Esame del 09/02/2023 ([GitHub](https://github.com/davide-dipierro/Prove-svolte-LSO/tree/main/09222023_B))

**Esercizio 1**

1. *Utilizzando awk si scriva un comando che stampi una lista dei file presenti nella directory corrente mostrando solo dimensione e nome.*

ls -l | awk '{print $5, $9}'

1. *Si calcoli la dimensione occupata in totale dai file regolari con dimensione maggiore di 1024 byte nella directory corrente*

ls -l | awk 'BEGIN {sum=0} { if ($5 > 1024 && substr($1,1,1) == "-") sum+=$5} END {print "sum="sum}'

ls -l | awk 'BEGIN {sum=0} { if ($5 > 1024 && $1 ~ /^-/) sum+=$5} END {print "sum="sum}'

1. *Si faccia in modo che il comando stampi solo i file maggiori di 1024 byte*

ls -l | awk '{ if ($5 > 1024) print $0 }'

1. *Trovare i file non acceduti negli ultimi 30 giorni*

ls -l --time-style=+%s | awk 'BEGIN { now=systime() } { if(now-$6 < 60\*60\*24\*30) print $0 }'

**Esercizio 2**

*Si realizzi uno script di shell BASH “menu”, che accetta come argomento un file “listino.txt” strutturato nel seguente modo:*

|  |  |  |
| --- | --- | --- |
| codice | quantità | costo |
| 01953 | 2 | 15 |
| 07934 | 1 | 20 |
| 084Gd | 10 | 30 |
| 9038H | 1 | 5 |

*e che implementi le seguenti funzioni accessibili da un menu:*

1. *Cerca - Chiede all’utente una stringa da ricercare all’intero del listino ed effettua la ricerca*
2. *Aggiungi - Chiede all’utente il codice del prodotto da aggiungere (primo campo del listino) e la quantità di articoli desiderati, verifica le scelte effettuate e le memorizza in un file carrello*
3. *Elimina - un prodotto dal carrello*

#!/bin/bash

echo "Benvenuto in menù"

if [ $# -ne 1 ]

then

     echo "Utilizzo: $0 <nomefile>"

     exit 1

fi

if [ ! -f $1 ]

then

     echo "Il file $1 non esiste"

     exit 2

fi

echo "Cosa vuoi fare?"

echo "1) Cerca"

echo "2) Aggiungi"

echo "3) Elimina"

echo "4) Esci"

read scelta

case $scelta in

     # Cerca

     1) echo "Inserisci il codice"

         read codice

         grep $codice $1

         ;;

     # Aggiungi

     2) echo "Inserisci il codice"

         read codice

         if [ $(grep -c $codice $1) -eq 0 ]

         then

             echo "Il codice non esiste"

             exit 3

         fi

         echo "Inserisci la quantità"

         read quantita

         quantita\_file=$(grep "$codice" $1 | awk '{print $2}')

         if [ $quantita\_file -lt $quantita ]

         then

             echo "Non ci sono abbastanza prodotti"

             exit 4

         fi

         echo $codice $quantita $costo >> carrello.txt

         ;;

     # Elimina

     3) echo "Inserisci il codice da eliminare"

         read codice

         sed -i "/$codice/d" carrello.txt

         ;;

     # Esci

     4) echo "Arrivederci"

         exit 0

         ;;

     # Scelta non valida

     \*) echo "Scelta non valida"

         exit 3

         ;;

esac

**Esercizio 3**

*Si realizzi un programma C il cui processo padre P0 dia il via alla generazione di n processi in gerarchia lineare. Dove n è un numero intero passato come argomento al programma. Cioè, P0 genera P1, P1 genera P2, ..., Pn genera Pn+1. Il P0 deve prendere in input una sequenza di N comandi (per semplicità, senza argomenti e senza opzioni). Ogni N processo deve eseguire il rispettivo N. L’applicazione termina quando l’ultimo processo ha terminato.*

#include "stdio.h"

#include "stdlib.h"

#include "unistd.h"

#include "sys/wait.h"

int main(int argc, char \*argv[]) {

    printf("\nEsecuzione di %s\n\n", argv[0]);

    for(int i = 1; i < argc; i++) {

        int pid = fork();

        if(pid == 0) {

            execlp(argv[i], argv[i], NULL);

            printf("Errore nell'esecuzione di %s\n", argv[i]);

            exit(1);

        }

    }

    while (wait(NULL) > 0);

    printf("Esecuzione terminata\n");

    return 0;

}