Fall 97 - Programming in Perl

Perl TEST - November 5, 1997

Instructor: Louis Ziantz

```
Question 1  [20 pts] : _______________

Question 2  [25 pts] : _______________

Question 3  [30 pts] : _______________

Question 4  [25 pts] : _______________

Total Score [100pts] : _______________
```

Read each question carefully and make sure to **show all of your work**.

Your name:____

Your RCS userid:___________________________

Have you ever used HTML to create a webpage? _____________

Question 1 *[20 pts]*
In the space provided below, indicate what the following snippet of Perl code prints (represent a newline by <*n*> and assume that the `-w` flag is not used so no warnings will be printed):

```
$a1 = 1;
$var1 = "15th " + "St" . "8th " . "St";
$var2 = 2;
$var3 = "N = " . 13 x $var2;
$var4 = 'camel\n';
chomp($var4);
$var5 = chop($var4);
@a1 = (1, 2, 3, 4);
@a2 = reverse(@a1);
($var6, $var7, $var8) = @a2[0,1];

print ( "$var1\n" ); # 1
print ( "$var2\n" ); # 2
print ( "$var3\n" ); # 3
print ( "$var4\n" ); # 4
print ( "$var5\n" ); # 5
print ( "$var6\n" ); # 6
print ( "$var7\n" ); # 7
print ( "$var8\n" ); # 8
print ( "$a1[$a2[$#a1]]\n" ); # 9
print ( "${a1}[$a2[-2]]\n" ); # 10
```

1) ______________________________

2) ______________________________

3) ______________________________

4) ______________________________

5) ______________________________

6) ______________________________

7) ______________________________

8) ______________________________

9) ______________________________

10) ______________________________

Question 2 *[25 pts]*
Read a series of numeric class grades from standard input into an array (assume one grade per line). You may assume all data entered is valid. After the values have been input, determine the number of A's, B's, C's, and F's based on the following scale:

Score	Grade
90–100	A
80–89	B
70–79	C
0–69	F

You should also compute the average grade for the class. Print the class average, the number of A's, the number of B's, the number of C's, and the number of F's. Each piece of information should be printed on a separate line with appropriate text labels, e.g.,

```
Class average : 75<newline>
Number of A's: 20<newline>
(and so forth)
```

Question 3 *[30 pts]*
Given the following arrays:

```
@day_nums = (1, 2, 3, 4, 5, 6, 7);
@day_strings = ('mon.', 'tue.', 'wed.', 'thur.', 'fri.', 'sat.', 'sun.');
```

(a) Assign the same values as in the first assignment above to `@day_nums` using the list constructor operator (`..`).

(b) Assign the same values as in the second assignment above to `@day_strings` without explicitly using either single or double quotes around the strings.

Given the following hash:

```
%days = (1, 'mon.', 2, 'tue.', 3, 'wed.', 4, 'thur.',
         5, 'fri.', 6, 'sat.', 7, 'sun.');
```

(c) Use the arrays `@day_nums` and `@day_strings` to create a hash `%days1` having the same key-value pairs as `%days`.

(d) Create a hash `%days2` having the same key-value pairs as `%days` using the `=>` operator and literals.

(e) Use the `foreach` control structure along with any built-in functions needed to print the values of the hash `%days` on separate lines.

(f) Iterate through the hash `%days` using the `each` function and print the values of the hash on separate lines with the first letter of each day capitalized.

Question 4 *[25 pts]*
Use the diamond operator to write a Perl script that functions as a simple `more` command. If a user types `more.pl file1`, the script should print the first `$max_num_lines` (say 20) lines of file1 and then prompt the user for input from standard input as follows:

```
(lines from file1)
Action [l or p]:
```

The script should continue prompting for and reading the `$action` until an `l` followed by a return or a `p` followed by a return is typed. If an `l` is entered, the script should print the next line of the file and prompt again for an action. If a `p` is input, the script should print the next `$max_num_lines` lines of file1 before prompting for an action again. This process (prompting and printing either the next line or the next `$max_num_lines` lines) should continue until the file's contents are exhausted.