**Papyrus shortcut:**

Alt+F => File option

Press 1 or 2 or desired number =>Select from Recent projects

Keep cursor on a particular line. Ctrl+D => copied & pasted that line

Ctrl+s => Find in code

head -148 cscums1\_test\_data|tail -73 >vb >>majher kono 1ta customer data select. Suppose- 3rd customer.

CTRL+C >>ends command line

ls \*.html >>list all files that end with .html extension

exit >>

**Cygwin cheatsheet**: <http://www.voxforge.org/home/docs/cygwin-cheat-sheet>

|  |  |  |
| --- | --- | --- |
| Pwd | print working directory. Current directory path. | |
| Crtl+Q | Toggle comment/uncomment  # -comment | |
| CTRL+k | comment | |
| Ctrl+D | Command end/break | |
| cp filename newfilename | Copy | |
| mv filename newfilename | Move | |
| rm filename | Remove/delete a file | |
| mkdir kohl | Makes new folder on the current directory | |
| Cd .. | Moves up one directory | |
| ls -l|head -3 |  | |
| cd print\ process  cd 'print process' | When Directory name with space | |
| cd c:/ |  | |
| Clear  ctrl+l | clear window | |
| Touch filename | create new, empty files | |
| cat filename | show what is in the file | |
| cat filename > another newfilename | redirect the data of old file to new file | |
| cat filename >> another newfilename | **append** data or text to newfile. | |
| ls | list all folders or files in a directory | |
| ls> new filename | list all files in a directory and shows on a newfile | |
| ls –l | list all files and info | |
| ls –al | list all files and info | |
| head –10 filename | shows 1st 10 lines of the file | |
| tail -10 filename>>newfilename | shows last 10 lines of file and shows in newfile | |
| wc filename | EXAMPLE:  wc cscums1\_test\_data  285 855 13047 cscums1\_test\_data  285 lines, 855 words,13047 cahracters | |
| Wc –l filename | Just show line no | |
| ls -l|wc –l | Show total line numbers of the output of  (ls-l)  ls –l erjnno 5ta o/p pele, ei command shows 5 | |
| cat filename| wc -l  285 | Show total line numbers of the output of  filename | |
| ls -l|head -3 | Shows 1st 3 result of listing of all files of that directory | |
|  |  | |
|  |  | |
| grep "^200" data\_filename | Shows all data starting with 200 | |
| grep -n "^200" data\_filename | Shows each line no | |
| grep -n "^200" data\_filename1 data\_filename 2> output\_filename | Search in multiple files | |
| grep -o “pattern” filename | outputs only the matched pattern. Not the whole line | |
| grep -o "name=[^ ]\*" | Giving ^ inside [ ] means “NOT”  Name=“Not space” pattern as many times | |
| ^.{24} | When you want to go to a large number position.  Keep cursor on a particular line. Press CTRL+F  On notepad++, it goes to column 25 | |
| cut -d "~" -f 1 cscums1\_test\_data > k | Cut with delimiter field number 1 | |
| cut -d "d" -f 1,3 p1.txt > mini | When we want to select more than 1 field output shows with delimiter between the fields | |
| p1.txt | jshdjk  jhdjkhdkj  dad god blue blue  world blue blue |
| mini | jsh  jhdkj  d go  worl |
| cut -d "~" -f 3 filename | head -10 | 1st 10 results of:  Cut with ~ and output field no 3 | |
| grep "^200" cscums1\_test\_data | wc -l | How many customers? | |
| grep "^.\*s" p1.txt | “\*” Means any number of times  “.” Means any character | |
| grep "^j.\*jkh.\*s" p1.txt | Looks for anything starts with “j” . Then anything any number of times.Then pattern “jkh”. Then anything any number of times. Then “s” | |
| grep -n "^200" cscums1\_test\_data | 3:200~013~020000000002~0510012016~0610312016~07FLB IQ CREDIT UNION~08FISHER'S LANDING BRANCH~09PO BOX 1739~10VANCOUVER WA 98668~12VANCOUVER~13WA~1498668~164~181~192~251`  76:200~013~020000000003~0510012016~0610312016~07SCB - IQ CREDIT UNION~08SALMON CREEK BRANCH~09PO BOX 1739~10VANCOUVER WA 98668-1739~12VANCOUVER~13WA~1498668-1739~164~181~193~251`  149:200~013~020000000004~0510012016~0610312016~07VMB IQ CREDIT UNION~08305 NE 81ST ST~09VANCOUVER WA 98665~12VANCOUVER~13WA~1498665~15,C09~164~181~194~251`  245:200~013~020000000005~0510012016~0610312016~07BGB IQ CREDIT UNION~08BATTLE GROUND BRANCH~09PO BOX 1739~10VANCOUVER WA 98668-1739~12VANCOUVER~13WA~1498668-1739~164~181~251`  272:200~014~020000000008~0510012016~0610312016~07PAYMENT SYSTEMS DEPART IQ CU~08IQ CU~09PO BOX 1739~10VANCOUVER WA 98668-1739~12VANCOUVER~13WA~1498668-1739~164~181~198~24none@iqcu.com~251` | |
| grep -n "^200" cscums1\_test\_data | cut -d ":" -f 1 | 3  76  149  245  272 | |
| grep -n "^200" cscums1\_test\_data | cut -d ":" -f 1| head -3 | 3  76  149 | |
| grep -n "^200" cscums1\_test\_data | cut -d ":" -f 1| head -3| tail -1 | 149 | |
| grep -n "^200.\*${cus}" uccumv2.sym.con | cut -d ":" -f 1 |  | |
| head -148 filename | 1st 148 lines | |
| head -148 filename|tail -73  148-76=72 +1 = 73 | Majher kono customer data select | |
| cut -c 3 filename | Only prints column 3 of the file  h  h  g | |
| cat > filename | Press ENTER  Then input some lines on console. Press CTRL+C. Those lines will be written into the “filename”.  hd  hd  g | |
| cut -c 3,5 write | Outputs col 3 and col 5 of every line | |
| cut -c 3-5 write | Outputs from col 3 to col 5 of every line | |
| cut -c-8 test.txt | Outputs from first to col 8 of every line | |
| cut -c 1,3-5 write | Outputs col 1 and from col 3 to col 5 of every line | |
| cut -d "g" -f 3-5 write | Divides fields with ‘g’ and outputs field from 3 to 5 | |
| echo $date | cut -c 5-8 | When you want to use CUT command on Variable then u need to echo that variable.  Bcz Cut is only used on filename | |
| sh shell1.txt | Run a shell file from cmd | |
| a=10  echo “$a”  b=$(($a\*5))  echo $b | For arithmetic operation use double parenthesis after $  Like this: $(( 4\*5 )) | |
| b=$(($(($c+1))/3)) | c=20, b=c+1=21, b/3=7 | |
| l=`ls -l`  echo "$l" | To store a command in a variable use ` (left top corner of keyboard) | |
| line="b jkjk nkoo"  for word in $line  do  echo $word  done  IFS=, | b  jkjk  nkoo  on for command it usually separates with space, using this separates with comma(,) | |
| for each in $e  > do  > a="af"$each"afrida"  > echo "$a"  > done | afloveafrida  afminiafrida  afoutafrida | |
| for i in {1..5}  do  echo "welcome $i times"  done | welcome 1 times  welcome 2 times  welcome 3 times  welcome 4 times  welcome 5 times | |
| for i in {1..10..2}  do  echo "welcome $i times"  done | welcome 1 times  welcome 3 times  welcome 5 times  welcome 7 times  welcome 9 times | |
| for i in 1 2 3 4  do  echo “welcome $i times"  done | welcome 1 times  welcome 2 times  welcome 3 times  welcome 4 times | |
| for i in 1,2,3  do  echo "hi5 $i times"  done | hi5 1,2,3 times  Here, 1,2,3 is considered as one string | |
| for i in {1,2,3}  do  echo "hi5 $i times"  done | hi5 1 times  hi5 2 times  hi5 3 times  When 2nd bracket given 1,2,3 is considered as individual value. It works as when space is given instead of comma | |
| while read line  do  echo $line  done < shell1.sh | Everything it reads, assign that within ‘line’. So when we echo ‘line’ we get a output.  Inputs are read from Shell.sh file | |
| read -p "please enter one number:" var1 | give some info to user what he has to give.  please enter one number:2  2 | |
| read name  echo $name | Takes i/p  Shows o/p | |
| # s=234  # echo $s"j"  # echo "${s}j" | 234j  234j | |
| d="c:/ISIS/docdef/uccumv1.dfa"  echo ${d#\*/} | #- prefix  One # means upto only one pattern that matches. Two # means as many as.  ISIS/docdef/uccumv1.dfa | |
| echo ${d##\*/} | uccumv1.dfa | |
| echo ${d%/\*} | %-suffix  c:/ISIS/docdef  pattern should be matched from the ending | |
| echo ${d%%/\*} | C:/ | |
| echo ${d^}  echo ${d^^} | ^ -capital letter  ^^ -All capital letter | |
| echo ${d,,} # , -small letter  echo ${#d} # string length  echo ${d:2:5} | index 2 upto length 5 | |
| echo "${d:0:5}${d:8:3}" | From index 0 upto length 5, from index 8 upto length 3. Cut rest of the parts in between | |
| echo "${d:0:5} afrida ${d:8:3}" | Includes “afrida” string in between | |
| wc -w filename | 6 filename.  Outputs total words. There were 6 words in the file. word counted when “space” found | |
| ls | wc -l | 5 5 112  Line word character | |
| echo $var | wc -w |  | |
| Wc –l | Asks for input. Then counts the word  Example-  L  Kl  Off  3 | |
| See notepad shell1.sh |  | |
| While read line  do  echo $line  done < filename | Read each line from file and print those lines | |
| n=${d#\*/}  echo $n | d="c:/path/klolk"  output:  path/klolk | |
| echo $n | cut -d "/" -f 2 | path | |
| Write username | Chatwindow appear. | |
| line=`grep -n "^200" uccumv2.sym.con | grep -n "$cus" | cut -d ":" -f 1`  b=$(($line+1)) | Jotogulo 200 | 18:blabla | 18  18+1=19 | |
| read cus  grep -n "^200.\*${cus}" uccumv2.sym.con | cut -d ":" -f 1 | # take the account number input  ## find in the line number of a particular customer/account number? | |
| line=`grep -n "^200" uccumv2.sym.con | grep -n "$cus" | cut -d ":" -f 1`  b=$(($line+1))  real=`grep -n "^200" uccumv2.sym.con | head -$b | tail -1 | cut -d ":" -f 1`  final=$(($real-1))  echo $final | ## find the line number where a particular customer ends?  1st theke 19th line porjnto select kore 19th line ta nibo. Okhan theke cut kore just main file er line number ta rakhbo. Suppose-> 234  234-1=233 | |
| Function func\_name  {  } |  | |
| function add  {  echo "hello"  echo $1 $2  val1=$1  val2=$2  sum=$(($val1+$val2))  sum=$(($1+$2))  echo $sum  }  add 5 2 |  | |
| function get\_customer\_info  {  customer\_no=`grep -n "^200" $3 | grep -n "$1" | cut -d ":" -f 1`  next\_cus=$(($customer\_no+1))  real\_lineno=`grep -n "^200" $3 | head -$next\_cus | tail -1 | cut -d ":" -f 1`  end\_line=$(($real\_lineno-1))  echo $end\_line  startline=`grep -n "^200.\*${1}" $3 | cut -d ":" -f 1`  print\_allinfo=$(($end\_line-$startline+1))  head -$end\_line $3 | tail -$print\_allinfo >> $2  } |  | |
| IFS=,  rm output  var="0000080202,0000080228,0000080230"    for each in $var  do  cd c:/ISIS/data  echo "enter customer account no:"  echo $each  get\_customer\_info $each "output" "uccumv2.sym.con"  done | IFS prevents whitespace to be a delimiter in for loop  Rm- removes outputfile if anything written on the file previously  Cus accnt no saved on a string named “var”. So u don’t need to input acc no each time  3 params- Acc no, output filename,input filename | |
| If[$a –eq $b ]  then  echo “same”  else  fi | a=100  b=25 | |
| var="afrida,a"  var2="afrida,a"  if [ “$var” = “$var2” ]  then  echo "equal"  else  echo "not equal"  fi | -eq for numeric comparision  = for string comparision | |
| if [ -s $opfile ]  then  rm $opfile  fi | s => check if file exists | |
| sed 's/me/red/' p1.txt |  | |
| sed –i 's/me/red/' p1.txt | Change in file | |
| sed 's/me/red/g' p1.txt | Global change | |
| var=`echo $var | sed 's/blue/red red go/'` | hoina. What does it mean? | |
| echo "${test}" | sed 's/is/jij/' | Replace “is”  Test ‘check this out’  check thjij out | |
| echo "${test/his/watt}" | Replace “his”  check twatt out | |
| echo "${test//c/watt}" | Replace all c’s  watthewattk this out | |
| echo "${test/#\*h/x}" | #is prefix. Replace from the beginning upto “h” with “x” | |
| echo "${test/%h/x}" | % suffix. Replace from the end upto “o”with “x” | |
| echo "${abc:-Not here}" | output:Not here  if abc is not defined otherwise print the value of abc | |
| echo "${abc:+Not here}" | output: (shows empty)  if abc not defined otherwise print the value of abc | |
| sed 's/blue/bel/' love.txt | Replace the first occurrence of “blue” with “bel” from all lines | |
| sed 's/blue/bel/2g' love.txt | Replace all “blue”s from 2nd occurrence from all lines | |
| sed `s/shell/& bel/gi` love.txt | & will have the value of “shell” and if shell found then it's value will be replaced with “shell bel”  *Output:*  jshdjk  jhjkhkj  blue success go blue success blue success  world blue success blue success blue success  mini blue success magic blue success blue success blue success  win blue success rot blue success | |
| sed '3,5 s/blue/far/' love. | Applies for 3rd to 5th line | |
| sed '2,$ s/blue/gold/' love.txt | Applies from 2nd to last line | |
| sed '/a/b/ s/blue/gold/' love.txt | Applies on 1st “a” to 1st “b” | |
| \b  \bshell\b  \bshell\B | \b : Word er boundary  \B : anything after word  Search for “shell” . not cshell or shelled | |
| sed '2,4 d' love | Deletes from 2nd to 4th lines | |
| sed '/shell/ d' love | Deletes the line containing this pattern | |
| sed -e '2,4 d' love -e '2,4 d' | For writing multiple commands use “-e” | |
| $ passwd  then the prompt will ask for current password and new password | changing password | |
| sed '4,$! d' love | deletes all lines except from 4th line to last line | |
| Touch forrest  mv forrest green | Creates file named “forrest”  And makes another file named “green” which contains the content of | |
| grep -l “^200~” \* | will output JUST the names of files which matched, without showing the actual match. | |
| find | Show all files with path in current directory+subdirectory and so on | |
| find -name "filename with extension" | Find specific file in any folder | |
| find -maxdepth 2 -name "view\*" | Search Upto 2 folders | |
| find -type d -name "p1\*" | Directory | |
| find -type f -name "p1.\*" | text | |
| find -name "????????.txt" | 1 ? = for 1 character | |
| find -name "[!0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9].txt" | Looks for filename containing 8 digits | |
| grep --color "^200" shell/p1.txt | Colored 200 | |
| echo $? | Status of last command. If true gives 0. If false gives other than 0 | |
| ^503.{37} | Selects Upto col 41 | |
| uniq "p11.txt" | A  B  C  C  D  C  Output:  A  B  C  D  c | |
| uniq -c "p11.txt" | 1 a  1 b  2 c  1 d  1 c | |
| sort "p11.txt" | A  B  C  C  C  d | |
| sort p11.txt | uniq -c | 1 a  1 b  6 c  1 d | |
| sort -r filename | Orders in descending | |
| sort -n filename | Considers number | |
| sed 's/.\*\?n/fee/' green | Regx:  Anything ending with a n gets replaced with fee.  #? mark here- zero or 1 character before n | |
| sed 's/^.\?n/fee/' green | From the 1st, any character zero or 1 time before n | |
| sed 's/\w\*/fee/' txtfile | Replace each word | |
| sed 's/\w/fee/' txtfile | Replace each alphabet | |
| sed 's/.\*/fee/' txtfile | Replace each line | |
| sed 's/\(.\*\)/fee \1/' green | .\* => anything as manytimes  \1 indicates \(.\*\)  “fee” then rest of the part.  Fee red greenhy  Fee blue red  Fee blue red yellow | |
| sed 's/\(.\*\)n/fee \1/' green | (.\*) this indicates \1  The line which contains anything with a “n” is replaced with “fee \1” | |
| sed 's/\(.\*\)n.\*/fee \1/' green |  | |
|  |  | |
| (\w\*\) \(\w\*\)  \1 | regx | |
| (\w\*\) \(\w\*\)  \2 | Regx:  Replaces both words with the 2nd word | |
| ^(\w\*)  \1 \1 | Regx:  Replaces 1st word with two 1st words | |
| (\w+)  \1 \1 | Regx: | |
| (\w\*)$  \1 \1 | Matches from last | |
| ^(\w+)  \1 \1 | Matches from 1st. takes only the 1st word | |
| var="last"  sed 's/'"$var"'/blue/' test44.txt | Replace value of a variable.  You need to end a single quote before using variable. Bcz inside single code everything is taking its literal meaning.No special menaing | |
| var="last"  sed 's/'"$var"'/blue/g' test44.txt | Replace all first | |
| sed -e 's/blue/she/' -e 's/.\*\?n/fee/' green |  | |
| sed -f new1.sh green |  | |