Examination questions for BINP13, 2014-10-31 (09.00 - 13.00).
Approximately 15p are required for passing the exam.

Part 1: Interprete Perl Code

Question 1 (3p): Create the following scalars:

- `$var1`: A reference to the named array `@arr`
- `$var2`: A reference to the anonymous hash consisting of the keys-values pairs: 'one'-3432 and 'Perl'-'Fun'
- `$var3`: A reference to then named hash `%tjohej`

```
$var1 = \@arr;
$var2 = {'one' => 3432, 'Perl' => 'Fun'};
$var3 = \%tjohej;
```

Question 2 (1p): What is the output of the following program?

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

my $var1 = 'Perl';
my $var2 = "$var1 is ";
my $var3 = $var2 . "really \"fun\"!";

print "$var3\n";
```

```
1 Perl is really "fun"!
```

Question 3 (2p): What is the output of the following program?

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

my $line1 = 'RGT+456 S453DFR 564++AD45QWE4 123456G 5f;;RT 45()23';
my $line2 = '---C456     ----IA23232AN-------- Q-------1454436565555';

my $cnt1 = 0;
while( $line1 =~ /[A-Z]{2,3}\s?\d+/g ) {
   $cnt1++;
}
print "$cnt1\n";

$line2 =~ s/(-|\s|A\d+A|[456])//g;
$line2 =~ tr/CQ/BP/;
print "$line2\n";
```

```
1 4
2 BINP13
```

Question 4 (2p): What is the output of the following program?

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

my @nums = (3,-3,5,8);

my $val = shift @nums;
my $i = 1;
while ($i <= $val) {
   push @nums, $i;
   $i++;
}

my $cnt = 0;
foreach my $val (@nums) {
    $cnt += $val;
}
print "$cnt\n";
```

```
1 16
```

Question 5 (2p): What is the output of the following program?

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

my $str = 'xxxxxI-A-L-GG-I-TWW-KE-A-P-GG-ER-TTA-Lyyyyyy';
$str =~ s/^[xy]+//;
$str =~ s/[xy]+$//;
$str =~ s/-A-/ /g;

my @parts = split /-G\w-|-T\w{2}-/, $str;

my $res = join '', @parts;
print "$res\n";
```

```
1 I LIKE PERL
```

Part 2: Write Perl code

Question 6 (3p): You have a nucleotide sequence defined in the scalar `$seq`. Write **Perl code** that replace all codons 'TAG', 'TAA' and 'TGA' with the character '-' and print the number of replacements done.

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

# Not actually part of the solution
my $seq = 'TCTAACCGATGGCTTCTTAGGTTAGCCCAGCATGAGCAGCGAGCTGACGGCTCGTTCCAG';

# Two step solution

# Count
my $cnt = 0;
while ( $seq =~ /TAG|TAA|TGA/g ) {
   $cnt++;
}

# Replace
$seq =~ s/(TAG|TAA|TGA)/-/g;

print "Made $cnt replacements\n";

# All in one step
my $cnt = ($seq =~ s/(TAG|TAA|TGA)/-/g);
print "Made $cnt replacements\n";
```

Question 7 (3p): The Unix command `grep` can be used to search for text in files. As an example,

```
>> grep sub parse.pl
```

will print all **lines** in the file 'parse.pl' that contains the string 'sub'.

Write a **Perl program** that works like the Unix command `grep`. The first argument to the Perl program should be the string to search for and the second argument should be the file to search in.

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

my $str = $ARGV[0];
open my $FH, '<', $ARGV[1] or die "Cannot open file $ARGV[1]. $!\n";

while (my $line = <$FH>) {
   print $line if( $line =~ /$str/ );
}
close $FH;
```

Question 8 (3p): Write **Perl code** that calculates the sum

$$\sum_{k=1}^{100} \frac{1}{k^2} = (1 + 1/4 + 1/9 + 1/16 + ... + 1/10000)$$

and prints it on the screen.

```
#! /usr/bin/perl -w
use strict;
use Math::Trig ':pi';

my $sum = 0;
for ( my $k = 1; $k <= 100; $k++ ) {
    $sum += 1/$k**2;
}
print "The sum is $sum\n";

# Another solution using map
my $sum2 = 0;
map {$sum2 += 1/$_**2} (1..100);

print "The sum is $sum2\n";

# Or the math solution :-)
print "The sum is approximately ", pi*pi/6, "\n";

```

Question 9 (3p): Write a **subroutine** that takes a reference to a hash as the argument. We can assume that all values in the supplied hash are numbers. Return a reference to an array containing all keys in the supplied hash that have values in the interval [0,10].

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

my %hh = ('Hank' => -30, 'Dolly' => 2 , 'Tanya' => 20, 'Johnny' => 9);

print "Keys that have values in [0,10]\n";
my $ref = filterKeys(\%hh);
foreach (@{$ref}) {
   print "$_\n";
}

sub filterKeys {
   my ($href) = @_;

   my @res;
   foreach my $key (keys %{$href}) {
      my $x = $href->{$key};
      push @res, $key if ($x >= 0 && $x <= 10);
   }

   return \@res;
}
```

Question 10 (3p): Write **Perl code** that reads a scalar from standard input. We can assume that this scalar contains a sequence of amino acid letters. Find the number of occurrences of the letter D in this sequence. You should solve this task using the builtin function `index` and not using regular expressions. The documentation for `index` follows here:

```
index STR,SUBSTR,POSITION
index STR,SUBSTR

   The index function searches for one string within another, but without
   the wildcard-like behavior of a full regular-expression pattern match.
   It returns the position of the first occurrence of SUBSTR in STR at or
   after POSITION. If POSITION is omitted, starts searching from the
   beginning of the string. POSITION before the beginning of the string or
   after its end is treated as if it were the beginning or the end, respectively.
   POSITION and the return value are based at zero. If the substring is not
   found, "index" returns -1.
```

```
1  #! /usr/bin/perl -w
2  use strict;
3
4  print "Give aa sequence: ";
5  my $str = <STDIN>;
6  chomp $str;
7
8  my ($off, $idx, $cnt) = (0, 0, 0);
9
10 # The argement here is not that important
11 # since I will terminate the loop with a last statement
12 while ( $idx != -1 ) {
13
14    $idx = index($str, 'D', $off);
15
16    # If $idx == -1, no more 'D' was found, then stop
17    last if ($idx == -1 );
18
19    # If we come here it means that a 'D' was found. Increase the counter
20    $cnt++;
21
22    # We also need to update the starting point of the next search
23    $off = $idx + 1;
24 }
25 print "Found $cnt occurences of 'D' in the string: \n$str\n";
```

Good Luck!

/Mattias