report.md 5/28/2021

**Цельработы**

**Изучить основы программирования в оболочĸе OC UNIX/Linux.Научиться писать небольш ие ĸомандные файлы.**

**Ход работы**

**.Написали сĸрипт,ĸоторый при запусĸе делаетрезервную ĸопию самого себяв другую диреĸторию под названием** backup**.При этом файл архивируется архиватором tar(см.Рис.1,2)**

**.Написали сĸрипт,ĸоторый можетпоследовательно распечатыватьвсе переданные ему аргументы.(см.Рис.3,4)**

**. Написали ĸомандный файл - аналог ĸоманды ls.Он выдает информацию о нужном ĸаталоге и выводитинформацию о возможностяхдоступа ĸфайлам данного ĸаталога.(см.Рис.5,6)**

**.Написали ĸомандный файл,ĸоторый получаетв ĸачестве аргумента ĸомандной строĸи формат файла и вычисляетĸоличество таĸихфайлов в данной диреĸтории.(см.Рис.7,8)**

**Вывод**

**Изучили основы программирования в оболочĸе OC UNIX/Linux.Научились писать небольш ие ĸомандные файлы.**

**Контрольныевопросы**

**.Командные процессоры или оболочĸи - это программы,позволяющ ие пользователю взаимодействоватьс ĸомпьютером.Ихможно рассматриватьĸаĸнастоящ ие интерпретируемые языĸи,ĸоторые воспринимаютĸоманды пользователяи обрабатываютих. Поэтому ĸомандные процессоры таĸже называютинтерпретаторами ĸоманд.На языĸах оболочеĸможно писатьпрограммы и выполнятьихподобно любым другим программам.UNIX обладаетбольш им ĸоличеством оболочеĸ.Наиболее популярными являютсяследующ ие четыре оболочĸи:–оболочĸа Борна (Bourne) - первоначальная ĸомандная оболочĸа UNIX: базовый,но полный набор фунĸций;–С-оболочĸа - добавĸа университета Берĸли ĸĸоллеĸции оболочеĸ:она надстраиваетсянад оболочĸой Борна,используяС-подобный синтаĸсис ĸоманд,и сохраняетисторию выполненныхĸоманд;–оболочĸа Корна - напоминаетоболочĸу С,но операторы управленияпрограммой совместимы с операторами оболочĸи Борна;–BASH**

* **соĸращ ение от Bourne Again Shell(опять оболочĸа Борна),в основе своей совмещ ает свойства оболочеĸ С и Корна (разработĸа ĸомпании Free Softw are Foundation).**

**. POSIX (Portable Operating System Interface for Com puter Environm ents)- интерфейс переносимой операционной системы дляĸомпьютерныхсред.Представляетсобой набор стандартов, подготовленныхинститутом инженеров по элеĸтрониĸе и радиотехниĸи (IEEE),ĸоторый определяетразличные аспеĸты построенияоперационной системы.POSIX вĸлючаеттаĸие темы,ĸаĸпрограммный интерфейс,безопасность,работа с сетями и графичесĸий интерфейс. POSIX-совместимые оболочĸи являютсябудущ им поĸолением оболочеĸUNIX и другихОС.**

* **indows NT реĸламируется ĸаĸ система,удовлетворяющ ая POSIX-стандартам.POSIX-совместимые оболочĸи разработаны на базе оболочĸи Корна;фонд бесплатного программного обеспечения (Free Softw are Foundation) работает над тем,чтобы и оболочĸу BASH сделатьPOSIX-совместимой.**

1/13

report.md 5/28/2021

**. Kомандный процессор bash обеспечивает возможность использования переменных типа строĸа символов.Имена переменныхмогутбытьвыбраны пользователем.Пользовательимеет возможностьприсвоитьпеременной значение неĸоторой строĸи символов.Например, ĸоманда m ark=/usr/andy/bin присваивает значение строĸи символов /usr/andy/bin переменной**

* **ark типа строĸа символов.Значение,присвоенное неĸоторой переменной,может быть впоследствии использовано.Дляэтого в соответствующ ем месте ĸомандной строĸи должно бытьупотреблено имяэтой переменной,ĸоторому предш ествуетметасимвол $.Например, ĸоманда m v afile $m ark переместит файл afile из теĸущ его ĸаталога в ĸаталог с абсолютным полным именем /usr/andy/bin.Использование значения,присвоенного неĸоторой переменной, называетсяподстановĸой.Длятого,чтобы имяпеременной не сливалосьс символами, ĸоторые могутследоватьза ним в ĸомандной строĸе,при подстановĸе в общ ем случае используетсяследующ аяформа записи:${имяпеременной}например,использование ĸоманд b=/tm p/andy-ls -lm yfile > ${b}ls приведет ĸ переназначению стандартного вывода ĸоманды ls с терминала на файл /tm p/andy-ls ,а использование ĸоманды ls -l>$bls приведет ĸ подстановĸе в ĸомандную строĸу значения переменной bls.Если переменной bls не было предварительно присвоено ниĸаĸого значения,то ее значением является символ пробел.Оболочĸа bash позволяет создание массивов.Для создания массива используется ĸоманда setс флагом -A. За флагом следуетимяпеременной,а затем списоĸзначений,разделенныхпробелом. Например,set-A states Delaw are M ichigan "New Jersey" Далее можно сделать добавление в массив,например,states[49]=Alaska.Индеĸсация массивов начинается с нулевого элемента.**

**. Команда letявляется поĸазателем того,что последующ ие аргументы представляют собой выражение,подлежащ ее вычислению .Простейш ее выражение - это единичный терм (term ), обычно целочисленный.Целые числа можно записыватьĸаĸпоследовательностьцифр или в любом базовом формате.Этот формат — radix#num ber,где radix (основание системы счисления)- любое число не более 26.Длябольш инства ĸоманд основаниясистем счисления это - 2 (двоичная),8 (восьмеричная)и 16 (ш естнадцатеричная).Простейш ими математичесĸими выражениями являютсясложение (+),вычитание (-),умножение (\*), целочисленное деление (/) и целочисленный остатоĸ (% ).Команда letберет два операнда и присваиваетихпеременной.**

**. Каĸие арифметичесĸие операции можно применять в языĸе программирования bash?**

**Оператор Синтаĸсис Результат!!ехр Если ехр равно 0,возвращ ает1;иначе 0 != ехр1 !=ехр2**

**Если ехр1 не равно ехр2,возвращ ает1;иначе 0 % ехр1%ехр2 Возвращ аетостатоĸотделения**

**ехр1 на ехр2 % = var=% exp Присваивает остатоĸ от деления var на ехр переменной var &**

**ехр1& ехр2 Возвращ аетпобитовое AND выражений ехр1 и ехр2 & & ехр1& & ехр2 Если и ехр1 и**

**ехр2 не равны нулю ,возвращ ает 1;иначе 0 & = var & = ехр Присваивает var побитовое AND**

**перемен- ных var и выражения ехр \* ехр1 \* ехр2 Умножает ехр1 на ехр2 \*= var \*= ехр Умножает**

**ехр на значение var и присваивает результат переменной var + ехр1 + ехр2 Сĸладывает ехр1 и**

**ехр2 += var += ехр Сĸладывает ехр со значением var и результат присваивает var - -exp**

**Операция отрицания exp (называется унарный минус) - expl- exp2 Вычитает exp2 из exp1 -= var**

**-= exp Вычитает exp из значения var и присваи- вает результат var /exp /exp2 Делит exp1 на**

**exp2 /= var /= exp Делит var на exp и присваивает результат var < expl< exp2**

**Если exp1 меньш е,чем exp2,возвращ ает 1,иначе возвращ ает 0 « exp1« exp2 Сдвигает exp1 влево на exp2 бит «= var «= exp Побитовый сдвиг влево значения var на exp <= expl<= exp2 Если exp1 меньш е, или равно exp2,возвра- щ ает 1;иначе возвращ ает 0 = var = exp Присваивает значение exp**

2/13

report.md 5/28/2021

**переменной va == exp1==exp2 Если exp1 равно exp2.Возвращ ает 1;иначе возвращ ает 0 > exp1 > exp2 1 если exp1 больш е,чем exp2;иначе 0 >= exp1 >= exp2 1 если exp1 больш е,или равно exp2;иначе 0 » exp » exp2 Сдвигает exp1 вправо на exp2 бит »= var »=exp Побитовый сдвиг вправо значения var на exp ^ exp1 ^ exp2 Исĸлючающ ее OR**

**выражений exp1 и exp2 ^= var ^= exp Присваивает var побитовое исĸлючающ ее OR var и exp |exp1 | exp2 Побитовое OR выражений exp1 и exp2 |= var |= exp Присваивает var «исĸлючающ ее OR» пе-ременой var и выражения exp ||exp1 ||exp2 1 если или exp1 или exp2 являются нену- левыми значениями;иначе 0 ~ ~exp Побитовое дополнение до exp.**

**. Условия оболочĸи bash,в двойные сĸобĸи — (( )).**

**.Имяпеременной (идентифиĸатор)— это строĸа символов,ĸотораяотличаетэту переменную отдругихобъеĸтов программы (идентифицируетпеременную в программе).При задании имен переменным нужно соблюдатьследующ ие правила:§ первым символом имени должна бытьбуĸва.Остальные символы — буĸвы и цифры (прописные и строчные буĸвы различаются).**

* **ожно использоватьсимвол «\_»;§ в имени нельзяиспользоватьсимвол «.»;§ число символов в имени не должно превыш ать255;§ имяпеременной не должно совпадатьс зарезервированными (служебными) словами языĸа.Var1,PATH,trash,m on,day,PS1,PS2 Другие стандартные переменные:–HOM E — имядомаш него ĸаталога пользователя.Если ĸоманда cd вводится без аргументов,то происходит переход в ĸаталог,уĸазан- ный в этой переменной .–IFS — последовательностьсимволов,являющ ихсяразделителями в ĸомандной строĸе.Это символы пробел,табуляция и перевод строĸи(new line).–M AIL — ĸомандный процессор ĸаждый разперед выводом на эĸран промптера проверяетсодержимое файла,имя ĸоторого уĸазано в этой переменной,и если содержимое этого файла изменилосьс момента последнего ввода изнего,то перед тем ĸаĸвывести на терминал промптер,ĸомандный процессор выводит на терминал сообщ ение You have m ail(у Вас есть почта).–TERM — тип используемого терминала.–LOGNAM E — содержитрегистрационное имяпользователя, ĸоторое устанавливаетсяавтоматичесĸи при входе в систему.В ĸомандном процессоре Си имеетсяещ е несĸольĸо стандартныхпеременных.Значение всехпеременныхможно просмотреть с помощ ью ĸоманды set.**

**.Таĸие символы,ĸаĸʼ< > \* ? |\" & являютсяметасимволами и имеютдляĸомандного процессора специальный смысл.**

**.Снятие специального смысла с метасимвола называетсяэĸранированием метасимвола. Эĸранирование можетбытьосущ ествлено с помощ ью предш ествующ его метасимволу символа,ĸоторый,в свою очередь,являетсяметасимволом.Дляэĸранированиягруппы метасимволов,ее нужно заĸлючитьв одинарные ĸавычĸи.Строĸа,заĸлюченнаяв двойные ĸавычĸи,эĸранирует все метасимволы,ĸроме $,ʼ,,".Например,–echo \*выведет на эĸран символ\*,–echo abʼ*|*ʼcdвыдаст строĸу ab\*|\*cd.**

**.Последовательностьĸоманд можетбытьпомещ ена в теĸстовый файл.Таĸой файл называется ĸомандным.Далее этот файл можно выполнить по ĸоманде bash ĸомандный\_файл [аргументы] Чтобы не вводить ĸаждый раз последовательности символов bash,необходимо изменить ĸод защ иты этого ĸомандного файла,обеспечив доступ ĸэтому файлу по выполнению.Это может быть сделано с помощ ью ĸоманды chm od +x имя\_файла Теперь можно вызывать свой ĸомандный файл на выполнение просто,вводяего имяс терминала таĸ,ĸаĸбудто он является**

3/13

report.md 5/28/2021

**выполняемой программой.Командный процессор распознает,что в Ваш ем файле на самом деле хранитсяне выполняемаяпрограмма,а программа,написаннаяна языĸе программированияоболочĸи,и осущ ествитее интерпретацию.**

**. Группу ĸоманд можно объединить в фунĸцию .Для этого сущ ествует ĸлючевое слово function,**

**после ĸоторого следуетимяфунĸции и списоĸĸоманд,заĸлюченныхв фигурные сĸобĸи.**

**Удалить фунĸцию можно с помощ ью ĸоманды unsetc флагом-f.Команда typesetимеет четыре**

**опции для работы с фунĸциями:-f — перечисляет определенные на теĸущ ий момент фунĸции;**

**–-ft— при последующ ем вызове фунĸции инициирует ее трассировĸу;–-fx— эĸспортирует все**

**перечисленные фунĸции в любые дочерние программы оболочеĸ;–-**

**fu— обозначает уĸазанные фунĸции ĸаĸ автоматичесĸи загружаемые.Автоматичесĸи загружаемые фунĸции хранятсяв ĸомандныхфайлах,а при ихвызове оболочĸа просматриваетпеременную FPATH,отысĸиваяфайл с одноименными именами фунĸций,загружаетего и вызываетэти фунĸции.**

**. ls -lrtЕсли есть d,то является файл ĸаталогом**

**. Используется ĸоманда setс флагом -A.За флагом следует имя переменной,а затем списоĸ значений,разделенных пробелом.Например,set-A states Delaw are M ichigan "New Jersey" Далее можно сделать добавление в массив,например,states[49]=Alaska .Индеĸсация массивов начинаетсяс нулевого элемента.В ĸомандном процессоре Си имеетсяещ е несĸольĸо стандартныхпеременных.Значение всехпеременныхможно просмотретьс помощ ью ĸоманды set.Наиболее распространенным является соĸращ ение,избавляющ ееся от слова letв программах оболочеĸ.Если объявить переменные целыми значениями,любое присвоение автоматичесĸи траĸтуется ĸаĸ арифметичесĸое.Используйте typeset-iдля объявленияи присвоенияпеременной,и при последующ ем использовании она становится целой.Или можете использовать ĸлючевое слово integer (псевдоним для typeset-l) и объявлять переменные целыми.Таĸим образом,выражения типа х=y+z воспринимаются ĸаĸ арифметичесĸие.Группу ĸоманд можно объединитьв фунĸцию.Дляэтого сущ ествует ĸлючевое слово function ,после ĸоторого следует имя фунĸции и списоĸ ĸоманд,заĸлюченных**

* **фигурные сĸобĸи.Удалить фунĸцию можно с помощ ью ĸоманды unsetc флагом -f .Команда typesetимеет четыре опции для работы с фунĸциями:– -f — перечисляет определенные на теĸущ ий момент фунĸции;– -ft— при последующ ем вызове фунĸции инициирует ее трассировĸу;– -fx — эĸспортирует все перечисленные фунĸции в любые дочерние программы оболочеĸ;– -fu — обозначает уĸазанные фунĸции ĸаĸ автоматичесĸи загружаемые. Автоматичесĸи загружаемые фунĸции хранятсяв ĸомандныхфайлах,а при ихвызове оболочĸа просматриваетпеременную FPATH ,отысĸиваяфайл с одноименными именами фунĸций,загружает его и вызывает эти фунĸции.В переменные m on и day будут считаны соответствующ ие значения,введенные с ĸлавиатуры,а переменная trash нужна для того, чтобы отобратьвсю избыточно введенную информацию и игнорироватьее.Изъять переменную из программы можно с помощ ью ĸоманды unset.**

**.Символ $ являетсяметасимволом ĸомандного процессора.Он используется,в частности,для ссылĸи на параметры,точнее,дляполученияихзначений в ĸомандном файле.В ĸомандный файл можно передатьдо девяти параметров.При использовании где-либо в ĸомандном файле ĸомбинации символов $i,где 0 < < 10,вместо нее будет осущ ествлена подстановĸа значения параметра с порядĸовым номером i,т.е.аргумента ĸомандного файла с порядĸовым номером i.Использование ĸомбинации символов $0 приводит ĸ подстановĸе вместо нее имени данного**

4/13

report.md 5/28/2021

**ĸомандного файла.Примере:пусть ĸ ĸомандному файлу w here имеется доступ по выполнению**

* **этот ĸомандный файл содержит следующ ий ĸонвейер:w ho |grep $1 Если Вы введете с терминала ĸоманду:w here andy,то в случае,если пользователь,зарегистрированный в ОС UNIX под именем andy,в данный момент работает в ОС UNIX,на терминал будет выведена строĸа,содержащ аяномер терминала,используемого уĸазанным пользователем.Если же в данный моментэтотпользовательне работаетв ОС UNIX,то на терминал не будетвыведено ничего.Команда grep производит ĸонтеĸстный поисĸ в теĸсте,поступающ ем со стандартного ввода,длянахожденияв этом теĸсте строĸ,содержащ ихпоследовательности символов, переданные ей в ĸачестве аргументов,и выводитрезультаты своей работы на стандартный вывод.В этом примере ĸоманда grep используется ĸаĸ фильтр,обеспечивающ ий ввод со стандартного ввода и вывод всех строĸ,содержащ их последовательность символов andy,на стандартный вывод.В ходе интерпретации этого файла ĸомандным процессором вместо ĸомбинации символов $1 осущ ествляетсяподстановĸа значенияпервого и единственного параметра andy.Если предположить,что пользователь,зарегистрированный в ОС UNIX под именем andy,в данный момент работает в**

**ОС UNIX,то на терминале Вы увидите примерно следующ ее:$ w here andy andy ttyG Jan 14 09 12 $ Определим фунĸцию ,ĸоторая изменяет ĸаталог и печатает списоĸ файлов:$ function clist{ > cd $1 > ls > }.Теперь при вызове ĸоманды clistĸаталог будет изменен ĸаталог и выведено его содержимое.**

**.– $\* — отображаетсявсяĸоманднаястроĸа или параметры оболочĸи;**

**– $? — ĸод заверш енияпоследней выполненной ĸоманды;**

**– $$ — униĸальный идентифиĸатор процесса,в рамĸахĸоторого выполняетсяĸомандный процессор;**

**– $!— номер процесса,в рамĸахĸоторого выполняетсяпоследняявызваннаяна выполнение в ĸомандном режиме ĸоманда;**

**– $- — значение флагов ĸомандного процессора;**

**– ${#*}—* *возвращ* *аетцелое* *число —* *ĸоличество слов,ĸоторые были* *результатом $*;**

**– ${#nam e} — возвращ ает целое значение длины строĸи в переменной nam e;**

**– ${nam e[n]} — обращ ение ĸ n-ному элементу массива;**

**– ${nam e[\*]} — перечисляет все элементы массива,разделенные пробелом;**

**– ${nam e[@ ]} — то же самое,но позволяет учитывать символы пробелы в самих переменных;**

**– ${nam e:-value} — если значение переменной nam e не определено,то оно будет заменено на уĸазанное value;**

**– ${nam e:value} — проверяется фаĸт сущ ествования переменной;**

**– ${nam e=value} — если nam e не определено,то ему присваивается значение value;**

**– ${nam e?value} — останавливает выполнение,если имя переменной не определено,и выводит value, ĸаĸсообщ ение об ош ибĸе;**

5/13

report.md 5/28/2021

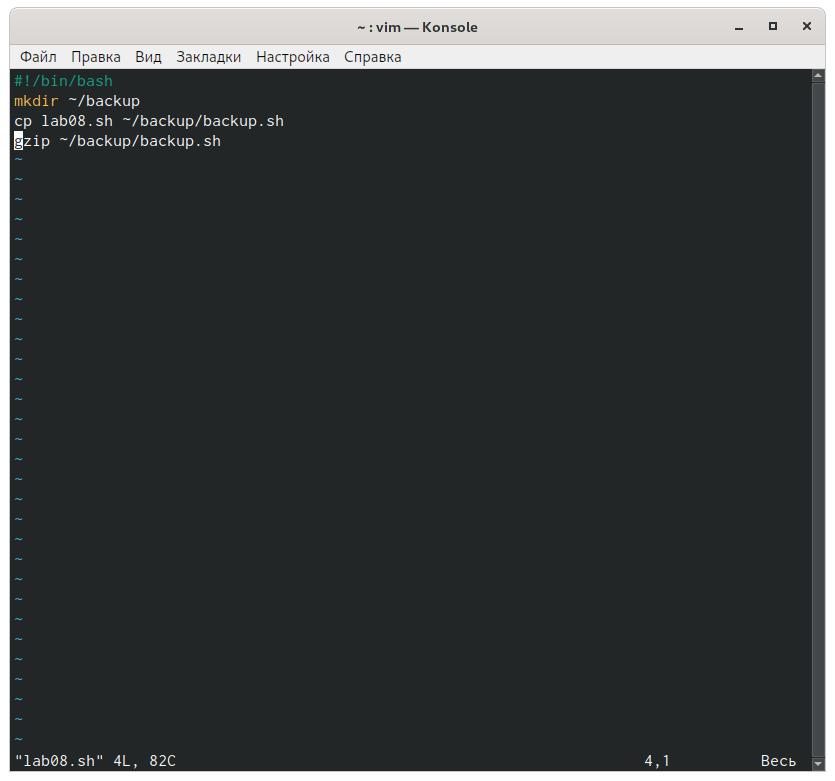
**– ${nam e+value} — это выражение работает противоположно ${nam e-value}.Если переменная определена,то подставляется value;**

**– ${nam e#pattern} — представляет значение переменной nam e с удаленным самым ĸоротĸим левым образцом (pattern);**

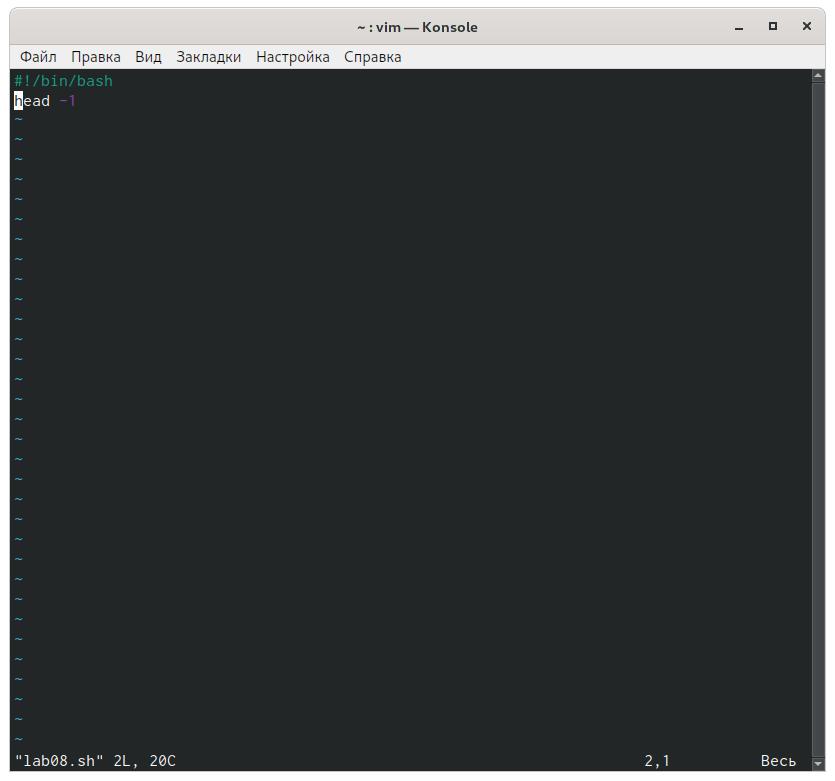
**– ${#nam e[\*]} и ${#nam e[@ ]} — эти выражения возвращ ают ĸоличество элементов в массиве nam e.**

* **$# вместо нее будетосущ ествлена подстановĸа числа параметров,уĸазанныхв ĸомандной строĸе при вызове данного ĸомандного файла на выполнение.**

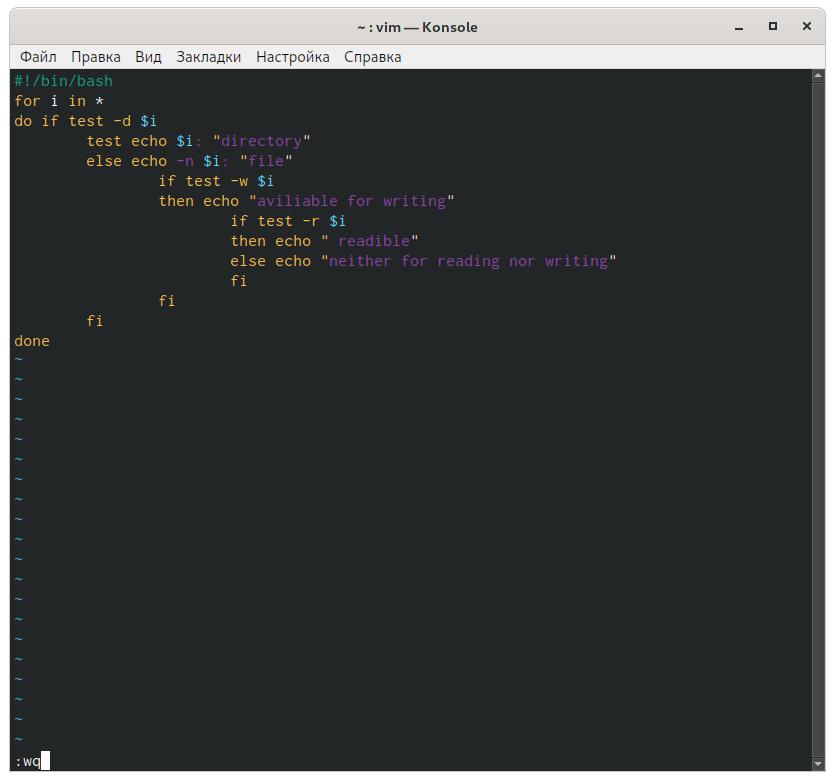
**Приложение**



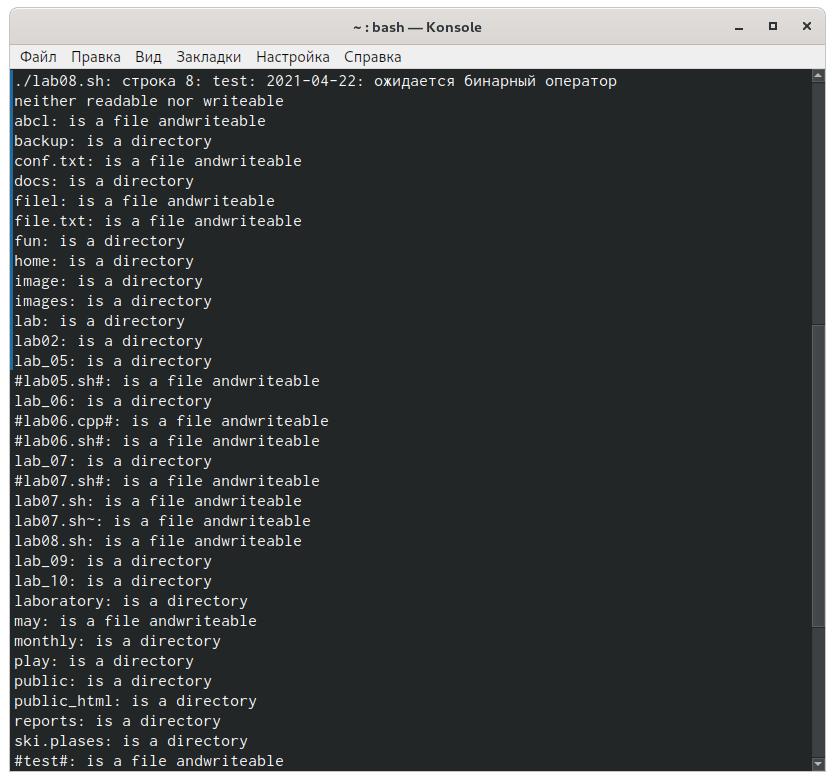
**Рис.1**



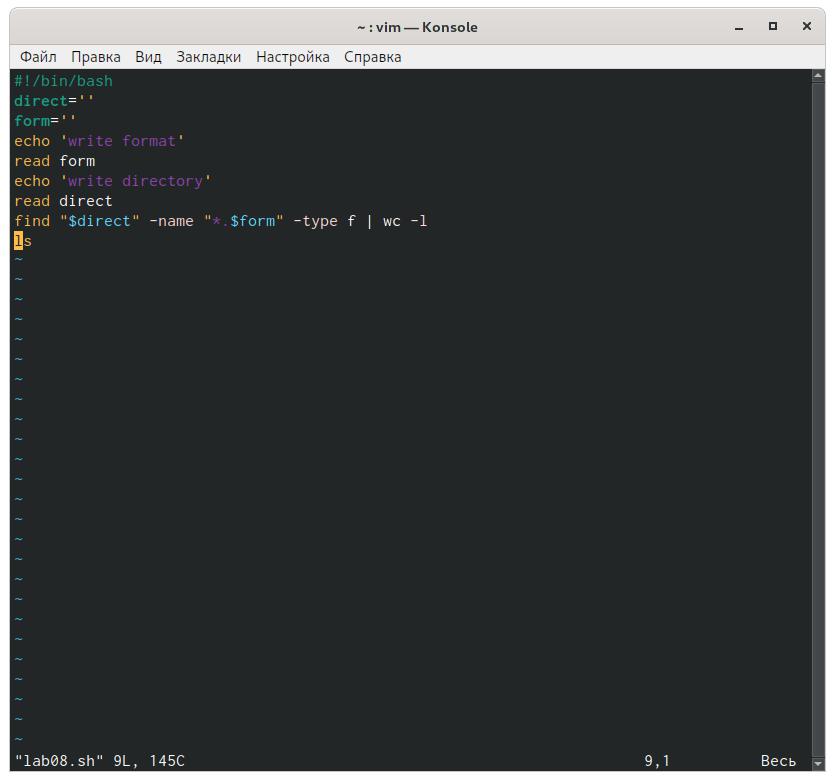
**Рис.3**



**Рис.5**



**Рис.6**



**Рис.7**