6.1.1 Segundoak ordutan

segundo2ordu.pl

```
#!/usr/bin/perl -w

use strict;

my ($ordu, $min, $seg, $segundoKop);


$| = 1;


print "Eman segundo-kopurua:\n";
$segundoKop = <>;
chomp($segundoKop);
$seg = $segundoKop;
$ordu = int($seg / 3600);
$seg = $seg - 3600*$ordu;
$min = int($seg / 60);
$seg = $seg - 60*$min;

print "$segundoKop segundoak, $ordu ordu, $min minutu eta $seg segundo
dira\n";
```

6.1.2 Zenbat segundo

ordu2segundo.pl

```
#!/usr/bin/perl -w

use strict;

my ($ordu, $min, $seg);

print "Eman 3 zenbaki, bakoitza lerro batean: orduak, minutuak eta
segundoak:\n";
$ordu = <>;
chomp($ordu);
chomp($min = <>);
$seg=<>;
chomp($seg);

my $emaitza;

$emaitza = $ordu * 3600 + $min * 60 + $seg;
print "Segundo-kopurua: $emaitza\n";
```

6.1.3 Celsiusetik Farenheitera

celsius2farenheit.pl

```
#!/usr/bin/perl -w

use strict;

my ($celsius, $faren);

print "Eman Farenheit sisteman adierazitako tenperatura:\n";
$faren = <>;
chomp($faren);
```

```
$celsius = ($faren-32)*(10/18);
print "$faren -> Farenheit\n";
print "$celsius -> Celsius\n";
```

6.1.4 Pezeta eta euroen bihurtzailea

euroKonbert.pl

```
#!/usr/bin/perl

use strict;

my ($kop,$jarraitu,$opzioa,$emaitza);

print "Sartu kopuru bat:";
$kop = <>;
chomp($kop);
$jarraitu=1;
while ($jarraitu) {
  print "$kop kopurua sartu duzu. Egin zure aukera.\n 1) Euro->pta\n 2)
pta->Euro\n";
  $opzioa = <>;
  chomp($opzioa);
  if ($opzioa == 1) {
    $emaitza = $kop * 166.386;
    print "$kop euro $emaitza peseta dira.\n";
    $jarraitu=0;
  } else {
    if ($opzioa == 2) {
      $emaitza = $kop / 166.386;
      print "$kop peseta $emaitza euro dira.\n";
      $jarraitu=0;
    } else {
      print "Aukera baliogabea. Saia zaitez berriro.\n\n";
    }
  }
}
```

6.3.1 Ordenatu bi zenbaki

txikiHandi.pl

```
#!/usr/bin/perl -w

my ($z1, $z2);

print "Sartu 2 zenbaki, bakoitza lerro batean\n";
$z1 =<>;
$z2 =<>;
chomp($z1);
chomp($z2);
if ($z1 > $z2) {
  print "$z1, $z2\n";
} else { # $z2 >= $z1
  print "$z2, $z1\n";
}
```

6.3.2 Asmatu zenbaki bat

zbkAsmatu.pl

```
#!/usr/bin/perl -w
print "Inork ikusi gabe, sar ezazu zenbaki bat:\n";
$zbk = <>;
chomp($zbk);
# system("clear"); # Pantaila garbitu
$jarraitu=1;
print "Zenbaki bat gordeta dut. Zein izango da ?\n";
while ($jarraitu != 0) {
  print "Egin zure aukera:";
  $aukera = <>;
  chomp($aukera);
  if ($aukera == $zbk) {
    print "Arraioa. Asmatu duzu eta !\n";
    $jarraitu = 0;
  } elsif ($zbk > $aukera) {
    print "... mmmm, nik daukadan zenbakia $aukera baino handiagoa da \n";
  } else {
    # puntu honetan, beti gertatuko da $zbk < $aukera dela
    print "... mmmm, nik daukadan zenbakia $aukera baino txikiagoa da \n";
  }
}
```

6.3.3 Balio absolutua

absolutu.pl

```
#!/usr/bin/perl -w

my ($zbk);

# $| = 1;

print "Sartu zenbaki bat:";
$zbk =<>;
chomp($zbk);


if ($zbk < 0) {
      $zbk = - $zbk;
    };
print "Zenbakiaren balio absolutua hau da: $zbk\n\n";
```

6.3.4 Lehenengo zenbakia bigarrenaren multiploa

multiplo.pl

```
#!/usr/bin/perl -w
my ($zbk1, $zbk2, $hondarra);
print "Sartu bi zenbaki, bakoitza lerro batean:\n";
$zbk1 =<>;
chomp($zbk1);
$zbk2 =<>;
chomp($zbk2);

$hondarra = $zbk1 % $zbk2;
print "$zbk1 eta $zbk2 zenbakien zatiketaren hondarra $hondarra da.\n\n";

if ($hondarra == 0) {
         print "$zbk1 zenbakia $zbk2 zenbakiaren multiploa da.\n\n";
```

```
  }
else { print "$zbk1 zenbakia ez da $zbk2 zenbakiaren multiploa.\n\n"; }
```

6.4.1 Zenbat bokal lerroan

zenbatBokal.pl

```
#!/usr/bin/perl

use strict;
use locale;  # ñ-ak etab. kontuan hartzeko

my ($lerro, $zenbat);

print "Idatzi lerro oso bat:";
$lerro = <>;
chomp($lerro);
$zenbat=0;
while ($lerro =~ /[aeiou]/gi) { # Espresio erregularraren opzioak:
                                # g globala
                                # i maiuskula/minuskula ez bereizi
  $zenbat++; # $zenbat = $zenbat + 1
}
print "Lerroan $zenbat bokal daude.\n"
```

6.4.2 Zenbat letra lerroan

zenbatLetra.pl

```
#!/usr/bin/perl

use strict;
use locale;  # ñ-ak etab. kontuan hartzeko

my ($lerro, $zenbat);

print "Idatzi lerro oso bat:";
$lerro = <>;
chomp($lerro);
$zenbat=0;
while ($lerro =~ /[\w]/gi) { # Espresio erregularraren opzioak:
                                # g globala
                                # i maiuskula/minuskula ez bereizi
  $zenbat++; # $zenbat = $zenbat + 1
}
print "Lerroan $zenbat letra daude.\n"
```

6.4.3 Karakterea zenbatetan lerroan

zenbatKar.pl

```
#!/usr/bin/perl

use strict;
use locale;  # ñ-ak etab. kontuan hartzeko

my ($lerro, $kar, $zenbat);

print "Idatzi lerro oso bat:";
$lerro = <>;
chomp($lerro);
print "Zein karaktere zenbatu nahi duzu:";
$kar=<>;
chomp($kar);
$zenbat=0;
while ($lerro =~ /$kar/gi) { # Espresio erregularraren opzioak:
                             # g globala
                             # i maiuskula/minuskula ez bereizi
  $zenbat++; # $zenbat = $zenbat + 1
}
print "$kar karakterea $zenbat aldiz agertzen da lerroan.\n"
```

6.4.4 Zenbat karaktere fitxategian

zenbatKarFitx.pl

```
#!/usr/bin/perl

use strict;
use locale;  # ñ-ak etab. kontuan hartzeko

my ($lerro, $kar, $fIzena, $zenbat);

print "Zein karaktere zenbatu nahi duzu ?:";
$kar=<>;
chomp($kar);
print "Zein fitxategian begiratu nahi duzu ?:";
$fIzena=<>;
chomp($fIzena);

if (!open (FI, $fIzena)) {
  die "Ezin dut $fIzena fitxategia ireki !\n"; # die -> Mezua igorri eta
programatik irten
}

$zenbat=0;
while ($lerro = <FI>) {
  chomp($lerro);
  while ($lerro =~ /$kar/gi) { # Espresio erregularraren opzioak:
                               # g globala
                               # i maiuskula/minuskula ez bereizi
    $zenbat++; # $zenbat = $zenbat + 1
  }
}
print "$kar karakterea $zenbat aldiz agertzen da $fIzena fitxategian.\n";
close(FI); # Fitxategia itxi
```

6.5.1 Letra bakoitza zenbatetan fitxategian

```
% letraMaiztasuna.pl
#!/usr/bin/perl
use strict;
use locale;  # ñ-ak etab. kontuan hartzeko

my ($lerro, $kar, $fIzena,  %frek, $gako, $balio);

print "Zein fitxategitan begiratu nahi duzu ?:";
$fIzena=<>;
chomp($fIzena);

open (FI, $fIzena);
while ($lerro = <FI>) {
  chomp($lerro);
  while ($lerro =~ /\w/gi) {  # Espresio erregularraren opzioak:
                              # g globala
                              # i maiuskula/minuskula ez bereizi
    $kar = $&; # parekatu-berria den karakterea
    if ($frek{$kar}) {     # Karaktere hori aurretik bazegoen ?
      $frek{$kar}++;       # Bai. Gehitu, beraz, bere agerpen-kopurua
    } else {
      $frek{$kar} = 1;     # Ez. Hasieratu bere agerpen-kopurua
    }
  }
}
close(FI); # Fitxategia itxi



foreach $kar (sort (keys(%frek))) {
   print "$kar $frek{$kar} \n";
}

print "----------------------------------------\n";
print "Orain maiztasunaren arabera aterako da:\n";
print "----------------------------------------\n";

# Hurrengo agindua konplexu-samarra da.
# Honekin, hash baten gakoak atxikitua duten balioen arabera
# ordenatuko dira, balio handiena duena lehen agertuko delarik.

foreach $kar (sort ({$frek{$b} <=> $frek{$a}} (keys %frek))) {
  print "$kar\t$frek{$kar}\n";
}
```

6.5.2 Bigramak zenbatetan fitxategian

```
% bigramaMaiztasuna.pl
#!/usr/bin/perl

use strict;
use locale;  # ñ-ak etab. kontuan hartzeko

$| = 1;

my ($lerro, $kar, $aurrekokar, $fIzena,  %frek, $gako, $balio);

print "Zein fitxategitan begiratu nahi duzu ?:";
$fIzena=<>;
chomp($fIzena);

if (!open (FI, $fIzena)) {
  die "Ezin dut $fIzena fitxategia ireki !\n"; # die -> Mezua igorri eta
programatik irten
}


while ($lerro = <FI>) {
  chomp($lerro);
  $kar = ' ';                        $aurrekokar = ' ';
  while ($lerro =~ /\w/gi) { # Espresio erregularraren opzioak:
                             # g globala
                             # i maiuskula/minuskula ez bereizi
    $aurrekokar = $kar;
    $kar = $&; # parekatu-berria den karakterea
    if ($frek{$aurrekokar . $kar}) { # bikote hori aurretik bazegoen ?
      $frek{$aurrekokar . $kar}++;   # Bai. Gehitu, beraz, bere agerpen-
kopurua
    } else {
      $frek{$aurrekokar . $kar} = 1; # Ez. Hasieratu bere agerpen-kopurua
    }
  }
}
close(FI); # Fitxategia itxi

while (($gako,$balio) = each(%frek)) {
  print "$gako bikotea $balio aldiz agertu da\n";
}
print "------\n";

# Hala ere, seguruenik, frekuentzi horiek ordenatuak nahi ditugu, hau da,
# maiztasun handiena izan duen letra lehendabizi etab.

# Ondorengo lerro honek hasharen gakoak ardenatuko ditu, balioen arabera.
Zenbakizko ordenazioa
# burutuko du, eta balio handiak txikiak baino aurrerako egongo dira

my @gakoOrdenaduak = sort {$frek{$b} <=> $frek{$a}} keys(%frek);
my $gk;
foreach $gk (@gakoOrdenaduak) {
  print "$gk bikotea $frek{$gk} aldiz agertu da\n";
}
```

6.5.3 Hitzen maiztasuna lerroan

lerro1HitzMaiztasun.pl

```
#!/usr/bin/perl -w

use strict;
use locale; # ñ-ak etab.
$| = 1;

my ($l, @hitzak, $h, %Maiz);

print "Sartu hainbat hitz, lerro berean, eta espazioekin beriziak:";
$l =<>;
chomp($l);
      # banatzaileak espazioak eta tabuladoreak dira
      # Kontuz! puntuak, komak etab. hitzaren partea kontsideratuko dira
@hitzak = split(/\s+/,$l);

foreach $h (@hitzak) {
      # hitza aurretik baldin bazegoen, orduan bat gehitu
      # bestela sortu sarrera berria, agerpen bat kontatuz
  if (defined($Maiz{$h})) {
    $Maiz{$h} = $Maiz{$h} +1;
  } else {
    $Maiz{$h} = 1;
  }
}
     # ordenatu alfabetikoki
foreach $h (sort (keys (%Maiz))) {
  print "$h hitza $Maiz{$h} aldiz agertu da\n";
}
print "\n";
```

6.5.4 Hitzen maiztasuna fitxategian

fitxHitzMaiztasun.pl

```
#!/usr/bin/perl -w

use strict;
use locale; # ñ-ak etab.


my ($l,@hitzak, $h, %Maiz);

open (FI, "EH_zatia.txt");

while ($l = <FI>) {
  chomp($l);
                # banatu hitzak. Banatzaileak koma, puntua eta zuriuneak
dira
  @hitzak = split(/[\,\.\s]+/,$l);
  foreach $h (@hitzak) {
   if (defined($Maiz{$h})) {
      $Maiz{$h} = $Maiz{$h} +1;
    } else {
      $Maiz{$h} = 1;
    }
  }
}

     # Hash-eko hitzak ordenatu eta agerpen-kopurua idatzi
```

```
foreach $h ( sort (keys (%Maiz))) {
  print "$h hitza $Maiz{$h} aldiz agertu da\n";
}
print "\n";
```

6.5.5 Hitzen maiztasuna eta maiztasun erlatiboa fitxategian

fitxHitzMaiztasunerlat.pl

```
#!/usr/bin/perl -w

use strict;
use locale; # ñ-ak etab.


my ($l,@hitzak, $hitzakguztira, $h, %Maiz);

open (FI, "EH_zatia.txt");

$hitzakguztira = 0;

while ($l = <FI>) {
  chomp($l);
                # banatu hitzak. Banatzaileak koma, puntua eta zuriuneak
dira
  @hitzak = split(/[\,\.\s]+/,$l);
  foreach $h (@hitzak) {
   if (defined($Maiz{$h})) {
      $Maiz{$h} = $Maiz{$h} +1;
    } else {
      $Maiz{$h} = 1;
    };
   $hitzakguztira = $hitzakguztira + 1;
  }
}

my $maiztasuna;


my @gakoOrdenaduak = sort {$Maiz{$b} <=> $Maiz{$a}} keys(%Maiz);


foreach $h (@gakoOrdenaduak) {
  $maiztasuna = 100 * ($Maiz{$h} / $hitzakguztira) ;

  print "$h hitza \%$maiztasuna maiztasunarekin agertu da agertu da\n";
}
```

6.5.6 “garri”-z bukatzen diren hiztegiko sarrerak

EHgarriIzond.pl

```
#!/usr/bin/perl


# Sarerra "garri"-z bukatu eta kategoria "izond" duten sarrera
# guztiak lortu.

use strict;
use locale;

my (@a,$l);
open(FI, "EH_hiztek.txt");

while ($l=<FI>) {
  chomp($l);
  @a=split(/\t/,$l);
                    # bilatu garri hitz-bukaeran
  if (($a[0] =~ /.garri$/) && ($a[1] =~ /izond/)) {
    print "$a[0]\n";
  }
}
close (FI);
```

6.5.7 “garri”-z bukatzen diren hiztegiko sarrerak beren definizioekin

EHgarriIzond_gehi_def.pl

```
#!/usr/bin/perl

# Sarerra "garri"-z bukatu eta kategoria "izond" duten sarrera
# guztiak lortu, beren definizioekin.

use strict;
use locale;

my (@a,$l);
open(FI, "EH_hiztek.txt");

while ($l=<FI>) {
  chomp($l);

  @a=split(/\t/,$l);

  if (($a[0] =~ /.garri$/) && ($a[1] =~ /izond/)) {
    print "$a[0]:     ";
    print "$a[3]\n";
  }
}
close (FI);
```

6.5.8 “garri”-z bukatzen diren sarreren definizioetako hitzen maiztasuna

EHgarriMaiz.pl

```
#!/usr/bin/perl

# Definizioko hitzen maiztasuna zenbatu.
#
# Programa hau hobetzeko:
#   - Soilik "garri"-z bukatzen diren sarreren definizioak zenbatu

use strict;
use locale;

my (@a, $l, %h, $k);
open (EH, "EH_zatia.txt");

%h=();
while ($l=<EH>) {
  chomp($l);
  @a=split(/\t/,$l);

  if (($a[0] =~ /.garri$/) && ($a[1] =~ /izond/)) {
     foreach $k (split(/\W+/, $a[3])) {
       # Definizioko hitz bakoitzeko
       if (defined($h{$k})) {
         $h{$k}++;
       } else {
         $h{$k}=1;
              }
       }
   }
}
close (EH);

# Hurrengo agindua konplexu-samarra da.
# Honekin, hash baten gakoak atxikitua duten balioen arabera
# ordenatuko dira, balio handiena duena lehen agertuko delarik.

foreach $k (sort {$h{$b} <=> $h{$a}} (keys %h)) {
  print "$k\t$h{$k}\n";
}
```

6.5.9 Akatsen bila: definizioko sarrera definizioan (zirkulartasuna)

EHzirk.pl

```
#!/usr/bin/perl -w

# Hiztegian, zenbat definiziok dute definitzen ari den hitza ?

use strict;
use locale;

my (@a, $l);

open(EH, "EH_hiztek.txt");

while ($l=<EH>) {
  chomp($l);
         # banatzaileak tabuladoreak dira
  @a = split(/\t/, $l);
         # ondoren begiratu ea sarrera definizioan dagoen.
         # aurrizki moduan (\b hitz hasiera da) egotea eskatuko da
         # dena dela, honek "ama" hitza ez du aurkituko "hainbeste amen
etxetan"
         # testuan. Horretarako lematizazioa beharko genuke!
  if ($a[3] =~ /\b$a[0]/) {
    print "$l\n";
  }
}
close (EH);
```

6.5.10 Hitz bat duten lerroak idatzi aurreko lerroarekin batera

lerro1.pl

```
#!/usr/bin/perl
use strict;

my ($esp, $fIzena, $lerr, $aurLerr);

print "Zein espresio bilatu nahi duzu ?:";
$esp=<>;
chomp($esp);
print "Zein fitxategian begiratu nahi duzu ?:";
$fIzena=<>;
chomp($fIzena);

if (!open (FI, $fIzena)) {
  die "Ezin dut $fIzena fitxategia ireki !\n";
           # die -> Mezua igorri eta programatik irten

}
$aurLerr="";
while ($lerr = <FI>) {
  chomp($lerr);
  if ($lerr =~ /$esp/) {
    print "$aurLerr\n";
    print "$lerr\n";
    print "\n";
  }
  $aurLerr = $lerr;
}
```

6.5.11 Hitz bat duten lerroak idatzi aurreko eta hurrengo lerroarekin batera

lerro2.pl

```
#!/usr/bin/perl
use strict;
$| = 1;

my ($esp, $fIzena, $lerr, $aurLerr, $hurLerr);

print "Zein espresio bilatu nahi duzu ?:";
$esp=<>;
chomp($esp);
print "Zein fitxategian begiratu nahi duzu ?:";
$fIzena=<>;
chomp($fIzena);

if (!open (FI, $fIzena)) {
  die "Ezin dut $fIzena fitxategia ireki !\n";
           # die -> Mezua igorri eta programatik irten
}
$aurLerr="";
$hurLerr="";

while ($hurLerr = <FI>) {
  chomp($hurLerr);
  if ($lerr =~ /$esp/) {
    print "$aurLerr\n";
    print "$lerr\n";
    print "$hurLerr\n";
    print "\n\n\n";
  }
  $aurLerr = $lerr;
  $lerr= $hurLerr;
}
```

6.5.13 Ingelesezko testutik aditzak atera

aditzak_bilatu.pl

```
#!/usr/bin/perl -w

use strict;

my ($lerro);
open(FIN, "Hobbes_Leviathan.txt");

while ($lerro = <FIN>) {
    while ($lerro =~ /to [\w]*/g) {
      print "Aditz bat: $& \n";
    }
}

print "agur\n\n";

close (FIN);
```

6.5.14 Ingelesezko testutik aditzak ordenatuta atera

aditzak_bilatu_eta_ordenatu.pl

```
#!/usr/bin/perl -w
#programa honek testu bat ingelesez hartu eta bere "aditzak" aterako ditu
# aditz bezala "to" hitzaren ondoren datorren hitza hartuko dugu
# honek txarto egingo ditugu honakoak: "to John"
# lehenengo ordena alfabetikoan emango dira emaitzak, eta gero
maiztasunaren arabera

use strict;

my ($lerro, %aditzen_lista, %maiztasunen_lista, @giltzak, $hitza);
open(FIN, "Hobbes_Leviathan.txt");

while ($lerro = <FIN>) {
    while ($lerro =~ /to [\w]*/g) {
      $aditzen_lista{$&}++;
    }
}

close (FIN);

@giltzak = sort ( keys(%aditzen_lista));

foreach $hitza (@giltzak) {
    print "Agerpen-kopurua: $aditzen_lista{$hitza} $hitza\n";
}


print "\n\n\n Orain maiztasunaren arabera agertuko dira \n\n\n\n\n";

# Ondorengo lerro honek hasharen gakoak ardenatuko ditu, balioen arabera.
Zenbakizko ordenazioa
# burutuko du, eta balio handiak txikiak baino aurrerako egongo dira

my @gakoOrdenatuak = sort {$aditzen_lista{$b} <=> $aditzen_lista{$a}}
keys(%aditzen_lista);
my $gk;
foreach $gk (@gakoOrdenatuak) {
  print "$gk aditza $aditzen_lista{$gk} aldiz agertu da\n";
}
```

6.5.15 Ingelesezko testutik adjektiboa eta bere ondoko hitza lortu

% artifizial_bilatu.pl

```
#!/usr/bin/perl -w
use strict;

my ($lerro);
open(FIN, "Hobbes_Leviathan.txt");

while ($lerro = <FIN>) {
    while ($lerro =~ /artificial [\w]*/g) {
      print "Adibide bat: $& \n";
        print "$lerro\n"
    }
}
print "agur\n\n";
close (FIN);
```