

Tham Khảo Nhanh Perl

Biến, regex, file I/O, references, modules cơ bản

Cơ Bản

Hello World

```
#!/usr/bin/perl
use strict;
use warnings;
print "Hello, World!\n";
say "Hello, World!"; # with use feature 'say';
```

Chạy Perl

```
perl script.pl # run a file
perl -e 'print "hi\n"' # run inline
perl -ne 'print' file # process file line by line
```

Chú Thích & Tài Liệu

```
# single-line comment
=pod
Multi-line POD documentation
=cut
```

Biến

Sigils

\$scalar	Giá trị đơn (chuỗi, số, tham chiếu)
@array	Danh sách có thứ tự các scalars
%hash	Cặp key-value
\$array[0]	Truy cập phần tử mảng đơn (ngữ cảnh scalar)
\$hash{key}	Truy cập giá trị hash đơn (ngữ cảnh scalar)

Biến Scalar

```
my $name = "Perl"; # string
my $version = 5.40; # number
my $count = 42; # integer
my $undef; # undefined (undef)
my $combined = "$name v$version"; # interpolation
```

Ngữ Cảnh

```
my @arr = (1, 2, 3);
my $count = @arr; # scalar context: 3
my @copy = @arr; # list context: (1, 2, 3)
my $len = scalar @arr; # force scalar context
```

Biến Đặc Biệt

\$_	Biến mặc định (topic)
@_	Tham số của subroutine
\$_!	Thông báo lỗi hệ thống
\$_@	Lỗi từ eval
\$0	Tên chương trình
@ARGV	Tham số dòng lệnh
%ENV	Biến môi trường

Toán Tử

Toán Tử So Sánh

==, !=, <, >, <=, >=	So sánh số
eq, ne, lt, gt, le, ge	So sánh chuỗi
<=>	Spaceship số (trả về -1, 0, 1)
cmp	Spaceship chuỗi
==~	Khớp / ràng buộc regex
!~	Không khớp regex

Toán Tử Chuỗi

```
my $full = "Hello . " . " . "World"; # concatenation
my $line = "-" x 40; # repetition
my $len = length($full); # 11
```

Toán Tử Logic

&& / and	Logic AND (ưu tiên thấp: and)
 / or	Logic OR (ưu tiên thấp: or)
//	Defined-or (trả về trái nếu đã định nghĩa)
! / not	Logic NOT
? :	Điều kiện ba ngôi

Luồng Điều Khiển

Điều Kiện

```
if ($x > 0) { print "positive\n"; }
elsif ($x == 0) { print "zero\n"; }
else { print "negative\n"; }
print "yes\n" if $condition; # postfix if
print "no\n" unless $condition; # postfix unless
```

Vòng Lặp

```
for my $i (0..9) { print "$i\n"; }
foreach my $item (@array) { print "$item\n"; }
while ($line = <STDIN>) { chomp $line; }
until ($done) { last if check(); }
```

Điều Khiển Vòng Lặp

next	Bỏ qua lần lặp tiếp theo (như continue)
last	Thoát vòng lặp (như break)
redo	Khởi động lại lần lặp hiện tại
next LABEL	Bỏ qua lần lặp tiếp theo của vòng lặp có nhãn
last LABEL	Thoát vòng lặp có nhãn

Given / When

```
use feature 'switch';
given ($status) {
    when ("ok") { say "success"; }
    when ("error") { say "failed"; }
    default { say "unknown"; }
}
```

Subroutines

Subroutine Cơ Bản

```
sub greet {
    my ($name) = @_;
    return "Hello, $name!";
}
my $msg = greet("Alice");
```

Tham Số Mặc Định & Có Tên

```
sub connect {
    my (%opts) = @_;
    my $host = $opts{host} // "localhost";
    my $port = $opts{port} // 5432;
    return "$host:$port";
}
connect(host => "db.example.com", port => 3306);
```

Tham Chiếu Subroutine

```
my $double = sub { return $_[0] * 2; };
print $double->(5); # 10
my @sorted = sort { $a <=> $b } @nums;
```

Prototypes & Signatures

```
use feature 'signatures';
sub add($a, $b) { return $a + $b; }
sub greet($name, $greeting = "Hello") {
    return "$greeting, $name!";
}
```

Regex

Khớp

```
if ($str =~ /pattern/) { print "matched\n"; }
if ($str =~ /(d+)/) { print "number: $1\n"; }
my @matches = ($str =~ /(w+)/g); # all matches
```

Thay Thế

```
$str =~ s/old/new/; # first occurrence
$str =~ s/old/new/g; # global (all occurrences)
$str =~ s/^\s+|\s+$//g; # trim whitespace
(my $clean = $str) =~ s/\W//g; # non-destructive copy
```

Modifiers

/i	Không phân biệt hoa/thường
/g	Toàn cục (tất cả khớp)
/m	Nhiều dòng (^ và \$ khớp ranh giới dòng)
/s	Một dòng (. khớp ký tự xuống dòng)
/x	Mở rộng (cho phép khoảng trắng và chú thích)

Pattern Phổ Biến

\d, \D	Chữ số / không phải chữ số
\w, \W	Ký tự từ / không phải ký tự từ
\s, \S	Khoảng trắng / không phải khoảng trắng
\b	Ranh giới từ
(? ...)	Nhóm không bắt
(?<name> ...)	Bắt có tên (truy cập qua \${name})

File I/O

Mở & Đọc

```
open(my $fh, '<', 'data.txt') or die "Cannot open: $!";
while (my $line = <$fh>) {
    chomp $line;
    print "$line\n";
}
close($fh);
```

Ghi & Thêm Vào

```
open(my $fh, '>', 'out.txt') or die "Cannot open: $!";
print $fh "Hello\n";
close($fh);
open(my $fh, '>>', 'log.txt') or die "Cannot open: $!";
print $fh "entry\n";
close($fh);
```

Đọc Toàn Bộ File

```
use File::Slurp;
my $content = read_file('data.txt');
my @lines = read_file('data.txt', chomp => 1);
```

Kiểm Tra File

-e \$path	File tồn tại
-f \$path	Là file thường
-d \$path	Là thư mục
-r / -w / -x	Đọc được / ghi được / thực thi được
-s \$path	Kích thước file tính bằng byte (0 nếu rỗng)
-z \$path	File có kích thước bằng 0

Tham Khảo Nhanh Perl

Arrays & Hashes

Mảng

```
my @arr = (1, 2, 3, 4, 5);
push @arr, 6;          # append
my $last = pop @arr;  # remove last
my $first = shift @arr; # remove first
unshift @arr, 0;      # prepend
my @slice = @arr[1..3]; # slice
```

Hàm Mảng

scalar @arr	Số phần tử
push / pop	Thêm/xóa từ cuối
shift / unshift	Xóa/thêm từ đầu
splice(@a, 2, 1)	Xóa 1 phần tử tại chỉ số 2
sort @arr	Sắp xếp theo thứ tự bảng chữ cái
reverse @arr	Đảo ngược thứ tự
grep { /pat/ } @arr	Lọc theo pattern
map { \$_ * 2 } @arr	Biến đổi mỗi phần tử
join(' ', @arr)	Nối thành chuỗi

Hashes

```
my %user = (name => "Alice", age => 30);
$user{email} = "a@b.com"; # add pair
delete $user{age};        # remove pair
my @keys = keys %user;
my @vals = values %user;
```

Lặp Qua Hash

```
while (my ($k, $v) = each %hash) {
    print "$k => $v\n";
}
for my $key (sort keys %hash) {
    print "$key: $hash{$key}\n";
}
```

References

Tạo References

```
my $scalar_ref = \$name;
my $array_ref  = \@arr;
my $hash_ref   = \%hash;
my $anon_arr   = [1, 2, 3]; # anonymous array ref
my $anon_hash  = {a => 1, b => 2}; # anonymous hash ref
```

Dereference

```
print $$scalar_ref; # dereference scalar
print $array_ref->[0]; # arrow notation
print $hash_ref->{key}; # arrow notation
my @copy = @$array_ref; # dereference to array
my %copy = %$hash_ref; # dereference to hash
```

Cấu Trúc Dữ Liệu Phức Tạp

```
my @users = (
    { name => "Alice", age => 30 },
    { name => "Bob",   age => 25 },
);
print $users[0]->{name}; # "Alice"
```

Hàm ref()

ref(\$r) eq 'SCALAR'	Tham chiếu đến scalar
ref(\$r) eq 'ARRAY'	Tham chiếu đến array
ref(\$r) eq 'HASH'	Tham chiếu đến hash
ref(\$r) eq 'CODE'	Tham chiếu đến subroutine

Modules

Dùng Modules

```
use strict;
use warnings;
use List::Util qw(sum max min);
use File::Basename;
use Cwd qw(abs_path);
```

Tạo Module

```
# MyModule.pm
package MyModule;
use Exporter 'import';
our @EXPORT_OK = qw(helper);
sub helper { return "help"; }
1; # module must return true
```

Core Modules Phổ Biến

List::Util	sum, max, min, reduce, any, all
File::Basename	basename, dirname, fileparse
File::Path	make_path, remove_tree
Getopt::Long	Phân tích tham số dòng lệnh
JSON	encode_json, decode_json
LWP::Simple	get(\$url) — HTTP client đơn giản
Data::Dumper	Debug dump cấu trúc dữ liệu
Carp	croak, confess — thông báo lỗi tốt hơn

CPAN

```
cpan install Module::Name # install from CPAN
cpanm Module::Name       # cpanminus (faster)
perldoc Module::Name     # read module docs
```