Operazioni per la parte Shell

Rimuovere caratteri windows → sed -i -e 's/\r$//' nomefile.sh

Pwd (Printing Working Directory): mostra la directory corrente
Id: visualizza le informazioni (UID e GID) dell'utente ed inoltre groups
Echo $HOME: visualizza la nostra home directory
Echo $PATH: visualizza le directory in cui la shell deve ricercare ogni comando da eseguire **Ps (process status)**: processi attivi dell'utente

Opzione	**Descrizione**
-f (Full)	Fornisce maggiori informazioni
-l (Long)	Fornisce ulteriori informazioni come ad esempio se lo stato del processo è in S(Sleeping), R(Running)…
-e (Extended)	Fornisce ulteriori processi non visibili precedentemente

Man (manual): fornisce il manuale del comando fornito
Which: fornisce dove si trova il dato comando
Whereis: fornisce dove si trova il dato comando e dove si trova il suo manuale
Sh (shell): invoca una ulteriore shell (alla quale si esce con il comando exit)

Opzione	**Descrizione**
-x	Mostra cosa espande/sostituisce la shell prima di eseguire il comando
-v	Stampa le linee del file comandi come sono lette dallo shell

Who: visualizza utenti attivi in un sistema UNIX/LINUX
W: visualizza utenti attivi in un sistema UNIX/LINUX (simile a who ma fornisce risultati leggermente differenti) **Touch:** modifica la data di un file e gli viene immessa la data ed ora corrente / Crea un file vuoto. **Echo:** Serve per la scrittura su stdout o su file.
Vi/vim: comando editor di testo
Mkdir *nomedir* (make directory): costruisce una nuova directory
Rmdir *nomedir* (remove directory): cancella una directory VUOTA
Cd *nomedir* (change directory): modifica la directory corrente
Cp *filesorg dest* (copy): copia il filesorg nel file di nome dest, se dest è una directory copia il file con un nome filesorg in quella directory
Ln *nomefile1 nomefile2* (link): funzione link hardware

Opzione	**Descrizione**
-s (Software) *nomefile (in forma assoluta!)*	Crea un link software

Mv *nomefile1 nomefile2* (move): sposta il file nel grafo
Rm *nomefile* (remove): cancellazione di un file/directory. Le informazioni non sono eliminate se ci sono altri link (HW), viene decrementato nl; le informazioni sono eliminate solo quando nl è uguale a 0

Opzione	**Descrizione**

-i (interactive)	Richiede la conferma di eliminazione
-r/R	Rimuove la directory

Cat *nomefile1 … nomefilen* (concatenate): il contenuto dei file indicati viene riportato sullo standard output uno di seguito all'altro
More: il contenuto dei file indicati viene riportato sullo standard output uno di seguito all'altro in forma paginata **Ls (list):** fornisce la lista degli elementi nella directory

Opzione	**Descrizione**
-l (Long)	Lista, oltre al nome, tutte le informazioni associate ai file, cioè al tipo del file, permessi, numero di link, proprietario, gruppo, dimensione in byte, ora di ultima modifica
-a	Lista anche i nomi dei file "nascosti", cioè il cui nome inizia con '.'
-A (All)	Come -a, escludendo però . e ..

-F	Lista i nomi dei file visualizzando i file eseguibili con suffisso *, le directory con suffisso /
-d *nomedir*	Lista le informazioni associate alla directory considerata come file, senza listarne con il contenuto
-R (recursive)	Lista ricorsiva dei file contenuti nella gerarchia
-i	Lista gli i-number dei file oltre al loro nome
-r (reverse)	Lista i file in ordine opposto al normale ordine alfabetico
-t	Lista i nomi dei file in ordine di ultima modifica, dai più recenti, fino ai meno recenti

Chmod [u g o a] [+ -] [rwx] nomefile: Permette di modificare i diritti di file/directory
Chmod DirittiInOttale nomefile: Permette di modificare i diritti di file/directory (espressi in ottale) **Sort:** ordina le linee dello standard output

Opzione	**Descrizione**
-r (reverse)	Esegue l'ordinamento in inverso
-f	Esegue l'ordinamento ignorando le minuscole/maiuscole
-c (check)	Verifica se un file è ordinato o meno, inoltre riporta la prima linea che non verifica l'ordinamento
-C (check)	Verifica se un file è ordinato o meno. In

	questo caso NON riporta alcuna linea. Per verificare se è andato a buon fino oppure no utilizzare: echo $? (se tale valore torna 0 significa (in norma) successo altrimento significa INSUCESSO)
-u	Ordina eliminando le linee ripetute

Grep *stringa* (General Regular Expression Print): cerca dei pattern nel file di testo

Opzione	**Descrizione**
-n (number)	Riportare il numero d'ordine delle linee trovate (riporta ovviamente anche le linee)
-i (ignore)	Ignora la differenza fra maiucole o minuscole
-v	Invertire la ricerca, vengono mostrate solo le linee che NON contengono il pattern
'^*stringa/carattere*'	Cerca tutte le linee che incominciano per una cerca stringa/carattere
'*carattere*$'	Cerca tutte le linee che finiscono per un cerco stringa (**Nota bene:** se l'ultimo carattere è il . bisogna introdurre il metacarattere \ e quindi scivere \.)

Wc (word count): conta le linee, le parole ed i caratteri di un file di testo

Opzione	**Descrizione**
-l (length)	Ritorna solo il numero di linee
-w (word)	Ritorna solo il numero di parole
-c (characters)	Ritorna solo il numero di caratteri

N.B.: Se il risultato del comando wc ci serve per assegnarlo ad una variabile o per fare delle comparazioni con un numero, bisogna che lo usiamo con la ridirezione dello standard input!

Head *[-numerolinee]*: mostra le prime linee di un file di testo (se non specificato il numero di linee stampa le prime 10)

Tail *[-numerolinee]*: mostra le ultime linee di un file di testo (se non specificato il numero di linee stampa le ultime 10)

Rev: Rovescia i caratteri presenti, linea per linea, in un file di testo

Kill: serve per terminare un comando lanciato in background

Diff *nomefile1 nomefile2*: riporta le linee che presentano delle differenze nei due file

Date: fornisce la data ed orario corrente del sistema

Find *directory -name nomefile*: riporta tutti i nomi assoluti del file di cui viene fornito il nome a partire alla directory specificata

Env (environment): mostra l'ambiente corrente di ogni processo shell

Export *nomevariabile*: trasforma la variabile data in variabile di ambiente

Unset *nomevariabileambiente*: trasforma una variabile di ambiente in una variabile

normale **Eval:** utilizzato nel caso l'espansione risultante richieda nuove sostituzioni, bisogna richiedere esplicitamente alla shell di effettuarla
Shift: opera la traslazione del valore dei parametri verso il basso
Set: E' possibile riassegnare ai parametri (**set *expr1 expr2…***) gli argomenti del comando set sono assegnati secondo la posizione ai parametri
Test: usato in particolare nel costrutto if. Serve per la valutazione di una espressione che ha come valore di ritorno 0 in caso di successo, altrimenti un valore di ritorno diverso da zero in caso di insuccesso.

Opzione	**Descrizione**
-f *nomefile*	Esistenza file nomefile
-d *nomedir*	Esistenza directory nomedir
-r (w oppure x) *nomefile/dir*	Diritto di lettura (scrittura oppure esecuzione) sul file/dir
Stringa1* = *stringa2	Valuta se due stringhe sono uguali (**Nota bene:** ci deve essere uno spazio prima e uno dopo il metacarattere di =)
-z *stringa1*	Valuta se la stringa è nulla
Stringa1	Valuta se la stringa non è nulla
Numero1* [-eq -ne -gt -ge -lt -le] *numero2	Confronta tra loro due stringhe numeriche, usando uno degli operatori relazionali indicati

Read *var1 var2…*: le stringhe fornite in ingresso dall'utente (o prelevate da un file in ridirezione) vengono attribuite alle variabili secondo la corrispondenza posizionale

Ulteriori variabili:

Nome variabile	**Descrizione**
$*	Insieme di tutte le variabili posizionali, che corrispondono agli argomenti del comando: $1, $2, ecc
$#	Il numero di argomenti passati al comando (**$0 escluso**)
$?	Il valore di ritorno dell'ultimo comando eseguito
$$	Il numero del processo in esecuzione

Controlli

Controllo su un solo parametro: if test $# -ne 1 then exit 0 fi	**Controllo su due parametri:** if test $# -ne 1 -a $# -ne 2 then exit 0 fi
Controllo se percorso assoluto, relativo o relativo semplice: case $n in /*) Percorso assoluto; exit 1;;	**Controllo se esiste il file/directory:** if test ! -f/d $n then exit 0 fi

/) Percorso relativo; exit 2;; *) Relativo semplice; exit 3;; esac	#Se il file/la directory non esiste esce.
Eseguire operazioni: var=`expr $var +/-/*// qualcosa` #tra le variabili bisogna lasciare uno spazio prima e dopo Oppure: echo $((Var=Operazioni)) > /dev /null **Utilizzo di grep:** grep [range] variabile	**Ricorsione (in fondo al file ricorsivo):** for i in * if test -d $i -a -x $i then filericorsivo.sh $1/$i $2 fi done
Settaggio variabile PATH (Prima della chiamata del file FCR.sh) PATH=`pwd`:$PATH Export PATH	**Numero di linee in cui è contenuto qualcosa:** n=`grep condizione $var | wc -l`
Lettura inserimento utente: `read risposta` `case $risposta in` `  s* | S* | y* | Y*);;`	**Creazione di file temporaneo:** >/tmp/tmp 1 solo >/tmp/tmp$$ cambia in base all'id del processo

`  *);;` `esac`	
Lettura righe file e scrittura del contenuto: `List=wc -l < /tmp/Files` `echo Numero files creati: $List` `for i in cat /tmp/Files` `do` `  echo File=$i contiene\:` `  echo -n Contenuto:` `  more $i` `done` `rm /tmp/Files`	**Ottenere la linea desiderata:** `Linea=head -5 $i \| tail -1` **Ordinare un file tmp:** `for i in cat /tmp/File` `do` `  echo "$i wc -c<$i"` `  echo "Do you want to sort this file?(yes/no)">/dev/tty` `  read answ` `  case $answ in` `  y* | Y*) sort -f $i;;` `  *)echo $i not sorted;;` `  esac` `done`
Controllare se il parametro è un numero: case $1 in *[!0-9]*) echo Errore\(02\)\: \$1 non un numero; exit 2;; *) if test $1 -eq 0 then echo Errore\(03\)\: \$1 uguale a 0; exit 3 fi ;; esac	**Utilizzare una variabile per contenere il nome dei file:** files= files="$files $F" chiamata al main.c main $files chiamata ricorsiva: for i in * do if test -d $i -a -x $i then #chiamata ricorsiva $0 `pwd`/$i $2 $3 fi done

```
#Inserimento dell'ultimo parametro in un variabile
#
Count=1
for i in $*
do
        if test $Count -eq $#
        then Last=$i
                #echo \$Last=$Last
        fi
        Count=`expr $Count + 1`
done

#Controllo sulle gerarchie (da fare sempre in caso di gerarchie).
#
Count=1
for i in $*
do
        if test $Count -ne $#
        then
                case $i in
                        /*) if test ! -d $i -o ! -x $i
                                 then echo Error\(03\): Parameter must be a directory;
exit 3
                                fi;;
                        *) echo Error\(04\): Parameter is not a directory; exit 4;;
                esac
        fi
        Count=`expr $Count + 1`
done
```

Chiamata ricorsiva con controllo se il parametro non directory viene inserito (se non in prima posizione, altrimenti utilizzare shift)
#

```
Count=1
for i in $*
do
        if test $Count -ne $#
        then
                FCR.sh $i $Last /tmp/File
        fi
        Count=`expr $Count + 1`
done
```

Indicazioni generali file esame:

File FCP.sh:

1. Controllare il numero di parametri se necessario con $#.
2. Fare i necessari controlli sui parametri, dipendentemente dalla richiesta.
3. Creare il file temporaneo se richiesto.
4. Effettuare il controllo sulle gerarchie (vedi sopra).
5. Esportare la variabile PATH.
6. Effettuare la chiamata ricorsiva del file FCR.sh.
7. Nel caso richiesto stampare a richiesta il file temporaneo.
8. Eliminare, se esiste, il file temporaneo.

File FCR.sh:

1. Spostarsi nella directory passata come parametro
(ATTENZIONE ai parametri). 2. Effettuare i controlli e i comandi
richiesti dalla consegna.
3. Implementare il for per la ricorsione.

Per chiamare la parte c bisogna scrivere: NomeEseguibileGenerato p1 p2 … pn