Perl Cheatsheet

Overview

Perl is a high-level, general-purpose programming language that is commonly used for web development, system administration, and network programming. It was created by Larry Wall in 1987 and is known for its flexibility, expressiveness, and powerful regular expression support.

Variables

Perl variables are prefixed with a sigil that indicates the type of variable:

- `$` for scalar variables (single values)
- `@` for array variables (ordered lists of values)
- `%` for hash variables (unordered collections of key-value pairs)

```
# Scalar variable
my $name = "Alice";

# Array variable
my @numbers = (1, 2, 3, 4, 5);

# Hash variable
my %ages = ("Alice" => 30, "Bob" => 40, "Charlie" => 50);
```

Functions

Perl has a large number of built-in functions for common tasks such as string manipulation, file I/O, and regular expressions. Functions are called using parentheses after the function name.

```
# String manipulation
my $name = "Alice";
my $length = length($name);  # Returns 5

# File I/O
open(my $fh, "<", "input.txt") or die "Cannot open file: $!";
my @lines = <$fh>;
close($fh);

# Regular expressions
my $text = "hello world";
if ($text =~ /world/) {
    print "Found 'world' in text\n";
}
```

Loops

Perl has several types of loops, including `for`, `foreach`, `while`, and `until`. The `for` and `foreach` loops are used to iterate over arrays or lists, while the `while` and `until` loops are used to repeat a block of code while a condition is true or false, respectively.

```perl
# For loop
for (my $i = 0; $i < 10; $i++) {
    print "$i\n";
}

# Foreach loop
my @numbers = (1, 2, 3, 4, 5);
foreach my $num (@numbers) {
    print "$num\n";
}

# While loop
my $i = 0;
while ($i < 10) {
    print "$i\n";
    $i++;
}

# Until loop
my $i = 0;
until ($i >= 10) {
    print "$i\n";
    $i++;
}
```

Conditionals

Perl has several conditional statements, including `if`, `elsif`, `else`, `unless`, and `given/when`. These statements are used to control the flow of a program based on certain conditions.

```perl
# If statement
my $age = 30;
if ($age >= 18) {
    print "You are an adult\n";
}

# If-else statement
my $age = 15;
if ($age >= 18) {
    print "You are an adult\n";
} else {
    print "You are a minor\n";
}

# If-elsif-else statement
my $age = 25;
if ($age < 18) {
    print "You are a minor\n";
} elsif ($age < 65) {
    print "You are an adult\n";
```

```
} else {
    print "You are a senior\n";
}

# Unless statement
my $age = 15;
unless ($age >= 18) {
    print "You are a minor\n";
}

# Given/when statement
my $fruit = "apple";
given ($fruit) {
    when ("apple") {
        print "It's an apple\n";
    }
    when ("banana") {
        print "It's a banana\n";
    }
    default {
        print "It's something else\n";
    }
}
```

File Manipulation

Perl provides several functions for manipulating files, including `open`, `close`, `read`, `write`, and `rename`.

```
# Open file for reading
open(my $fh, "<", "input.txt") or die "Cannot open file: $!";

# Read file contents
my @lines = <$fh>;

# Close file
close($fh);

# Open file for writing
open(my $fh, ">", "output.txt") or die "Cannot open file: $!";

# Write to file
print $fh "Hello, world!\n";

# Close file
close($fh);

# Rename file
rename("input.txt", "input.old") or die "Cannot rename file: $!";
```

Resources

- Perl documentation
- Perl programming at Perl.com
- PerlMonks (community forum)
- Learn Perl in Y minutes (quick reference guide)