1. V lupini *bash* napišite program *poslji*, ki za vsakega uporabnika prijavljenega na sistemu vpraša, ali naj mu pošlje obvestilo, shranjeno v datoteki *obvestilo*. Če je odgovor da (D), mu to obvestilo pošlje, sicer ne.

```
for i in `who | cut -d" " -f1`
do
  echo "posljem msg $i?"
  read odg
  if [ $odg = "d" ]; then
    write $i<obvestilo
  fi
done
```

2. Ukaz *find [dir] file* išče datoteko *file* v delovnem direktoriju ali direktoriju *dir*, nato pa rekurzivno po datotečni strukturi od tega direktorija navzdol. Vzemimo, da tega ukaza ni na sistemu in ga nadomestimo z ukazno datoteko napisano v lupini *bash*. Napiši tako datoteko.

```
for f in `ls $1`; do
  if [ -d $1/$f ]; then
    $0 $1/$f $2
  fi
  if [ $f = $2 ]; then
    echo $1/$f
  fi
done
```

3.Napišite program v lupini, ki pregleda datoteko /etc/passwd in izpiše vse lupine, ki jih uporabniki sistema uporabljajo, ter koliko uporabnikov uporablja posamezno lupino.

```
cut -f 7 -d : /etc/passwd | sort -r | uniq -c >bla
cut -c 1-5 bla > num
cut -c 6- bla | grep "/" > shell
echo brez lupine >>shell
paste shell num
rm num
rm shell
```

4.Napišite program v lupini, ki za vse datoteke v nekem direktoriju ter rekurzivno v vseh poddirektorijih izpiše datum njihov zadnje spremembe in zadnjega dostopa. Direktorij naj bo podan v ukazni vrstici.

```
ls -alRSa $1 > acesstime.log
ls -alRSc $1 > changetime.log
cat acesstime.log | sort +8 > accessSorted
cat changetime.log | sort +8 > changeSorted
cut -c43-55 accessSorted >a
cut -c43-70 changeSorted >b
paste a b > x
```

5.Napišite program v lupini, ki preišče vse datoteke v podanem direktoriju in njegovih poddirektorijih in izpiše najnovejšo datoteko (po datumu zadnje spremembe).

```
ls  -l -R | cut -c 43- |
sed -e s/Jan/01/ | sed -e s/Feb/02/ |sed -e s/Mar/03/ |sed -e s/Apr/04/ |
sed -e s/May/05/ | sed -e s/Jun/06/ |sed -e s/Jul/07/ |sed -e s/Aug/08/ |sed -e s/Sep/09/ |
sed -e s/Oct/10/ |sed -e s/Nov/11/ |sed -e s/Dec/12/ |
sort -r -k 1.7 |sort -r -k 1.4  | sort -r >>temp1
more -P -s temp1
echo -n Najnovejsa je:
more temp1 | head -1
echo --
rm temp1
```

6.Napišite program v lupini, ki v trenutnem direktoriju preimenuje datoteke s podano končnico v drugo končnico. Končnici sta podani v ukazni vrstici

```
#!/bin/bash
if test $# != 2
then
  echo "Uporaba : naloga7 koncnica1 koncnica2"
fi
for datoteka in `ls *$1 2>/dev/null`
do
```

mv $datoteka `basename $datoteka $1`$2
done

7.Napišite program v lupini, ki vsake pet minut preveri, če se na sistemu pojavi nek proces in ga poskusa ubiti. Ime procesa podamo v ukazni vrstici

```
while true; do
    ps -eo "%t %c %p" | grep $1 > dd
    while read p; do
        if [ `echo $p | grep -o ":" | wc -l` -eq 1 ]; then
            if [ `echo $p | cut -d":" -f1` -lt 5 ]; then
                kill `echo $p | awk '!/(Avail|shm|udev)/{print $3}'`
            fi
        fi
    done < dd
    sleep 300
    rm dd
done
```

8.Napišite program v lupini, ki beleži dnevnik dosegljivosti računalnikov. Kot argument mu podamo datoteko, ki vsebuje spisek računalnikov, program pa naj vsakih pet minut preveri, če so dosegljivi. V neko datoteko (dnevnik) naj zapisuje, če kateri od računalnikov ni dosegljiv, njegovo ime in čas, ko ni bil dosegljiv.

while true
do
for i in `cat spisek.txt`
do
ping -c 2 $i &>output|| echo "$i ni dosegljiv ob `date +%T`">>logfile
done
sleep 3
done

9. Napišite program v lupini, ki pregleda datoteko /etc/passwd in izpiše vse lupine, ki jih uporabniki sistema uporabljajo, ter koliko uporabnikov uporablja posamezno lupino.

```
cut -f 7 -d : /etc/passwd | sort -r | uniq -c >bla
cut -c 1-5 bla > num
cut -c 6- bla | grep "/" > shell
echo brez lupine >>shell
paste shell num
rm num
rm shell
```

10. Napišite program *izpisi [dir]* v lupini *bash*, ki za vsako datoteko v direktoriju *[dir]* ter rekurzivno v vseh poddirektorijih interaktivno vpraša ali naj to datoteko prekopira v domači direktorij uporabnika (~). Če je odgovor da, (Y) potem to datoteko tja prekopira, sicer pa ne.

```
[$# -eq0]&& set--
for i in $1/*
do
        if[-d $i];then
                izpis $i
        fi
        if [-f $i];then
                echo -e "Kopiram? [dn]\c"
                vprasaj && cp $i ~/
        fi
done
```

11. V lupini bash napišite program fakulteta, za izračun fakultete števil, n!. Program naj prebere n kot argument ukazne vrstice.

```
declare -i n=$1
declare -i i=n
let "i=i-1"
while [ $i -gt 1 ]; do
   let "n=n*i"
   let "i=i-1"
```

```
done
echo $n
```
12. V lupini bash napisite program zaporedje, ki izracuna in zapise prvih n stevil zaporedja ki je definirano kot Y(i)=Y(i-1) + 3 * Y(i-2), i>=2
in sta Y(0) = 1 in Y(1) = 2. Program naj prebere n kot argument ukazne vrstice.
```
Y[0]=1
Y[1]=2
echo "Y[0] = 1"
echo "Y[1] = 2"
n=$1
i=2
while [ $i -lt $n ]; do
   declare -i a=${Y[`echo "$i-1"|bc`]}
   declare -i b=${Y[`echo "$i-2"|bc`]}
   Y[$i]=`echo "$a + ( 3 * $b )" | bc`
```
13. napisat program v lupini, ki v danem direktoriju izbriše datoteke s "slabimi" imeni(ki vsebujejo "|", "@", " €" .... neki takega). Uporabnik mora potrditi brisanje vsake datoteke.
```
for dat in `ls -f |  grep ["|@€"]` ; do
echo "naj pobrisem $dat? 1 brise"
read odgovor
if [ $odgovor -eq 1 ] ;then
rm $dat
echo "$dat pobrisana"
fi
done
```
14. Iz foruma, gostota procesa ki je podan kot argument.
```
>today
>yesterday
zacetek=`date +%d`
while true ; do
novdan=`date +%d` # kej takega pomojem fali v skripti
if [ $novdan -ne $zacetek ] ;then
cp today yesterday
>today
zacetek=$novdan # da nebo spet padel v to zanko cez 30 sekund
fi
echo "dne `date +%d` je stevilo iskanih procesov">>today
ps -e | grep $1 | wc -l >>today
sleep 30
done
```
15. Napišite ukazno datoteko *gostota* v lupini *bash*, ki beleži gostoto uporabe nekaterih procesov na sistemu. Ukazna datoteka sprejema en argument, ki je ime datoteke, ki hrani imena procesov, ki nas zanimajo. Evidenco vodite za ne več kot dva dneva nazaj. Gostoto preverjajte vsakih pet minut, evidenca pa naj hrani tudi informacijo o datumu, uri in seveda procesih.
```
if [ $# -ne 1 ]
  then
    echo 'dovoljen samo en parameter: ime datoteke s seznamom procesov'
  else              # tu se pa zacne resno delo
    echo >evidenca1
    echo >evidenca2
    echo >evidenca
    let dan=`date +%j` # date +%j vrne stevilko dneva v letu
    while true        #neskoncna zanka
      do
        let ndan=`date +%j`
        if [ $dan -ne $ndan ] #arhiv za dva dni nazaj
          then
            rm evidenca2
            cp evidenca1 evidenca2
            cp evidenca evidenca1
```

```
        echo > evidenca
       fi
      date >>evidenca
      for neki in `cat $1` # NAVEDNICE!!!!
       do
        let stevec=0
        ps -A|cut -c25-40 |grep $neki >seznam
        for drugo in `cat seznam`
        do
         let stevec=$stevec+1
        done # FOR
        echo $neki : $stevec krat >>evidenca
        rm seznam
       done # FOR
      echo >>evidenca
      sleep 300 #spanje 300 sekund
    done # WHILE
fi
```

16. V lupini bash napišite program preveri_geslo, ki preveri veljavnost danega gesla glede na dane pogoje. Geslo naj bo podano kot argument programa. Veljavna gesla mora zadoščati naslednjih pogojem: 1.) Minimalna dolžina je osem znakov; 2.) Vsaj en znak mora biti numeričen znak in; 3.) Vsebovati mora vsaj enega od naslednjih ne-alfanumeričnih znakov. @ #, $ ali %.

```
int main(int argc, char **argv) {
  char ukaz[100] = "/bin/";
  char *parametri[100];
  //poberemo argumente
  if(argc == 1) {
    printf("Ni podanih argumentov\n");
    return 0;
  }  else if(argc > 1) {
    int index = 1;
    strcat(ukaz, argv[1]);   //nastavimo ukaz, nastane npr. /bin/ls , ce klicemo da izvrsi ls
    parametri[0] = ukaz;
    //prepisemo argumente v array
    int counter;
    for(counter = 2; counter < argc; ++counter){
      parametri[index] = argv[counter];
      index++;
    }
  }
  int pid, return_status;
  int pogoj = 0;
  while(pogoj == 0) {
    pid = fork();
    if (pid < 0) {  //new proccess failed
          printf("Napaka pri zagonu novega procesa!\n");
          pogoj = 1;
    } else if (pid == 0) {
      execv(ukaz, parametri);
      exit(1);
    } else {
      wait(&return_status);
          if (WEXITSTATUS(return_status) != 0) {
        printf("Napaka pri zagonu ukaza!\n");
        pogoj = 1;
      }
       }
    sleep(5);
```

```
    }   return 0;
}
```