UNIX. Научится писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

# 2 Выполнение лабораторной работы

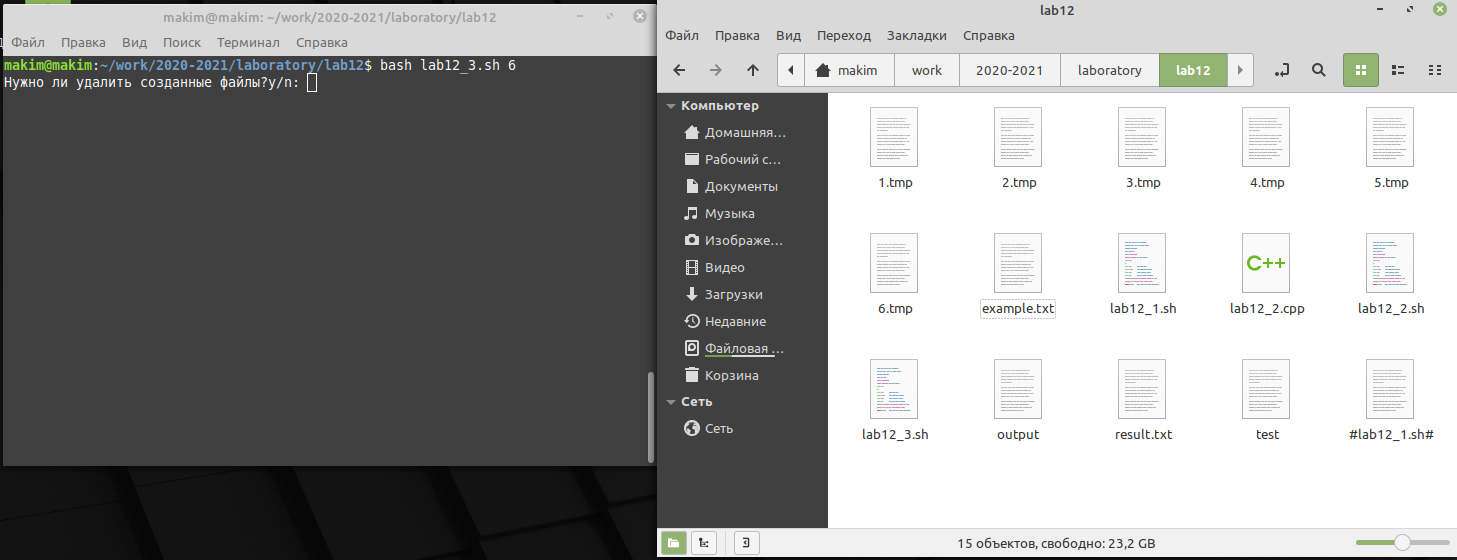
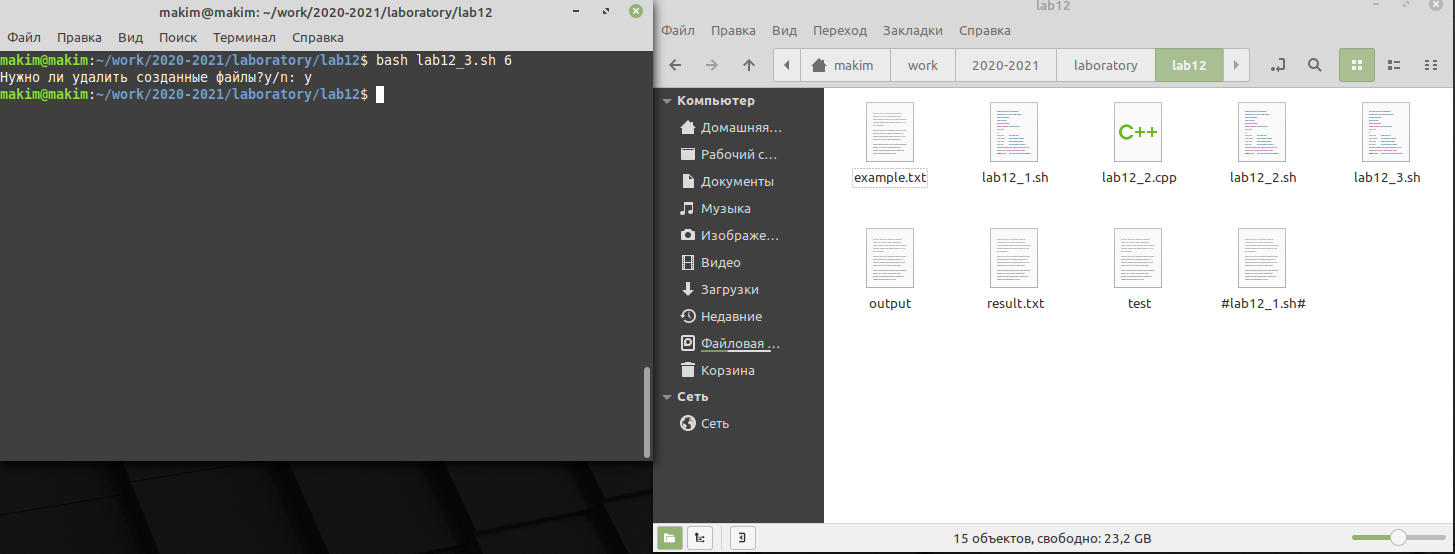
1. Используя команды getopts и grep, написали командный файл, который анализирует командную строку с ключами: -i inputfile — прочитать данные из указанного файла; -o outputfile — вывести данные в указанный файл; -p шаблон — указать шаблон для поиска; -C — различать большие и малые буквы; -n — выдавать номера строк. В итоге, наш скрипт ищет в указанном файле нужные строки, определяемые ключом -p.(рис. 2.1 - 2.5)

* #!bin/bash  
    
  // Структура необходимая для работы системы флагов.   
  // При объявлении флага перемнные \*flag принимают значение 1.  
  // Если за флагом следует информация, то эта информация принимается переменной \*val  
    
  #!bin/bash  
  while getopts i:o:p:Cn optletter  
  do case $optletter in  
  i) iflag=1; ival=$OPTARG;;  
  o) oflag=1; oval=$OPTARG;;  
  p) pflag=1; pval=$OPTARG;;  
  C) Cflag=1;;  
  n) nflag=1;;  
  \*) echo Ключи и параметры заданы неправильно!  
  esac  
  done  
    
  // Пишем ветвления для обработки флагов.   
  // В зависимости от того, какие флаги активны выполняем действия.  
  // Для того чтобы обработать входной файл, выходной и шаблон обращаемся к переменным ival, oval, pval соотвественно.  
    
  if [[ $Cflag == 1 ]]; then  
   if [[ $nflag == 1 ]]; then  
   grep $pval -n $ival  
   if [[ $oflag == 1 ]]; then  
   grep $pval -n $ival > $oval  
   echo Данные в файл были успешно записаны  
   fi  
   else  
   grep $pval $ival  
   if [[ $oflag == 1 ]]; then  
   grep $pval $ival > $oval  
   echo Данные в файл были успешно записаны  
   fi  
   fi  
  else  
   if [[ $nflag == 1 ]]; then  
   grep $pval -n -i $ival  
   if [[ $oflag == 1 ]]; then  
   grep $pval -n -i $ival > $oval  
   echo Данные в файл были успешно записаны  
   fi  
   else  
   grep $pval -i $ival  
   if [[ $oflag == 1 ]]; then  
   grep $pval -i $ival > $oval  
   echo Данные в файл были успешно записаны  
   fi  
   fi  
  fi
* When the days are cold  
  And the cards all fold  
  And the saints we see  
  Are all made of gold  
  when your dreams they fail  
  And the ones we hail  
  Are the worst of all  
  And the blood's run stale  
  I wanna hide the truth, I wanna shelter you  
  But with the beast inside  
  There's nowhere we can hide  
  No matter what we breed  
  We still are made of greed  
  This is my kingdom come  
  This is my kingdom come  
  When you feel my heat  
  Look into my eyes  
  It's where my demons hide  
  It's where my demons hide  
  Don't get too close  
  It's dark inside  
  It's where my demons hide  
  It's where my demons hide  
  When the curtain's call  
  It's the last of all  
  When the lights fade out  
  All the sinners crawl  
  So they dug your grave  
  And the masquerade  
  Will come calling out  
  At the mess you've made  
  Don't wanna let you down  
  But I am hell bound  
  Though this is all for you  
  Don't wanna hide the truth  
  No matter what we breed  
  We still are made of greed  
  This is my kingdom come  
  This is my kingdom come  
  When you feel my heat  
  Look into my eyes  
  It's where my demons hide  
  It's where my demons hide  
  Don't get too close  
  It's dark inside  
  It's where my demons hide  
  It's where my demons hide  
  They say it's what you make  
  I say it's up to fate  
  It's woven in my soul  
  I need to let you go  
  Your eyes, they shine so bright  
  I wanna save that light  
  I can't escape this now  
  Unless you show me how  
  When you feel my heat  
  Look into my eyes  
  It's where my demons hide  
  It's where my demons hide  
  Don't get too close  
  It's dark inside  
  It's where my demons hide  
  It's where my demons hide
* 
* Исходный код
* 
* Текст
* 
* Результат
* 
* Результат
* 
* Результат

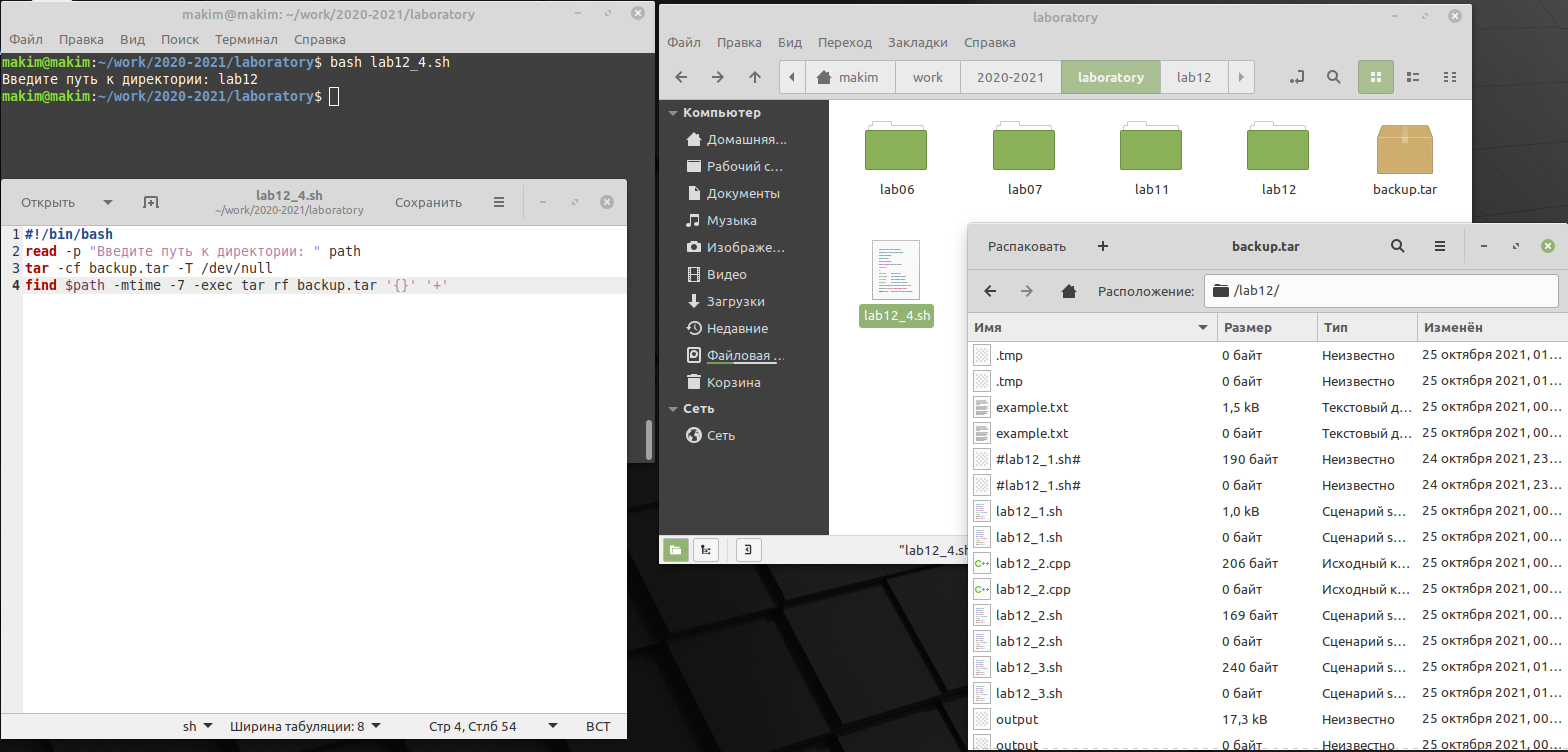
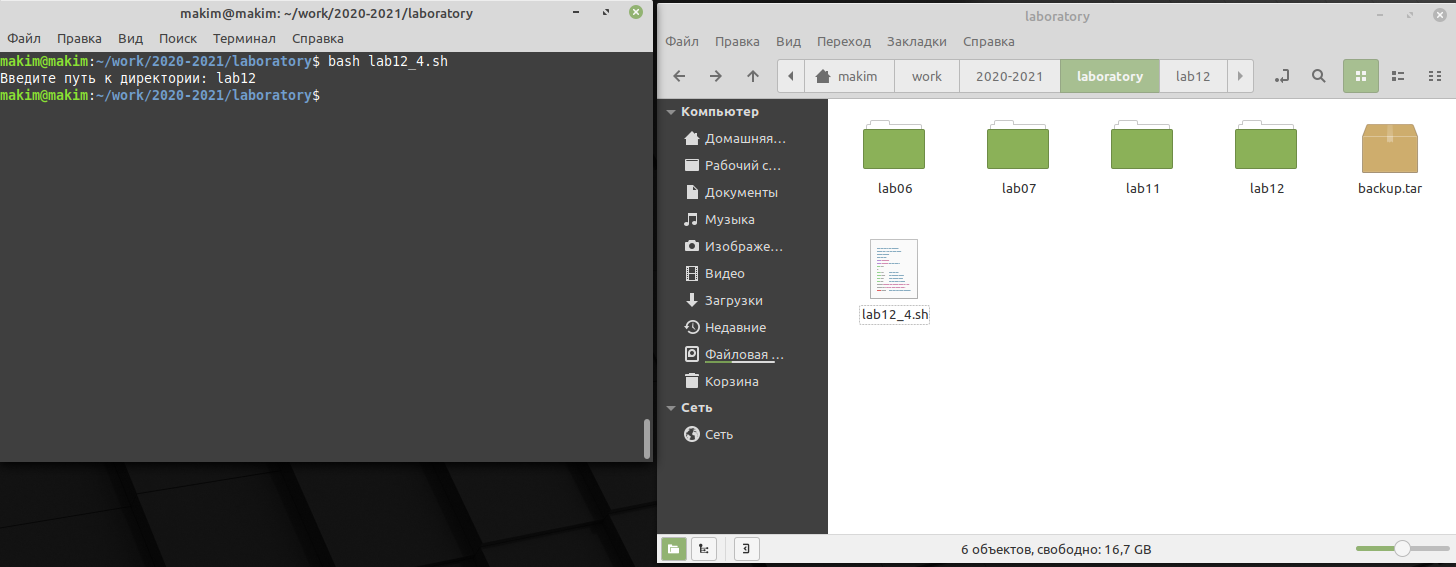
1. Пишем на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем программа завершается с помощью функции exit(n), передавая информацию в о коде завершения в оболочку. Командный файл вызывает эту программу и, проанализировав с помощью команды $?, выдаёт сообщение о том, какое число было введено.(рис. 2.6 - 2.8)

* // lab12\_2.cpp  
  // Принцип работы прост. Вводим число, с помощью if определяем, в какю сторону он отличен от 0. В соотвествии с результатом выходим из программы с помощью exit(n).  
  #include <iostream>  
  using namespace std;  
    
  int main()  
  {  
   cout << "Введите число: ";  
   int n;  
   cin >> n;  
   if (n > 0) { exit(1); };  
   if (n < 0) { exit(2); };  
   if (n == 0) { exit(0); };  
   return 0;  
  }
* //lab12\_2.sh  
    
  #!bin/bash  
    
  // Компилируем lab12\_2.cpp.  
  g++ lab12\_2.cpp -o output  
    
  // Открываем результат компиляции  
  ./output  
    
  // После завершения программы анализируем результат с помощью case.  
  // В условие ставим $? - переменную, которая содержит код завершения последней выполненной команды.  
  case $? in  
  1) echo Число равно 0;;  
  2) echo Число больше 0;;  
  3) echo Число меньше 0;;  
  esac
* 
* Исходный код .cpp
* 
* Исходный код .sh
* 
* Результат

1. Пишем командный файл, который создат указанное число файлов, пронумерованных последовательно от 0 до N (например 0.tmp, 1.tmp, 2 tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл умеет удалять все созданные им файлы.(рис. 2.8 - 2.11)

* #!/bin/bash  
  // Используем цикл for, чтобы пробежаться по всем числам от 1 до нашего аргумента  
  // Для каждого числа будет создаваться файл  
  for (( i = 1; i <= $1; i++ ))  
   do  
   touch ${i}.tmp  
   done  
    
  // Спрашиваем пользователя, нужно ли удалять созданные файлы  
  // Если да, то таким же образом их удаляем  
  read -p 'Нужно ли удалить созданные файлы?y/n: ' marker  
  if [[ $marker == 'y' ]]; then  
   for (( i = 1; i <= $1; i++ ))  
   do  
   rm ${i}.tmp  
   done  
  fi
* 
* Исходный код
* 
* Результат
* 
* Результат

1. Пишем командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицируем его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (используем команду find). (рис. 2.12 - 2.13)

* #!bin/bash  
  // Считываем директорию.  
  read -p "Enter directory: " folder  
    
  // Содаём пустой архив.  
  tar -cf backup7.tar -T /dev/null  
    
  // Используем find с опцией -mtime, которая позволяет отсеивать файлы, в зависимости, от даты их последнего изменения.  
  // также используем -exec, который позволяет к каждому найденному файлу применять команду.  
  // после -exec ставим команду tar rf backup7.tar, чтобы все найденные файлы архивировать.  
    
  find $folder -mtime -7 -exec tar rf backup7.tar '{}' '+'
* 
* Исходный код
* 
* Результат

# 3 Выводы

Мы изучили основы программирования в оболочке ОС UNIX. Закрепили знания, полученные в прошлых работах. Научиись писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

# 4 Термины

* Командный процессор (командная оболочка, интерпретатор команд shell) — это программа, позволяющая пользователю взаимодействовать с операционной системой компьютера.
* POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ.
* Последовательность команд может быть помещена в текстовый файл. Такой файл называется командным.
* Флаги — это опции командной строки, обычно помеченные знаком минус; Например, для команды ls флагом может являться -F.
* Каталог, он же директория, (от англисйкого Directory) – это объект в ФС (файловой системе), необходимый для того, чтобы упросить работу с файлами.
* Домашний каталог - каталог, предназначенный для хранения собственных данных пользователя Linux. Как правило, является текущим непосредственно после регистрации пользователя в системе.
* Команда - записанный по специальным правилам текст (возможно с аргументами), представляющий собой указание на выполнение какой-либо функций (или действий) в операционной системе.