Skriptsprachenorientierte Programmierung -Probeklausur

Alle Aufgaben sind in der Programmiersprache Perl zu lösen.
Die üblichen Einrückregeln müssen beachtet werden.

Aufgabe 1 (20 Punkte)

a**) Schreiben Sie ein kleines Perl-Programm**, das einen Radius einliest und das Volumen einer Kugel berechnet und ausgibt (v = 4/3*pi*r^3 mit pi = 3.1415926).
Verwenden Sie use strict;

```
use strict;
print "Radius:";
my $radius = <STDIN>;
my $volumen = 4/3 * 3.1415926 *$radius**3 ;
print "Volumen: $volumen\n";
```

b) **Wie können in Perl Zeichenketten aneinander gehängt werden**?
Speichern Sie Hallo und Welt in zwei Stringvariablen ab und erzeugen Sie „Hallo Welt“ durch Stringkonkatenation und geben Sie es **dreimal** aus.

```
Punktoperator: .
my $z1 = "Hallo";
my $z2 = "Welt";
my $z3 = $z1.$z2
print "$z3"x3;
```

c) Wie können Sie die Länge einer Zeichenkette bestimmen?
Lesen Sie einen String ein und vergleichen Sie ihn mit „Hallo Welt“.

```
Funktion: length
chomp(my $text = <STDIN>);
if ($text eq "Hallo Welt") #Inhalt vergleichen
{
  print "Hallo Welt\n";
  }
if (length($text) == length("Hallo Welt")) # Länge vergleichen
{
  print "Laenge gleich\n";
}
```

d) **Schreiben Sie ein Unterprogramm**, das die Mehrwertsteuer zu einem Nettopreis berechnet und den Bruttopreis zurückliefert.

```
use strict;
sub rechnung
{
  return $_[0]*1.19;

}
```

Wie werden in Perl Argumente an ein Unterprogramm übergeben?
Rufen Sie das Unterprogramm aus d) auf.
Funktionsaufruf mit Name des Unterprogramms und Argumente in ()
Argumente stehen dem Unterprogramm im vordefinierten Array @_ zur Verfügung. Übergabe an die Subroutine mit Name der Subroutine und Klammer.

```
 #Test
my $netto = 100;
my $brutto = &rechnung($netto);
print "$brutto\n";
```

e) Geben Sie die Anweisungen in Perl an um eine **beliebige Anzahl von Zahlen von Tastatur einzulesen und aufsteigend sortiert** wieder auszugeben.

Tastatur einzulesen und aufsteigend sortiert wieder auszugeben.

```
use strict;
my @zahlen;
while (my $zahl = <STDIN>)
{
  push @zahlen, $zahl;
}
@zahlen = sort{$a <=> $b}    @zahlen;
print "@zahlen";
```

f) Ersetzen Sie durch entsprechende Perl-Anweisungen in
my $satz = “The black cat climbed the green tree”
“black” durch “white” und “climbed” durch ” jumped from”.

```
use strict;
my $satz = "The black cat climbed the green tree";
$satz =~ s/black/white/;
$satz =~ s/climbed/jumped from/;
print "$satz\n";
```

Aufgabe 2 (20 Punkte)

a) Schreiben Sie eine**Subroutine fahrenheit,** die einen Temperaturwert in Celsius als Parameter hat und den entsprechenden Wert in Fahrenheit zurückliefert. Die Formel für die Umrechnung lautet:
Fahrenheit = 9/5 * Celsiuswert + 32.

b) Legen Sie ein Array mit CelsiusTemperaturwerten an und durchlaufen Sie das Array und wandeln alle Werte mit der subroutine fahrenheit in Fahrenheit um. Geben Sie die Werte auf Bildschirm aus.

c) Verändern Sie das Programm aus b) so, dass die **Ausgabe auf Datei** erfolgt (Überschreiben). Der Dateiname soll **Temp.txt** sein. Falls sich die Datei nicht öffnen lässt, beenden Sie das Programm mit einer Fehlermeldung.

```
#!/usr/bin/perl
use strict;
sub fahrenheit
{
   return   9/5 * $_[0] + 32 ;

}
     # Datei öffnen
 if (!open OUT,">temp.txt"){
   die "Datei laesst sich nicht oeffnen $!";
 }


 my @werte=(23.5,25.7,30.8,20.6,18.9);
 foreach (@werte )
 {
   my $celsius = &fahrenheit($_);
   print "$_ : $celsius\n";
   print OUT "$celsius\n";

 }
# Schließen der Datei
 close OUT;
```

Aufgabe 3 (22 Punkte)

Verwalten Sie in einem Hash Länder und zughörige Hauptstädte, also
Deutschland, Berlin
Frankreich, Paris
Italien, Rom
u.s.w.

a) Legen Sie eine **Hash mit vier Einträgen** an.

```
my %staedte = (Deutschland=> 'Berlin' ,Frankreich=> 'Paris',Italien=> 'Rom', Spanien=> 'Madrid');
```

b) Mit welcher(n) Anweisung(en) können Sie alle Länder mit ihren Hauptstädten ausgeben?

```
my ($key, $value) ;
while (($key, $value) = each %staedte) {
  print "$key: $value\n";
  }
```

c) Schreiben Sie ein kleines Quizprogramm:

Geben Sie nacheinander alle Länder des Hash aus und lassen vom Benutzer die zugehörige Hauptstadt eingeben. Richtige Antworten werden gezählt, bei falschen Antworten wird die richtige Hauptstadt ausgegeben.
Nach einem Durchgang wird die Anzahl der richtigen Antworten ausgegeben.

```
my $count=0;
while (($key, $value) = each %staedte) {
  print "$key: ";
  chomp(my $ant = <STDIN>);
  if ($ant eq $value){
    $count++;
    }else{
       print “$value“;
  }
  print "$count richtige Antworten\n";
```

d) Geben Sie die Anweisung(en) an um einen weiteren Eintrag in den Hash aufzunehmen.
Prüfen Sie dabei, ob das Land schon vorhanden ist. Wenn ja, überprüfen Sie, ob die gleiche Stadt gespeichert ist. Wenn ja, bleibt der Eintrag unverändert, wenn nein: Fragen Sie den Benutzer, ob er den Eintrag verändern will. Will er dies, dann überschreiben Sie die Stadt.
Noch nicht vorhandene Einträge werden übernommen.

```
print "Neues Land\n";
my $land;
 chomp($land = <STDIN>);
 print "Stadt: ";
 chomp (my $stadt= <STDIN>);
 if (exists($staedte{$land}))
 {
   if ($staedte{$land} ne $stadt)  {
     print "Ueberschreiben?";
     chomp(my $z = <STDIN>);
     if ($z eq "Ja"){
          $staedte{$land} = $stadt;
   }
  }

 }else
 {  $staedte{$land} = $stadt;
}
```

Aufgabe 4 (17 Punkte)

a) Folgende Liste (Array) ist gegeben:
@Staedte = (“Paris“,“London“,“Frankfurt“,“München“,“Prag“,“Rom“);

Geben Sie für die folgenden Perl-Anweisungen an, ob hier skalarer Kontext oder Listenkontext vorliegt und kommentieren Sie kurz das Ergebnis:

$Anz = @Staedte; **skalar, Größe des Array**

print “($#Staedte)+1“; **skalar, Ausgabe höchster Index + 1 (als String)**

@kopie = @Staedte; **Liste, Kopie des Array Staedte**

$Zahl = 1 + @Staedte; **skalar, Größe des Array + 1**

b)Folgende Liste (Array) ist gegeben:
@zutaten = ("Zwiebel","Tomaten","Knoblauch","Chilli","Pfeffer","Salz",);

Erstellen Sie aus zutaten zwei Listen. Eine enthält Zwiebel und Tomaten, die andere den Rest.

```
my @teil1 = @zutaten[0..1];
my $last = $#zutaten;
my @teil2 = @zutaten[2..$last];
```

c) Vertauschen Sie mit Hilfe von slice das erste und das letzte Element von Zutaten.

```
my $last = $#zutaten;
@zutaten[0,$last] = @zutaten[$last,0];
```

d) Was ist der Unterschied von $zutaten[0] und @zutaten[0] ?
$zutaten[0] : erstes Element der Liste (skalar)
@zutaten[0]: Listenscheibe mit einem Element (Liste)

e) Entfernen Sie das erste Element von Zutaten und fügen Sie anschließend Paprika hinzu
shift @zutaten;
push @zutaten,"Paprika";

Aufgabe 5 (23 Punkte)

a) Geben Sie die regulären Ausdrücke für folgende Zeichenfolgen
Weltmeister oder Europameister am Ende einer Textzeile
/(Welt)|(Europa)meister$/

Wort Weltmeister in einer Textzeile
/\bWeltmeister\b/

Beliebig viele Sterne, gefolgt von mindestens einem a und einem c oder einem d.
/a+[cd]/ oder /**(ac)|d)+/**

- Beliebig viele, aber mindestens eine Wiederholung der drei Buchstaben abc.
/(abc)+/

- fünfmal der Inhalt der Variablen $text.
/($text){5}/

b) Ergänzen Sie in der folgenden Tabelle jede Zeile für den Match True oder False:

RegEx	Untersuchte Strings	Match?
/al.e/	Alle Voegel fliegen hoch Finale am Sonntag Walter liebt Waerme.	**false** **false** **true**
/al[lt]?e/	Alle Voegel fliegen hoch Finale am Sonntag Walter liebt Waerme.	**false** **true** **true**
/al*e/i	Alle Voegel fliegen hoch Finale am Sonntag Walter liebt Waerme.	**true** **true** **true**

c) Das Grundalphabet sei {a,b}. Konstruieren Sie reguläre Ausdrücke, die folgende Zeichenketten beschreiben:

hier mit Wortgrenzen

- Alle Strings mit mindestens einem b

/\bb+\b/

- Alle Strings, die mit einem a anfangen und mit einem b enden.

/\ba.*b\b/ oder /\ba[ab]*b\b/

Aufgabe 6 (18 Punkte)

a) Eine Bankleitzahl besteht aus drei Ziffern, wobei die erste Ziffer keine 0 sein darf, gefolgt von einem Minuszeichen. Dann kommen wieder drei Ziffern, ein weiteres Minuszeichen und zwei abschließende Ziffern.
Schematisch hat eine Bankleitzahl also folgenden Aufbau:

zzz-zzz-zz, wenn jedes z für eine Ziffer steht.

Schreiben Sie einen regulären Ausdruck, der eine Bankleitzahl beschreibt.
mit Wortgrenzen: **/\b[1-9]\d{2}-\d{3}-\d{2}\b/**

Suchen Sie in einer Texteingabe von Konsole alle Bankleitzahlen und speichern Sie sie in einer Datei „Bankleitzahl.txt“

```
use strict;
if (!open OUT,">blz.txt"){
  die "Cannot open file";
  }
  while((my $zeile =<STDIN>))
  {
    if ($zeile =~/ /\b[1-9]\d{2}-\d{3}-\d{2}\b/){
      print  OUT "$&\n";
      }
}
close OUT;
```

b) Schreiben Sie ein Perl-Programm, das in einem Text der über Tastatur eingegeben wird, das Wort Stadt durch Dorf ersetzt. Die Ersetzung soll allerdings nur stattfinden, wenn Stadt alleine steht, also nicht in zusammengesetzten Wörtern, wie Stadtpark oder Hauptstadt.

```
use strict;
while (my $zeile = <STDIN>){
  $zeile  =~ s/\bStadt\b/Dorf/g;
  print "$zeile\n";
}
```

c) Was wird hier ausgegeben?

```
my @zeichen = split(/\./, "21.12.2010");
print join(":", @zeichen);
```

21:12:2010