Programming in Perl
CSCI-2962
Final Exam

Rules and Information:

1. TURN OFF ALL CELULAR PHONES AND PAGERS!
2. Make sure you are seated at least one empty seat away from any other student.
3. Write your first name, last name, and RCS ID (not your RIN!) on the top of the *back* of EACH page
4. Clear your desks/tables of all Calculators, PDAs, and Laptops
5. You may use any book (Camel, Llama, or other) and any typed or hand-written notes.
6. You have until 6:00pm EST to complete this exam.
7. If you have a question, raise your hand. Justin and/or I will come to you.
8. When you have completed the exam bring it to the table in the front of the room
9. If you submitted a final exam question for extra credit, see me **before** turning in your exam to receive your bonus points (if any).
10. Solutions to this exam will be posted on the course website just after 6pm today.
11. When the Exams have been graded, an announcement will be posted to the course website informing you where and when you may pick up your exams.
12. Good luck with the remainder of your classes and exams.
13. If you are interested in being an undergraduate Teaching Assistant for this course next semester, please email me at lallip@cs.rpi.edu for more information or to apply.

Regular Expressions

1) 1 point:
List all the strings the following regexp will match:
```
/prob|l|r|n|ate/
```

'prob', 'l', 'r', 'n', or 'ate'

2) Write an expression to… (2 points each)

a) match a standard telephone number (ex: 1-518-276-8677)
/\d-\d{3}-\d{3}-\d{4}/

b) remove a duplicate word from a string. (ex: take out the second 'good' from "Have a good good day")
s/(\w+) \1/$1/;

c) correct accidental usage of the Caps Lock key. (ex: Change "hELLO, i AM pAUL" to "Hello, I am Paul".)
tr/a-zA-Z/A-Za-z/;

3) 3 points:
Write a regular expression that matches "Rensselaer" if and only if it is followed by "Polytechnic Institute". (ONLY actually match Rensselaer, not anything else)

/Rensselaer (?=Polytechnic Institute)/

4) 4 points:
Currently, the following pattern match will succeed if a digit is contained anywhere in the input. Modify it to restrict input to a single non-negative digit only. (ie, If user enters anything other than 0, 1, 2, 3, 4, 5, 6, 7, 8, or 9, it fails).
```
chomp($input = <STDIN>);
print "You entered a digit\n." if ($input =~ /\d/);
```

$input =~ /^\d$/

5) 3 points:
Write a piece of code to determine if a given string (`$string`) begins and ends with the same character.

$string =~ /^(.).*\1$/

6) 5 points:
Why is this a bad method for ending a user's input to a script?

```
while ($input = <STDIN>){
  if ($input =~ /quit/){
    print "Quitting now, goodbye!\n";
    last;
  } else {
    # do stuff
  }
}
```

If the user enters something like "quite good!", the program will end.

7) 8 points:
Write a script that takes the name of a file from the command line. The script will simply output all lines of the file with the exception of the fact that when it encounters a line in the form ‘<command>’ it will run the command in the shell and output the results of the command. For example, if the input is

```
    The current time is
    <date>
    Good-bye.
```

The output will be

```
    The current time is
    Wed Dec 2 16:00:00 EST 2001
    Good-bye.
```

```
$file = $argv[0];
open FILE, $file or die "...";
while ($line = <FILE>){
        if ($line =~ /^<(.*)>$/){
                print `$1`;
        } else {
                print $line;
        }
}
```

Subroutines

1) 5 points
When the function funct1 is called, what are the contents of ...?
$_[0] _"Perl"__ $_[3] __3.5____ $_[4] ___1_______

```
@arr1 = (1, 5, 10);
@arr2 = ("Perl", "is", "great");
sub funct1(@arr2, 3.5, @arr1, "hello world", 0);
```

2) 5 points
A certain function is designed to return an entire array called @list if the function is called in list context. If it's called in scalar context, it returns the last element of @list. Write the `return` line of this function
return wantarray ? @list : $list[$#list];

3) 10 points
Assume function record() is declared as follows:

```
sub record(\@\$@);
```

Which of the following function calls will NOT generate errors? (check all that apply)

- ☐ `record(@foo, "hello world", 5, 4, 3);`
- [✓] `record(@help, $I_need, "somebody", "not just anybody");`
- ☐ `record((1, 2, 10), $bar, @world);`
- [✓] `record(@Dayo, $Me_Say, @Dayo);`

Random Bits of Perl

1) 5 points
QUESTION TOSSED OUT IN CLASS. EVERYONE GETS 5 POINTS

2) 5 points
What is the output of this code?

```
@nums = (1..10);
map {$_ *= 2} @nums;
print "@nums\n";
```

2 4 6 8 10 12 14 16 18 20

Objects

1) 10 points

Write a constructor for a class `Person` that has data-members Name and Age. Your constructor should set Name to the null string, and Age to 0.

```
package Person;
sub new {
	my $ref = {Name => “”, Age => 0};
	bless ($ref, Person);
	return $ref;
}
```

CGI

1) 2 points

What are the first two lines of any Perl CGI script?

```
#!/usr/local/bin/perl –w
use CGI qw(:standard);
```

2) 3 points

What HTML code does this print out?

```
print start_html('My CGI program');
```

```
<html><head><title>My CGI program</title></head><body>
```

3) 5 points

Write a piece of CGI script to determine if the user pushed a button named 'Add' (labeled 'Add item') or a button named 'Delete' (labeled 'Delete item'). In each case, just print out a message saying which button was pressed. You do not have to write the code to actually generate these buttons.

```
if (param('Add')){
	print "Add button was pressed\n";
} elsif (param('Delete')){
	print "Delete button was pressed\n";
}
```

4) 10 points

Write a short script that uses the CGI.pm package to print out 2 HTML pages. The first page will display 3 checkboxes and a submit button as follows:

☐ `alpha`

☐ `beta`

☐ `gamma`

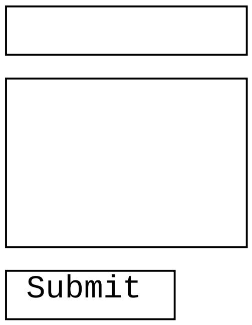
[`Submit`]

After the submit button is pushed, your script will simply print out a list of all the boxes that were checked (if no boxes checked, print “No boxes were checked”)

```
#!/usr/local/bin/perl –w
use CGI qw(:standard);

print header;
print start_html;
if (!param()){
        print startform;
        print "<input type='checkbox' name='checks' value='alpha'>Alpha\n",
                "<input type='checkbox name='checks' value='beta'>Beta\n",
                "<input type='checkbox' name='checks' value='gamma'>Gamma\n",
                "<input type='submit' name='Submit' value='Submit'>\n";
        print endform;
} else {
        if param('checks){
                @checks = param('checks');
                foreach (@checks){
                        print "$_<br>\n";
                }
        } else {
                print "No boxes were checked\n";
        }
}
```

5) 10 points
Write a CGI Script to print a textfield, a text area, and a submit button, like so:



When the submit button is pressed, your script creates a new text file (in whatever directory the script was run from) with the name entered in the textfield, and with the contents entered in the text area. Do not concern yourself with any error checking. So, if the text field contains "readme.txt" and the text area contains "Hello World", your script creates a new file called 'readme.txt', prints "Hello World" to this file, and closes the file.

```
#!/usr/local/bin/perl –w
use CGI qw(:standard);

print header;
print start_html;
if (!param()){
        print startform;
        print "<input type='text' name='Name'>\n",
                "<textarea name='Contents'></textarea>\n",
                "<input type='submit' name='Submit' value='Submit'>\n";
        print endform;
} else {
        $name = param('Name');
        $contents = param('Contents');
        open FILE, ">$name";
        print FILE $contents;
        close FILE;
}
```

Bonus Question.

1) 5 points

Write a Perl program to prompt the user for a number and then prompt for a string. Print the string that number of times on separate lines. Do not use any looping structures.

```
#!/usr/local/bin/perl –w
print "Enter a number:\n";
chomp ($num = <STDIN>);
print "Enter a string:\n";
$string = <STDIN>;
print $string x $num;
```