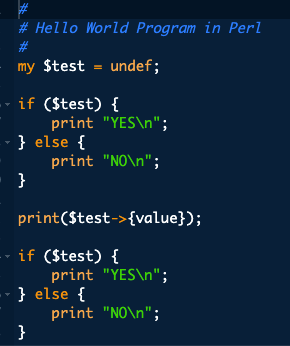
# General

<https://masonbook.houseabsolute.com/book/chapter-4.html>

* If you try to access an undefined perl object, perl will actually create the object so that next time you test it, it will be defined!!

  
  
The 1st print will be :NO but the 2nd will be YES because the   
$test->{value} actually creates the test object!!

Explaining the following code:  
$result =   
 PI::Membership::Service->cancel(name => $name)

It is calling (invoking) the subroutine PI::Membership::Service::cancel with three arguments.

1. "PI::Membership::Service"
2. "name"
3. $name

Given normal naming conventions, this is calling a subroutine called cancel in the package PI::Membership::Service, defined in a file named PI/Membership/Service.pm somewhere along your @INC path (there are many abnormal naming conventions, however, so there is no guarantee you will find such a file). And if the PI::Membership::Service package (class) inherits from one or more other packages, the cancel subroutine might actually be defined in one of those packages.

# Data Types

## Boolean

* The following are considered false:
  + undef
  + 0
  + ""
  + "0"
  + Any blessed objects that overload conversion to boolean to return false
  + Empty lists and empty hashes
* All other values are true.
  + Note: even the string ‘false’ evaluates to true!
* Negation of a true value by "!" or "not" returns a special false value. When evaluated as a string it is treated as '', but as a number, it is treated as 0.

## Strings

String concatenation:

* Using the ‘.’ Operator:

$name = checkbook';

$filename = '/tmp/' . $name . '.tmp';

# $filename now contains "/tmp/checkbook.tmp"

## Hash

Hash/dictionary is defined by the % symbol:

my %myHash = ();

my %days\_in\_summer = ( "July" => 31, "August" => 31, "September" => 30 );

#usage:

print $days\_in\_summer{"September"}; # 30, of course.