CERTLABZ CERTIFICATION BOARD

Certified Perl Programmer (CPRP)

Ten Sample Questions

These ten questions are written at the difficulty of the live exam. Six carry a figure, a data table, a configuration or a code listing; four are plain scenario questions, as on the live paper. Every one is a scenario a working Perl programmer would meet, and none of them appears in the live question bank.

The live exam is multistage adaptive: how you score on one stage selects the difficulty of the next, so no two candidates sit the same paper. It is a performance-based exam: each form can carry up to 5 to 12 performance-based questions alongside the multiple-choice questions, which means you build, sequence or complete something rather than only choosing from a list. Answers and full explanations follow each question here; the live exam does not disclose item-level answers.

Question 1

Domain 1: Perl Fundamentals and Syntax

CONFIG

Emma Wilson runs this short Perl script and checks the output.

```
my $x = 5;
my $y = 3;
print $x + $y;
```

Listing 1.1 Adding two scalars.

What does the script print?

A. 53

B. 8

C. undef

D. Runtime error

Correct answer: B

Perl evaluates the numeric addition of 5 and 3, so print outputs 8. The plus operator forces numeric context and both scalars hold numbers, so no string joining or error occurs.

Question 2

Domain 2: Scalars, Arrays and Hashes

CONFIG

James Carter assigns an array to a scalar variable and prints it.

```
my @nums = (4, 8, 15, 16);
my $count = @nums;
print $count;
```

Listing 2.1 Array in scalar context.

What is printed?

- A The whole list
- B Nothing at all
- C 4
- D The last value

Correct answer: C
Assigning an array to a scalar puts the array in scalar context, which yields its element count. The array holds four elements, so $count becomes 4 and print shows 4. The list itself is not stored in the scalar.

Question 3

Domain 3: Control Flow and Subroutines

CONFIG

Olivia Davis defines a subroutine and calls it with one argument.

```
sub square {
  my ($n) = @_;
  return $n * $n;
}
print square(6);
```

Listing 3.1 A squaring subroutine.

What does the call print?

- A 36
- B 12
- C 66
- D undef

Correct answer: A
The subroutine reads its single argument from @_, multiplies it by itself, and returns the result. Calling square with 6 returns 36, which print then displays. The other values do not match the multiplication.

Question 4

Domain 4: Regular Expressions and Text Processing

CONFIG

Liam Anderson applies a substitution to a scalar and prints it.

```
my $s = "color";
$s =~ s/color/colour/;
print $s;
```

Listing 4.1 A single substitution.

What does the script print?

A color
B colorcolour
C Nothing prints
D colour

Correct answer: D
The substitution operator finds color and replaces it with colour in place, changing $s. Printing the variable then shows colour once, since the match happens a single time without the global flag.

Question 5

Domain 5: References, Modules and the Standard Library

DATA

Ava Thompson maps each backslash reference to what it produces, with one entry missing.

Code	Refers to
\@arr	Array reference
\%h	(to choose)
\$s	Scalar reference
\&sub	Code reference

Table 5.1 Backslash references.

What does the code \%h create?

A A brand new hash copy
B A flattened key list
C An array reference
D A hash reference

Correct answer: D
The backslash operator takes a reference to the variable that follows it, so a backslash before a hash produces a hash reference. It does not copy the data or flatten it into a list, and the sigil stays a percent sign for the hash.

Question 6

Domain 6: File IO, Error Handling and Best Practices

CONFIG

David Brown opens a file for reading with a guard clause.

```
open(my $fh, "<", "log.txt")
  or die "Cannot open";
my $line = <$fh>;
close($fh);
```

Listing 6.1 Opening a file safely.

What does the or die clause guard against?

- (A) A syntax error at compile
- (B) A failed open call
- (C) A missing newline char
- (D) A slow disk read

Correct answer: B
The open call can fail when a file is missing or unreadable, and it returns false in that case. The or die guard stops the program with a message when the open call fails, so the code never reads from an unopened handle.

Question 7 Domain 1: Perl Fundamentals and Syntax

A developer runs a script where a typo in a variable name silently creates a new variable and causes a subtle bug. Which pragma catches this at compile time?

- (A) use warnings
- (B) use strict
- (C) use feature
- (D) use lib

Correct answer: B
The strict pragma requires variables to be declared, so a mistyped name fails to compile instead of silently creating a new variable. Warnings report many issues at run time, but strict is what catches the undeclared name early.

Question 8 Domain 2: Scalars, Arrays and Hashes

A programmer needs to store unique keys mapped to values, such as usernames mapped to email addresses. Which Perl data type fits best?

- (A) A scalar
- (B) An array
- (C) A hash
- (D) A code reference

Correct answer: C
A hash stores unique keys mapped to values, which fits a lookup from usernames to email addresses. An array keeps ordered items by position and a scalar holds a single value, so neither models keyed lookups as well.

Question 9 Domain 4: Regular Expressions and Text Processing

A log parser must pull the year out of many date strings and reuse it later in the program. Which regex feature remembers part of a match for reuse?

A. Anchors
B. The global flag
C. Character classes
D. Capture groups

Correct answer: D
Capture groups use parentheses to remember part of a match, making the year available afterward through capture variables. Anchors and character classes shape where and what matches, and the global flag repeats the match, but none of them stores a captured piece.

Question 10 Domain 5: References, Modules and the Standard Library

A team wants a well tested date parsing module rather than writing their own from scratch. Where should they look first for reusable Perl modules?

A. CPAN
B. The local cache
C. A random blog post
D. The system PATH

Correct answer: A
CPAN is the Comprehensive Perl Archive Network, the main home for reusable and well tested Perl modules. A team should search there first rather than writing a date parser from scratch, since a maintained module already solves the problem.

Published by the CertLabz Certification Board, a division of CertLabz LLC. Companion to objectives document CPRP-001. https://certlabz.com/certifications/certified-perl-programmer.html