Elements of Programming in Perl

<H16-2/3>

VARIABLES and CONTROL STRUCTURES

Josep F. Abril

jabril@imim.es

Basic Data Structures in Perl

$** ⇨ **$calar values

```
$dnaseq = "ATGGGTA";
```

@ ⇨ **@rrays of scalars**

```
@nucleotides = ( "A", "C", "T", "G" );
```

$nucleotides[1]

@nucleotides[1,2]

$#nucleotides

0 1 2 3

% ⇨ **%ash variables**

```
%stop_codons = ( "ochre" => "UAA",
                 "amber" => "UAG",
                  "opal" => "UGA");
```

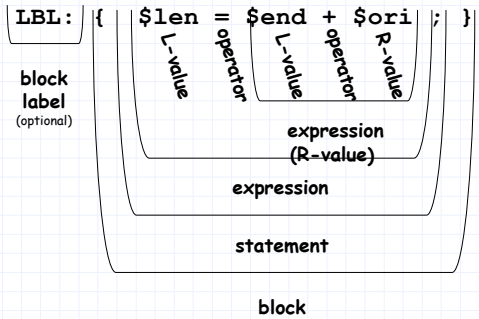
$stop_codons{ochre}

**** ⇨ **References**

```
$sequence = \$dnaseq;
```

$$sequence

Code Blocks



Control Structures

STATEMENT MODIFIERS

```
STATEMENT if EXPR;
STATEMENT unless EXPR;
STATEMENT while EXPR;      do BLOCK while EXPR
STATEMENT until EXPR;      do BLOCK until EXPR
STATEMENT foreach LIST;
```

COMPOUND STATEMENTS

```
if (EXPR) BLOCK
if (EXPR) BLOCK else BLOCK
if (EXPR) BLOCK elsif (EXPR) BLOCK ... else BLOCK
LABEL while (EXPR) BLOCK
LABEL while (EXPR) BLOCK continue BLOCK
LABEL for (EXPR; EXPR; EXPR) BLOCK
LABEL foreach VAR (LIST) BLOCK
LABEL foreach VAR (LIST) BLOCK continue BLOCK
LABEL BLOCK continue BLOCK
```

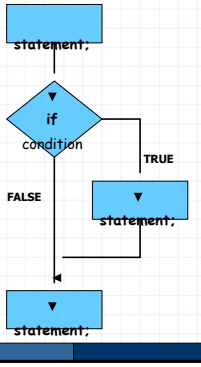

True/False in Boolean Context

FALSE values:

The integer value	`0`
The floating point value	`0.0`
The string	`'0'`
The empty (null) string	`''`
The empty list	`()`
The special undefined value	`undef`
in numeric context	`undef == 0`
in string context	`undef eq ''`

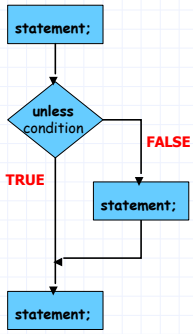
All other values are TRUE

Control Structures: if



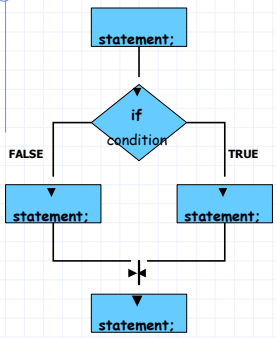

```
if (condition) {
    statements;
};

(condition) && do {
    statements;
};
```

Control Structures: **unless**



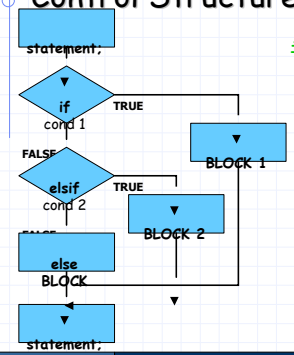

```
unless (condition) {
    statements;
};

(condition) || do {
    statements;
};
```


Control Structures: **if/else**



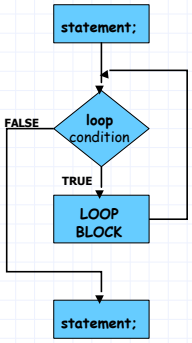

```
if (condition) {
    statements;
} else {
    statements;
};

(condition) ? yes : no
```


Control Structures: **if/elsif/else**



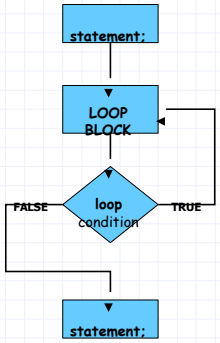

```
if (condition 1) {
    statements;
} elsif (condition 2) {
    statements;
} elsif (condition 3) {
    statements;
    ...
} elsif (condition N) {
    statements;
} else {
    statements;
};
```

Control Structures: **loops**



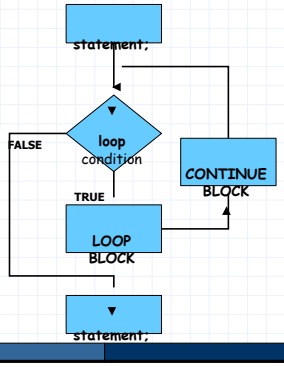

```
while (condition) {
    statements;
};

until (condition) {
    statements;
};

for (initialization;
     condition;
     iteration) {
    statements;
};

foreach (list) {
    statements;
};
```


Control Structures: **do loops**




```
do {
    statements;
} while (condition);

do {
    statements;
} until (condition);
```


Control Structures: **continue block**




```
while (condition) {
    statements;
} continue {
    statements;
};

foreach (list) {
    statements;
} continue {
    statements;
};
```

Loop Control

```
# ... Previous code
$counter = 0;
while ($counter < 100) {
        # ... First statement set
        ($counter == 50) && ($counter++, redo);
        # ... Second statement set
        ($counter == 25) && do {
                #... more statements;
                next;
        };
        # ... Third statement set
        ($counter == 99) && last;
        # ... Fourth statement set
        $counter = $counter + 1;
} continue {
        # ... Continue statement set
        print STDOUT $counter, "\n";
};
# ... More code
```


Control Structures: **switch**

```
SWITCH: {
        (condition_1) && do {
                statements; last SWITCH;
        };
        (condition_2) && do {
                statements; last SWITCH;
        };
        ...
        (condition_N-1) && do {
                statements; last SWITCH;
        };
        statements; # condition_N (default)
};
```