Exercises

• If you’re new to programming or Perl, read and run all the examples!

• Advance preparatory reading:
  • Chapter 2 of the textbook
  • we’ll be using Perl’s regular expression (regex) engine for this
Perl Wierdness

• Variable types:
  • Every variable type has its own namespace.
  • This means that $foo and @foo are two different variables.
  • It also means that $foo[1] is a part of @foo, not a part of $foo.
    
    This may seem a bit weird, but that's okay, because it is weird.
Perl Wierdness

Reading Perl code (perldata) aloud:
- *Larry Wall was a linguist ...*
- The ''$' symbol works semantically like the English word "the" in that it indicates a single value is expected.

- Entire arrays (and slices of arrays and hashes) are denoted by '@', which works much like the word "these" or "those" does in English, in that it indicates multiple values are expected.

Also:
- Entire hashes are denoted by '%' ... (no translation)
- In addition, subroutines are named with an initial '&', though this is optional when unambiguous, just as the word "do" is often redundant in English.
Perl Arrays

• **Arrays:**
  • Idea: list of scalars
  • *like a simple ordered list (cf. Python lists)*
  • access by index: 0,1,2,...
  • negative indices count from the end -2..-1

• A note on output:
  • print @a zeroonetwothreefour
  • print “@a” zero one two three four

controlled by system variable $’
default value: a space
Perl Arrays

• Literal:
  • @ARRAY = ( ... , ... , ...)  (round brackets; comma separator)

• Access:
  • $ARRAY[INDEX]  (zero-indexed; negative indices ok; slices ok)

• Index of last element:
  • $#array  (a scalar)

• Coercion
  • @ARRAY = number of elements in scalar context

• Built-in arrays:
  • @ARGV  (command line arguments)
  • @_  (sub(routine) arguments)
Perl Arrays

There’s a special array called @ARGV

• Getting user input into your program:

There are a couple of special arrays too, such as @ARGV (the command line arguments to your script) and @ARGV (the arguments passed to a subroutine). These are documented in perlvar.

```perl
@ARGV
```

• @ARGV

The array @ARGV contains the command-line arguments intended for the script. $#ARGV is generally the number of arguments minus one, because $ARGV[0] is the first argument, not the program’s command name itself. See $0 for the command name.

```perl
C programming language:   int argc, char *argv[]
Shell script: $0  $1  $2  ..  ($#)
```
Perl Arrays

• Built-in functions:
  • sort @ARRAY;
  • reverse @ARRAY;
  • push @ARRAY, $ELEMENT;
  • pop @ARRAY;
  • shift @ARRAY;
  • unshift @ARRAY, $ELEMENT,
  • splice @ARRAY, $OFFSET, $LENGTH, $ELEMENT
  • Note: $ELEMENT above can also be @ARRAY
Perl Arrays

- by default sort works according to the ASCII chart, character by character

<table>
<thead>
<tr>
<th>Dec</th>
<th>Hex</th>
<th>Oct</th>
<th>Char</th>
<th>Dec</th>
<th>Hex</th>
<th>Oct</th>
<th>Char</th>
<th>Dec</th>
<th>Hex</th>
<th>Oct</th>
<th>Char</th>
<th>Dec</th>
<th>Hex</th>
<th>Oct</th>
<th>Char</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000</td>
<td>000</td>
<td>null</td>
<td>32</td>
<td>020</td>
<td>040</td>
<td>Space</td>
<td>64</td>
<td>040</td>
<td>060</td>
<td>Space</td>
<td>96</td>
<td>060</td>
<td>080</td>
<td>Space</td>
</tr>
<tr>
<td>1</td>
<td>001</td>
<td>001</td>
<td>start of heading</td>
<td>33</td>
<td>021</td>
<td>041</td>
<td>#</td>
<td>65</td>
<td>041</td>
<td>061</td>
<td>#</td>
<td>97</td>
<td>061</td>
<td>081</td>
<td>#</td>
</tr>
<tr>
<td>2</td>
<td>002</td>
<td>002</td>
<td>start of text</td>
<td>34</td>
<td>022</td>
<td>042</td>
<td>$</td>
<td>66</td>
<td>042</td>
<td>062</td>
<td>$</td>
<td>98</td>
<td>062</td>
<td>082</td>
<td>$</td>
</tr>
<tr>
<td>3</td>
<td>003</td>
<td>003</td>
<td>end of text</td>
<td>35</td>
<td>023</td>
<td>043</td>
<td>%</td>
<td>67</td>
<td>043</td>
<td>063</td>
<td>%</td>
<td>99</td>
<td>063</td>
<td>083</td>
<td>%</td>
</tr>
<tr>
<td>4</td>
<td>004</td>
<td>004</td>
<td>end of transmission</td>
<td>36</td>
<td>024</td>
<td>044</td>
<td>^</td>
<td>68</td>
<td>044</td>
<td>064</td>
<td>^</td>
<td>100</td>
<td>064</td>
<td>084</td>
<td>^</td>
</tr>
<tr>
<td>5</td>
<td>005</td>
<td>005</td>
<td>line feed</td>
<td>37</td>
<td>025</td>
<td>045</td>
<td>&amp;</td>
<td>69</td>
<td>045</td>
<td>065</td>
<td>&amp;</td>
<td>101</td>
<td>065</td>
<td>085</td>
<td>&amp;</td>
</tr>
<tr>
<td>6</td>
<td>006</td>
<td>006</td>
<td>carriage return</td>
<td>38</td>
<td>026</td>
<td>046</td>
<td>*</td>
<td>70</td>
<td>046</td>
<td>066</td>
<td>*</td>
<td>102</td>
<td>066</td>
<td>086</td>
<td>*</td>
</tr>
<tr>
<td>7</td>
<td>007</td>
<td>007</td>
<td>line feed (newline)</td>
<td>39</td>
<td>027</td>
<td>047</td>
<td>(</td>
<td>71</td>
<td>047</td>
<td>067</td>
<td>(</td>
<td>103</td>
<td>067</td>
<td>087</td>
<td>(</td>
</tr>
<tr>
<td>8</td>
<td>010</td>
<td>010</td>
<td>horizontal tab</td>
<td>40</td>
<td>028</td>
<td>048</td>
<td>)</td>
<td>72</td>
<td>048</td>
<td>068</td>
<td>)</td>
<td>104</td>
<td>068</td>
<td>088</td>
<td>)</td>
</tr>
<tr>
<td>9</td>
<td>011</td>
<td>011</td>
<td>vertical tab</td>
<td>41</td>
<td>029</td>
<td>049</td>
<td>#</td>
<td>73</td>
<td>049</td>
<td>069</td>
<td>#</td>
<td>105</td>
<td>069</td>
<td>089</td>
<td>#</td>
</tr>
<tr>
<td>10</td>
<td>A</td>
<td>012</td>
<td>(NL line feed, newline)</td>
<td>42</td>
<td>02A</td>
<td>052</td>
<td>-</td>
<td>74</td>
<td>052</td>
<td>06A</td>
<td>-</td>
<td>106</td>
<td>06A</td>
<td>08A</td>
<td>-</td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td>013</td>
<td>(vertical tab)</td>
<td>43</td>
<td>02B</td>
<td>053</td>
<td>+</td>
<td>75</td>
<td>053</td>
<td>06B</td>
<td>+</td>
<td>107</td>
<td>06B</td>
<td>08B</td>
<td>+</td>
</tr>
<tr>
<td>12</td>
<td>C</td>
<td>014</td>
<td>(SP blank, new page)</td>
<td>44</td>
<td>02C</td>
<td>054</td>
<td>-</td>
<td>76</td>
<td>054</td>
<td>06C</td>
<td>-</td>
<td>108</td>
<td>06C</td>
<td>08C</td>
<td>-</td>
</tr>
<tr>
<td>13</td>
<td>D</td>
<td>015</td>
<td>(line feed)</td>
<td>45</td>
<td>02D</td>
<td>055</td>
<td>-</td>
<td>77</td>
<td>055</td>
<td>06D</td>
<td>-</td>
<td>109</td>
<td>06D</td>
<td>08D</td>
<td>-</td>
</tr>
<tr>
<td>14</td>
<td>E</td>
<td>016</td>
<td>(shift out)</td>
<td>46</td>
<td>02E</td>
<td>056</td>
<td>-</td>
<td>78</td>
<td>056</td>
<td>06E</td>
<td>-</td>
<td>110</td>
<td>06E</td>
<td>08E</td>
<td>-</td>
</tr>
<tr>
<td>15</td>
<td>F</td>
<td>017</td>
<td>(shift in)</td>
<td>47</td>
<td>02F</td>
<td>057</td>
<td>-</td>
<td>79</td>
<td>057</td>
<td>06F</td>
<td>-</td>
<td>111</td>
<td>06F</td>
<td>08F</td>
<td>-</td>
</tr>
<tr>
<td>16</td>
<td>L002</td>
<td>020</td>
<td>data link escape</td>
<td>40</td>
<td>020</td>
<td>030</td>
<td>0</td>
<td>80</td>
<td>030</td>
<td>060</td>
<td>0</td>
<td>112</td>
<td>060</td>
<td>080</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>L021</td>
<td>021</td>
<td>(device control 1)</td>
<td>41</td>
<td>021</td>
<td>031</td>
<td>0</td>
<td>81</td>
<td>031</td>
<td>061</td>
<td>0</td>
<td>113</td>
<td>061</td>
<td>081</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>L022</td>
<td>022</td>
<td>(device control 2)</td>
<td>42</td>
<td>022</td>
<td>032</td>
<td>0</td>
<td>82</td>
<td>032</td>
<td>062</td>
<td>0</td>
<td>114</td>
<td>062</td>
<td>082</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>L023</td>
<td>023</td>
<td>(device control 3)</td>
<td>43</td>
<td>023</td>
<td>033</td>
<td>0</td>
<td>83</td>
<td>033</td>
<td>063</td>
<td>0</td>
<td>115</td>
<td>063</td>
<td>083</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>L024</td>
<td>024</td>
<td>(device control 4)</td>
<td>44</td>
<td>024</td>
<td>034</td>
<td>0</td>
<td>84</td>
<td>034</td>
<td>064</td>
<td>0</td>
<td>116</td>
<td>064</td>
<td>084</td>
<td>0</td>
</tr>
<tr>
<td>21</td>
<td>L025</td>
<td>025</td>
<td>(negative acknowledgement)</td>
<td>45</td>
<td>025</td>
<td>035</td>
<td>0</td>
<td>85</td>
<td>035</td>
<td>065</td>
<td>0</td>
<td>117</td>
<td>065</td>
<td>085</td>
<td>0</td>
</tr>
<tr>
<td>22</td>
<td>L026</td>
<td>026</td>
<td>(synchroneous idle)</td>
<td>46</td>
<td>026</td>
<td>036</td>
<td>0</td>
<td>86</td>
<td>036</td>
<td>066</td>
<td>0</td>
<td>118</td>
<td>066</td>
<td>086</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>L027</td>
<td>027</td>
<td>(end of transmission block)</td>
<td>47</td>
<td>027</td>
<td>037</td>
<td>0</td>
<td>87</td>
<td>037</td>
<td>067</td>
<td>0</td>
<td>119</td>
<td>067</td>
<td>087</td>
<td>0</td>
</tr>
<tr>
<td>24</td>
<td>L028</td>
<td>028</td>
<td>(cancel)</td>
<td>48</td>
<td>028</td>
<td>038</td>
<td>0</td>
<td>88</td>
<td>038</td>
<td>068</td>
<td>0</td>
<td>120</td>
<td>068</td>
<td>088</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>L031</td>
<td>031</td>
<td>(end of medium)</td>
<td>49</td>
<td>029</td>
<td>039</td>
<td>0</td>
<td>89</td>
<td>039</td>
<td>069</td>
<td>0</td>
<td>121</td>
<td>069</td>
<td>089</td>
<td>0</td>
</tr>
<tr>
<td>26</td>
<td>L032</td>
<td>032</td>
<td>(character substitute)</td>
<td>50</td>
<td>030</td>
<td>040</td>
<td>0</td>
<td>90</td>
<td>040</td>
<td>070</td>
<td>0</td>
<td>122</td>
<td>070</td>
<td>090</td>
<td>0</td>
</tr>
<tr>
<td>27</td>
<td>L033</td>
<td>033</td>
<td>(escapes)</td>
<td>51</td>
<td>031</td>
<td>041</td>
<td>0</td>
<td>91</td>
<td>041</td>
<td>071</td>
<td>0</td>
<td>123</td>
<td>071</td>
<td>091</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>L034</td>
<td>034</td>
<td>(file separator)</td>
<td>52</td>
<td>032</td>
<td>042</td>
<td>0</td>
<td>92</td>
<td>042</td>
<td>072</td>
<td>0</td>
<td>124</td>
<td>072</td>
<td>092</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>L035</td>
<td>035</td>
<td>(group separator)</td>
<td>53</td>
<td>033</td>
<td>043</td>
<td>0</td>
<td>93</td>
<td>043</td>
<td>073</td>
<td>0</td>
<td>125</td>
<td>073</td>
<td>093</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>L036</td>
<td>036</td>
<td>(record separator)</td>
<td>54</td>
<td>034</td>
<td>044</td>
<td>0</td>
<td>94</td>
<td>044</td>
<td>074</td>
<td>0</td>
<td>126</td>
<td>074</td>
<td>094</td>
<td>0</td>
</tr>
<tr>
<td>31</td>
<td>L037</td>
<td>037</td>
<td>(unit separator)</td>
<td>55</td>
<td>035</td>
<td>045</td>
<td>0</td>
<td>95</td>
<td>045</td>
<td>075</td>
<td>0</td>
<td>127</td>
<td>075</td>
<td>095</td>
<td>0</td>
</tr>
</tbody>
</table>

Source: www.asciitables.com
Perl Arrays

Numeric sort?

- **Idea:** to pass an inline comparison function as parameter...
- **Note:** fc (fold case)

use feature 'fc';

```perl
# sort lexically
@articles = sort @files;

# same thing, but with explicit sort routine
@articles = sort {$a cmp $b} @files;

# now case-insensitively
@articles = sort {fc($a) cmp fc($b)} @files;

# same thing in reversed order
@articles = sort {$b cmp $a} @files;

# sort numerically ascending
@articles = sort {$a <=> $b} @files;

# sort numerically descending
@articles = sort {$b <=> $a} @files;
```

*global variables $a and $b*

Binary "cmp" returns -1, 0, or 1 depending on whether the left argument is stringwise less than, equal to, or greater than the right argument.

Binary "<=" returns -1, 0, or 1 depending on whether the left argument is numerically less than, equal to, or greater than the right argument.
Perl Hashes

```
my %fruit_color = ("apple", "red", "banana", "yellow");
```

You can use whitespace and the `=>` operator to lay them out more nicely:

```
my %fruit_color = {
    apple => "red",
    banana => "yellow",
};
```

To get at hash elements:

```
$fruit_color("apple");  # gives "red"
```

You can get at lists of keys and values with `keys()` and `values()`.

```
my @fruits = keys %fruit_colors;
my @colors = values %fruit_colors;
```

Hashes have no particular internal order, though you can sort the keys and loop through them. Just like special scalars and arrays, there are also special hashes. The most well known of these is `%ENV` which contains environment variables. Read all about it (and other special variables) in `perldata`.
Perl Hashes

• Notes on arrays and hashes
  • **arrays** are indexed using integers
  • **hashes** are like arrays with user-defined indexing
    (**aka** associative array or hash table or dictionary (**Python**))

• initialization
  (use list notation (or shortcut): **round brackets and commas**)
  • `@a = ("zero", "one", "two", "three", "four");`
  • `%h = ("zero", 0, "one", 1, "two", 2, "three", 3, "four", 4);` (**key/value pairs**)

• access to individual elements
  (**square brackets vs. curly braces**)
  • `$a[1]”one”`
  • `$h{zero} 0`

Shortcut:

| 1. | qw(foo bar baz) |
|    | is semantically equivalent to the list: |
| 1. | "foo", "bar", "baz" |
Perl Hashes

• Notes on arrays and hashes
  • output
    • `print @a` zeroonetwothreefour
    • `print "@a"` zero one two three four
    • `print %h` three3one1zero0two2four4
      (note: different order)
    • `print "%h"` %h (literal, no interpolation done)
More on Hash tables

• Example:

```perl
use strict;
my %fruitColor = ("apple", "red", "banana", "yellow", "kiwi", "green");
foreach my $i (keys %fruitColor) {
    print "$i => $fruitColor{$i}\n"
}
```

• Output:

```
dhcp-10-142-144-54:Desktop sandiaway$ perl test.perl
banana => yellow
apple => red
kiwi => green
```

Unique key constraint:

```perl
my %fruitColor = ("apple", "red", "banana", "yellow", "kiwi", "green", "apple", "green");
```