BASH

1) Dato un file 'hotel.txt' che contiene la lista degli alberghi di una città con il costo giornaliero nel formato 'albergo : costo', scrivere uno script che stampa gli alberghi con costo tra un minimo e un massimo. I costi minimo e massimo sono dati come parametri di linea allo script che quindi viene attivato come
$script hotel.txt <min> <max>

Una Soluzione.

```
declare i
min max costo
if(($# != 3))
then
     echo "argomenti insufficienti"
     exit
fi
min=$2
max=$3
file=$1
while read linea
do
     costo=$(echo $linea|cut f22 d':')
     hotel=$(echo $linea|cut f11 d':')
     if(($costo > $min)) && (($costo < $max))
     then
          echo "l'albergo $hotel costa $costo euro"
     fi
done < $file
```

2) Scrivere uno script in Bash che realizza le seguenti funzioni usando un menu di interazione testuale con l'utente:
"premi [1] per l'elenco dei file regolari"
"premi [3] per eseguire 'nano&' sul file dato come argomento"
"premi [4] per stampare il contenuto del file dato come argomento di linea"
"premi [5] se vuoi uscire da questo script"

Una Soluzione.

```
while (:)
do
     echo
     echo "premi 1 per l'elenco dei file regolari"
     echo "premi 2 per eseguire nano& sul file"
     echo "premi 3 per stampare il contenuto del file"
     echo "premi 4 per uscire"
     echo
     read cmd
     case $cmd in
     1)
     for file in *
     do
          if [[ f $file ]]
          then
               echo "file regolare $file"
          fi
     done;;
     2)
     nano $1;;
     3)
          cat $1;;
     4) exit;;
     esac
done
```

3) Scrivere uno script chiamato 'mysed.sh <string1> <string2> <nomefile>' che rimpiazza tutte le occorrenze di string1 con string2 <b> in tutte le righe del file nomefile.
NB. Il comando per rimpiazzare una sottostringa s1 con la sottostringa s2 nella stringa s è: `${s/s1/s2}`

Una Soluzione.

```
#questo script funziona come lo stream editor sed
#
if(($# != 3))
then
     echo "argomenti insufficienti"
     exit
fi
s1=$1
s2=$2
file=$3
while read linea
do
     riga=${linea/$s1/$s2}
     echo $riga
done < $file >> temp
rm $file
mv temp $file
```

4) Scrivere uno script per cercare una password chiamata password ottenuta con un generatore casuale da 0 a 9 . L'algoritmo di ricerca sia semplicemente del tipo illustrato in questo pseudocodice:

```
while(not found) {
        leggi key;
        if(key==password) stampa “password trovata!”; esci;
        if(key>password) scrivi(dare un mnumero minore) else
        scrivi(dare un numero maggiore)
}
```

NB. Per generare la password scrivere una funzione che genera numeri casuali da 0 a 9 usare la variabile d'ambiente RANDOM tenendo conto che fornisce numeri da 0 a 32767.

Una Soluzione.

```
function genera
{
     ((a=$RANDOM % 10))
     return $a
}
nf=1 #not found flag
genera # genera una password casuale da 0 a 9
pwd=$?
echo "scrivi un numero da 0 a 9"
while [[ $nf = 1 ]]
do
     read n
     if [[ $n = $pwd ]]
     then
          echo "password trovata: $pwd"
          nf=0
     elif [[ $n > $pwd ]]
     then
          echo "scrivi un numero piu' piccolo"
          else
          echo "scrivi un numero piu' grande"
     fi
done
```

5) La struttura semplificata di un documento HTML è¨:
<HTML>
<BODY>
Contenuto della pagina
</BODY>
</HTML>
Considerando i seguenti tag da porre all'interno del campo BODY:
<h1> … </h1> per creare intestazioni del testo di primo livello
<h2> … </h2> per creare intestazioni del testo di secondo livello

 per inserire un nuova riga,
scrivere uno script in Bash, chiamato txt2html.sh, per tradurre un file di testo in un file html. Il file di testo e' formattato in un modo simile a questo:

primolivello: testo di primo livello testo di primo livello
secondolivello: testo di secondo livello testo di secondo livello
primolivello: testo di primo livello testo di primo livello
nuovariga: continua il testo di primo livello testo di primo livello

Cioè lo script mette il testo di primo livello dentro una coppia <h1> … </h1>, il testo di secondo livello dentro una coppia <h2> … </h2> e mette due tag
 al posto di 'nuovariga'.

Una Soluzione.

```
echo "<html>"
echo "<body>"
while read linea
do
     testo=$(echo $linea|cut d'' f2)
     tag=$(echo $linea | cut d'' f11)
# echo "testo=$testo" #debub
# echo "tag=$tag" #debug
     case $tag in
          primolivello:) echo "<h1>$testo</h1>";;
          secondolivello:) echo "<h2>$testo</h2>";;
          nuovariga:) echo "<br><br>$testo";;
     esac
done
echo "</body>"
echo "</html>"
```

L'uso dello script è, evidentemente:
```
$txt2html.sh < file.txt > file.html
```

```
#!/bin/bash
#Questo script legge tutti i file con estensione sh
#selezionando solo le righe che contengono la
#parola Giorgio e le scrivono accodandole sul
#file commenti
#Lo script non ha quindi parametri in linea.
#La sua attivazione può essere
#./script >> commenti
#
for n in *.sh
do
     print “file $n”
     while read linea
     do
          if [[ $linea = “#”* ]]
          then
               for nn in $linea
               do
                    if [[ $nn = “Giorgio” ]]
                    then
                         print “$linea”
                    fi
               done
          fi
     done < $n
done
```

```
#!/bin/bash
#Questo script legge un file di testo (in cui nome viene dato in
#linea) e accoda in un secondo file (anch'esso dato in linea) solo
#le righe che contengono parole che cominciano (opzione –d) o
#finiscono (opzione –c) con un numero.
#
#script [-d][-c] <nome1> <nome2>  Il nr. corretto di parametri e' 3
#
USAGE="script [-d][-c] <file1> <file2>"
if (( $# != 3 ))
then
    echo "errore nei parametri!"
    echo "$USAGE"
else
    case $1 in
    -d)     while read linea
            do
                 for n in $linea
                 do
                      if [[ $n = ["0"-"9"]* ]]
                      then
                           echo "$linea"
                      fi
                 done
              done < $2 >> $3 ;;
      -c)   while read linea
              do
                 for n in $linea
                 do
                      if [[ $n = *["0"-"9"] ]]
                      then
                           echo "$linea"
                      fi
                 done
              done < $2 >> $3;;
          *) echo "switch errato!";;
      esac
fi
```

```
#!/bin/bash
# questo script aggiunge una intestazione data ai file specificati
#script "lista file" "intestazione"
for i in $1    #per tutti i file specificati
do
          a="# $2 "
          echo $a > temp
          cat temp $i > temp1
          mv temp1 $i
done
```