Ειδικά Θέματα Βιοπληροφορικής

Διάλεξη 4η :

Perl (2) : Τύποι μεταβλητών : arrays,
Εντολές: foreach, sort, Πολυδιάστατοι πίνακες (2D & 3D),
Καθιερωμένη είσοδος: <STDIN>, Εντολή split, Άσκηση 2η.

Arrays

Τα arrays για τα πλαίσια αυτού του μαθήματος είναι συλλογές από scalars (αλληλουχίες χαρακτήρων ή αριθμοί). Επιφανειακά θυμίζουν τους πίνακες της C αν και η perl είναι μάλλον μονοδιάστατη σε ό,τι αφορά τα arrays (η δήλωση και χρήση πολυδιάστατων πινάκων απαιτεί τη χρήση των λεγόμενων αναφορών, references). Με τη μορφή παραδειγμάτων:

Ανάθεση τιμών και χρήση

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

@my_array_1 = (     1, 2, "three", "four", 5);
@my_array_2 = ( "one", 2,       3, "four", 5);

print @my_array_1, "\n";
print @my_array_2, "\n";
print "@my_array_1\n";
print "@my_array_1 @my_array_2\n";
```

Ανάθεση τιμών και χρήση

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

@my_array_1 = (     1, 2, "three", "four", 5);
@my_array_2 = ( "one", 2,       3, "four", 5);

print @my_array_1, "\n";
 print @my_array_2, "\n";
 print "@my_array_1\n";
 print "@my_array_1 @my_array_2\n";
```

```
12threefour5
one23four5
1 2 three four 5
1 2 three four 5 one 2 3 four 5
```

Ανάθεση τιμών και χρήση

Προσοχή στο "$" όταν αναφερόμαστε σε στοιχεία των arrays.

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

@my_array_1 = (     1, 2, "three", "four", 5);
@my_array_2 = ( "one", 2,       3, "four", 5);

print "$my_array_1[0] $my_array_1[1]\n";
print $my_array_1[0] + $my_array_2[1], "\n";
print $my_array_1[2], " and ", $my_array_1[3], "\n";
```

Ανάθεση τιμών και χρήση

Προσοχή στο "$" όταν αναφερόμαστε σε στοιχεία των arrays.

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

@my_array_1 = (     1, 2, "three", "four", 5);
@my_array_2 = ( "one", 2,       3, "four", 5);

print "$my_array_1[0] $my_array_1[1]\n";
print $my_array_1[0] + $my_array_2[1], "\n";
print $my_array_1[2], " and ", $my_array_1[3], "\n";
```

```
1 2
3
three and four
```

Ανάθεση τιμών μέσω scalars

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

$x = "5";
$y = "15";
@my_array = ( $x, $y, $x+$y, $x-$y, $x . $y);
print "Element[0] : $my_array[0]\n";
print "Element[1] : $my_array[1]\n";
print "Element[2] : $my_array[2]\n";
print "Element[3] : $my_array[3]\n";
print "Element[4] : $my_array[4]\n";
```

Ανάθεση τιμών μέσω scalars

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

$x = "5";
$y = "15";
@my_array = ( $x, $y, $x+$y, $x-$y, $x . $y);
print "Element[0] : $my_array[0]\n";
print "Element[1] : $my_array[1]\n";
print "Element[2] : $my_array[2]\n";
print "Element[3] : $my_array[3]\n";
print "Element[4] : $my_array[4]\n";
```

```
Element[0] : 5
Element[1] : 15
Element[2] : 20
Element[3] : -10
Element[4] : 515
```

Η εντολή foreach

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

$x = "5";
$y = "15";
@my_array = ( $x, $y, $x+$y, $x-$y, $x . $y);

foreach $element (@my_array)
  {
    print "Element : $element\n";
  }
```

Η εντολή foreach

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

$x = "5";
$y = "15";
@my_array = ( $x, $y, $x+$y, $x-$y, $x . $y);

foreach $element (@my_array)
   {
     print "Element : $element\n";
   }
```

```
Element : 5
Element : 15
Element : 20
Element : -10
Element : 515
```

Η εντολή foreach

```
#!/usr/bin/perl -w
$i = 0;
$x = "5";
$y = "15";
@my_array = ( $x, $y, $x+$y, $x-$y, $x . $y);


foreach $element (@my_array)
  {
    print "Element[$i] : $element\n";
    $i++;
  }
```

Η εντολή foreach

```
#!/usr/bin/perl -w
$i = 0;
$x = "5";
$y = "15";
@my_array = ( $x, $y, $x+$y, $x-$y, $x . $y);

foreach $element (@my_array)
   {
      print "Element[$i] : $element\n";
      $i++;
   }
Element[0] : 5
Element[1] : 15
Element[2] : 20
Element[3] : -10
Element[4] : 515
```

Arrays unlimited ...

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

@my_array = ( 1234, 2345 );
print "$my_array[1]\n";

$my_array[345] = "test";
print "$my_array[345]\n";
print "$my_array[1] . $my_array[345]\n";
print  $my_array[1] . $my_array[345], "\n";
```

Arrays unlimited ...

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

@my_array = ( 1234, 2345 );
print "$my_array[1]\n";

$my_array[345] = "test";
print "$my_array[345]\n";
print "$my_array[1] . $my_array[345]\n";
print  $my_array[1] . $my_array[345], "\n";
```

```
2345
test
2345 . test
2345test
```

Arrays unlimited ?

Ή πως βρίσκουμε το τρέχον μέγεθος ...

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

@my_array = ( 1234, 2345 );
$length = @my_array;
print "Current length is $length\n";

$my_array[345] = "test";
$length = @my_array;
print "Current length is $length\n";
```

Arrays unlimited ?

Ή πως βρίσκουμε το τρέχον μέγεθος ...

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

@my_array = ( 1234, 2345 );
$length = @my_array;
print "Current length is $length\n";

$my_array[345] = "test";
$length = @my_array;
print "Current length is $length\n";
```

```
Current length is 2
Current length is 346
```

sort

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

@my_array = ("John","George","Tania","nikos","Viky");
foreach $element ( @my_array )
  {
    print "$element\n";
  }

print "--- Sorted ---\n";
foreach $element ( sort @my_array )
  {
    print "$element\n";
  }
```

sort

```
John
George
Tania
nikos
Viky
--- Sorted ---
George
John
Tania
Viky
nikos
```

sort

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

@my_array = ("John","George","Tania","nikos","Viky");
foreach $element ( @my_array )
  {
    print "$element\n";
  }

@sorted_array = ( sort @my_array );

print "--- Sorted ---\n";
foreach $element ( @sorted_array )
  {
    print "$element\n";
  }
```

sort [2]

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

@my_array = ( 123, 3, 5, 234, 1643 );
foreach $element ( sort @my_array )
  {
    print "$element\n";
  }

print " --- Numerically sorted ---\n";
@num_sorted = ( sort { $a <=> $b } @my_array );
foreach $element ( @num_sorted )
  {
    print "$element\n";
  }
```

sort [2]

```
123
1643
234
3
5
 --- Numerically sorted ---
3
5
123
234
1643
```

Πολυδιάστατοι πίνακες: 2D

Προσοχή στη χρήση του "$$" ...

```
#!/usr/bin/perl -w
# Create 2D array

 $x = [];        # Do not forget this ...

 for( $i = 0 ; $i < 128 ; $i++ )
  {
    for( $j = 0 ; $j < 128 ; $j++ )
     {
       $$x[$i][$j] = sqrt( $i*$j );
     }
  }

 for( $i = 0 ; $i < 128 ; $i++ )
  {
    for( $j = 0 ; $j < 128 ; $j++ )
     {
       print "$i $j  $$x[$i][$j]\n";
     }
  }
```

Πολυδιάστατοι πίνακες: 2D

Προσοχή στη χρήση του "$$" ...

```
#!/usr/bin/perl -w
# Create 2D array

 $x = [];        # Do not forget this ...

 for( $i = 0 ; $i < 128 ; $i++ )
  {
    for( $j = 0 ; $j < 128 ; $j++ )
     {
       $$x[$i][$j] = sqrt( $i*$j );
     }
  }

 for( $i = 0 ; $i < 128 ; $i++ )
  {
    for( $j = 0 ; $j < 128 ; $j++ )
     {
       print "$i $j  $$x[$i][$j]\n";
     }
  }
```

Πολυδιάστατοι πίνακες: 3D

Προσοχή στη χρήση του "$$" ...

```
#!/usr/bin/perl -w
# Create 3D array

 $rho = [];

 for( $x = 0 ; $x < 100 ; $x++ )
   {
     for( $y = 0 ; $y < 100 ; $y++ )
       {
         for( $z = 0 ; $z < 100 ; $z++ )
           {
             $$rho[$x][$y][$z] = $x * $y * $z;
           }
       }
   }

 for ( $k = 0 ; $k < 100 ; $k++ )
   {
     print "rho[$k][$k][$k] = $$rho[$k][$k][$k]\n";
   }
```

Πολυδιάστατοι πίνακες: 3D

Προσοχή στη χρήση του "$$" ...

```
#!/usr/bin/perl -w
# Create 3D array

 $rho = [];

 for( $x = 0 ; $x < 100 ; $x++ )
   {
     for( $y = 0 ; $y < 100 ; $y++ )
       {
         for( $z = 0 ; $z < 100 ; $z++ )
           {
             $$rho[$x][$y][$z] = $x * $y * $z;
           }
       }
   }

 for ( $k = 0 ; $k < 100 ; $k++ )
   {
     print "rho[$k][$k][$k] = $$rho[$k][$k][$k]\n";
   }
```

Πολυδιάστατοι πίνακες: 3D

```
rho[0][0][0] = 0
rho[1][1][1] = 1
rho[2][2][2] = 8
rho[3][3][3] = 27
rho[4][4][4] = 64
rho[5][5][5] = 125
rho[6][6][6] = 216
rho[7][7][7] = 343
rho[8][8][8] = 512
rho[9][9][9] = 729
rho[10][10][10] = 1000
rho[11][11][11] = 1331
rho[12][12][12] = 1728
rho[13][13][13] = 2197
rho[14][14][14] = 2744
......
```

Ανάγνωση από την καθιερωμένη είσοδο

Η έκφραση "<STDIN>" διαβάζει μία γραμμή από την είσοδο:

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

$line1 = <STDIN>;
$line2 = <STDIN>;
print "First  line was : $line1";
print "Second line was : $line2";
```

Ανάγνωση από την καθιερωμένη είσοδο

Η έκφραση "<STDIN>" διαβάζει μία γραμμή από την είσοδο:

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

$line1 = <STDIN>;
$line2 = <STDIN>;
print "First  line was : $line1";
print "Second line was : $line2";
```

Που είναι το '\n' στο τέλος της print ;;;

Ανάγνωση από την καθιερωμένη είσοδο

Η "chomp" αφαιρεί το '\n' από το τέλος μίας αλληλουχίας χαρακτήρων (εάν βέβαια υπάρχει):

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

$line1 = <STDIN>;
$line2 = <STDIN>;
chomp( $line1 );
chomp( $line2 );
print "First  line was : $line1\n";
print "Second line was : $line2\n";
```

Ανάγνωση από την καθιερωμένη είσοδο

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

while ( $line = <STDIN> )
  {
    print $line;
  }
```

Ανάγνωση από την καθιερωμένη είσοδο

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

$lines = 0;
while ( <STDIN> )
  {
    $lines++;
  }

print "Read $lines lines.\n";
```

Ανάγνωση από την καθιερωμένη είσοδο

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

$index = 0;
while ( $line = <STDIN> )
  {
    chomp( $line );
    $input[$index] = $line;
    $index++;
  }

print "@input\n";
```

Ανάγνωση από την καθιερωμένη είσοδο

Χρήση arrays για ανάγνωση εισόδου

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

@all_input = <STDIN>;

print @all_input;

foreach $element ( @all_input )
  {
    print "Line from input: $element";
  }
```

Ανάγνωση από την καθιερωμένη είσοδο

Τα παραδείγματα που προηγήθηκαν δείχνουν ότι η ανάγνωση γραμμών κειμένου με την perl είναι εύκολη. Αλλά, πως μπορούμε να χειριστούμε αυτές τις γραμμές ώστε να διαβάσουμε τις 'λέξεις' ή τους αριθμούς που περιέχουν; Ένας τρόπος είναι μέσω της split την οποία θα ξαναδούμε όταν θα μιλάμε για regular expressions. Εδώ θα συζητήσουμε μόνο μία χρήση της split, με τη μορφή split(' ', $line) :

Ανάγνωση από την καθιερωμένη είσοδο

Χρήση της split

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

while ( $line = <STDIN> )
  {
    chomp $line;
    @words = split(' ', $line);
    foreach $word ( @words )
      {
        print "$word\n";
      }
  }
```

Ανάγνωση από την καθιερωμένη είσοδο

Χρήση της split

```
# ./test_split.pl
This is a test, but what isn't ?
This
is
a
test,
but
what
isn't
?
```

Ανάγνωση από την καθιερωμένη είσοδο

Χρήση της split

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

while ( $line = <STDIN> )
  {
    chomp( $line);
    @words = split(' ', $line);
    print "$words[1]\n";
  }
```

Ανάγνωση από την καθιερωμένη είσοδο

Χρήση της split

```
# ./test_split_2.pl
This is a test
is
but what isn't ?
what
```

Ανάγνωση από την καθιερωμένη είσοδο

Χρήση της split

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

$sum = 0.0;
while ( $line = <STDIN> )
  {
    @words = split(' ', $line);
    foreach $word ( @words )
      {
        $sum += $word;
      }
  }
print "Sum : $sum\n";
```

Ανάγνωση από την καθιερωμένη είσοδο

Χρήση της split

```
# ./test_sum.pl
1.23
5.67
2.45
1.23
Sum : 10.58
```

Ανάγνωση από την καθιερωμένη είσοδο

Χρήση της split

```
# ./test_sum.pl
1
2
a
Argument "a" isn't numeric in addition (+) ...
3
4
Sum : 10
```

ΑΣΚΗΣΗ

Γράψτε το πρόγραμμα που υλοποιεί τη μέθοδο Bradford χρησιμοποιώντας perl (αντί για C). Βεβαιωθείτε ότι το πρόγραμμα σας δίνει σωστά αποτελέσματα, και στη συνέχεια στείλτε το στον διδάσκοντα:

```
mail -s "My name, Pract 2" glykos@aspera.cluster.mbg.gr < myprog.pl
```