

Programming in Perl

Midterm Exam review

The exam will cover everything we've done so far

The exam will have 4 or 5 sections:

- True/False – simple statements about Perl
- What's wrong here? – You will be given a chunk of code that does not do what it's supposed to. Explain why, and fix it.
- Short answer – 2- or 3-sentence answers to questions about Perl.
- What's the output? – given a chunk of code, what does it print? (similar to class handouts)
- Code it – write a couple simple programs.

There will be one bonus question, worth either 5 or 10 points

Sample questions:

- The first member of the array @ARGV is always the name of the Perl script. True or False?
 - False. \$0 holds the program name. \$ARGV[0] is the first command line argument.
- If there are no command-line arguments, the <> operator acts on STDIN. True or False?
 - True
- This code fragment does not work. Why? Fix it:

```
$a = 30;
$b = 100;
if ($a lt $b){
    print "$a is less than $b\n";
} else {
    print "$a is greater than or equal to $b\n";
}
```

- lt is the string-comparison less than operator. Replace it with <

- This code fragment also does not work. Why? Fix it:

```
if ($a == 10){
    print "$a is equal to ten\n";
} else if ($a > 10){
    print "$a is greater than ten\n";
} else {
    print "$a is less than ten\n";
}
```

```
}
```

- o else if is a syntax error. either place { } around everything following the first else, or replace else if with elsif.
- Briefly list and describe the three main types of Perl variables
 - o Scalar – any single value, integers, floats, characters, strings
 - o Array – collection of scalars, indexed numerically starting at 0
 - o Hash – collection of scalars, indexed at any value whatsoever
- Briefly explain when/how variables \$1, \$2, \$3, etc get set.
 - o when a pattern match succeeds and part of it is captured in parentheses, whatever gets matched in the parentheses gets saved in those variables.

- What is the output of this code?

```
@animals = ("cat", "dog", "bird", "lizard");  
print "These are four animals: " . @animals . "\n";
```

- o These are four animals: 4<newline>
- What is the output of this code?

```
foreach ('a'..'z'){  
    next if /[aeiou]/;  
    print;  
    print "\n";  
}
```

- o b c d f g h j k l m n p q r s t v w x y z
- o (all separated by newlines)

- In the following chunk of code, assume the user enters “I love Perl!” and “Perl is my friend” at the prompt. What is the output of the code?

```
while (<>){  
    $i=0;  
    foreach (split / /){  
        print "$i:$_ ";  
        $i++;  
    }  
    print "\n";  
}
```

- o 0:I 1:love 2:Perl!
- o 0:Perl 1:is 2:my 3:friend
- Write ONE regular expression that will replace all instances of < with < and all instances of > with >

- o `$string =~ s/(<|>)/$1 eq '<' ? '<';' : '>';/ge;`
- Write a program that reads an unknown number of words from the command line, and prints the words out in the opposite order from which they were entered. Change all upper-case letters to lower-case, and all lower-case letters to upper-case. Do not use the `reverse` keyword

Example: if the user enters

```
prog1.pl Hello, My name is Paul
```

your program will output:

```
pAUL IS NAME mY hELLO,
```

```
#!/usr/bin/perl -w
for ($i = $#ARGV; $i >= 0; $i--){
    $ARGV[$i] =~ tr/a-zA-Z/A-Za-z/;
    print "$ARGV[$i] ";
}
```