By OnlineInterviewQuestions.com

Perl Interview Questions and Answers

PERL is the acronym for Practical Extraction and Reporting Language. It was developed by Larry Wall in 1987. It is a free open source language that supports object-oriented language like C++. PERL is a high-level, dynamic programming language licensed under General Public License (GNU) and is quite simple to learn as its syntax is similar to C.

The first version of PERL was 1.0 which was developed in December 1987 when Larry Wall was working as a developer in Unisys. Gradually, PERL 2 was developed in 1988 as a better version of the previous one, and PERL 3 was developed in 1989 which showed added support for binary data streams. PERL is a sensitive programming language and works with several other markup languages. The nature of PERL is procedural and supports object-oriented programming.

Read Latest Interview Questions and answers on Perl Programming

The market has a huge potential in the field of PERL programming. More and more avenues are being opened as a PERL programmer and if you want to excel in your career as a PERL programmer, then cracking the PERL interview is a must. We've listed important PERL interview questions that will make you ready for your next interview. Prepare for them and be ready to ace your interview.

Q1. List all the features of Perl programming.

Practical extraction and reporting language (PERL) is an open-source software that is used to develop web-based applications even though libraries are available to perform web server applications and networking components. The features of PERL programming are:

- PERL works with other markup languages like HTML, XML, etc.
- It enjoys the best features from languages such as C, awk, sh, sed, etc.
- The third-party databases like Oracle, Postgres, MySQL, etc. are all supported by the PERL database integration interface.
- PERL is a Y2K complaint that supports Unicode.
- PERL is quite flexible and easily readable.

Q2. What are the benefits of PERL in using it as a web-based application?

The benefits of using PERL programming are:

- PERL is known as the duct tape of the internet.
- PERL can accelerate the processing by up to 2000% when embedded into the web servers.
- Due to the rapid development cycle and text manipulation features, PERL is one of the most popular web

programming languages.
- PERL can handle encrypted web data and e-commerce transactions. The popular e-commerce site Amazon is built on PERL.
- Due to its simple syntax, PERL is very easy to comprehend.
- All versions of PERL do automatic data-typing and dynamically allocate memories to programmes and free it for use when not needed.

Q3. Talk about the applications of PERL.

PERL has myriad and varied applications. PERL is used to write Common Gateway Interface scripts and large projects like cPanel, Slash, RT, TWikietc. It is often used to write high-traffic websites like Ticketmaster, LiveJournal, DuckDuckGo, Slashdot, etc. PERL is also used by the suppliers of software to simplify packaging and maintenance of software build and deployment scripts. It can be made portable across Windows and Unix. In addition to this, PERL is also used as a glue language, and GUI may be developed using PERL.

Q4. What are different data types that Perl supports. Elaborate on them.

PERL has three basic data types. They are:

- **Scalar data type:** These are simple variables that are either a number, string or a reference. Scalar data types start with a dollar sign($). Scalar values are by default undefined. A scalar value is interpreted as TRUE in the Boolean sense if it not a null string.
- **Arrays of scalar:** These are the ordered list of scalars that can be accessed by a numeric index and starts with a 0. They start with "at" sign (@). In simpler words, array stores the list of scalar values.
- **Hashes of scalars:** Hashes are also known as associative arrays and are preceded by a percentage sign (%). Hashes are unordered sets of value pairs that can be accessed using the keys as subscripts. It stores associative arrays that use a key-value as an index instead of numerical indexes. It is a third major data type after scalars and arrays.

Q5. There are few arguments that are frequently used in Perl. What are they and what significance do they hold?

- -w : this argument highlights the warning.
- -c : this only helps in the compilation and not run.
- -d : it is used for debugging.
- -e : this argument helps in execution.
- -wd: it is a combination derived from –w and –d. we can use other combinations as well.
- -a : it automatically splits the group of input files.
- -T : it switches on the taint mode.
- -d:module : in this argument, the script is compiled and the control is transferred to the specified module.

Q6. Which functions in PERL allows you to include a module file. State their differences.

The functions that allow us to include a module file or a module is the “use” and “require” function. The differences between the two are:

1. The “use” function is used for loading the module at the compile time whereas the “require” function is used for loading the module at the runtime.
2. “Use” calls the import function of modules inbuilt whereas “require” calls the import module separately.
3. The “use” function is used only for the modules and that too for including .pm type file. The “require” function is used for both the libraries and modules.
4. In “use,” we are not required to specify the file extension whereas in “require,” it is mandatory to specify the file extension.

Q7. What are the various Perl data types based on the context?

Perl data types are treated differently based on the context in which they are accessed. They are-

- **Scalar Context:** It accesses data items as scalar values. Scalar context can be further divided into string context, numeric context and don’t-care context. Scalar context simply returns the whatever kind of scalar value they want to and let PERL convert numbers to strings in string context and strings to numbers in numeric context. Sometimes, the scalar contexts don’t care whether a string, number or a reference is returned, so no conversion happens in that case.
- **List Context:** It treats the lists and hashes as atomic objects. Assignment to an array or a hash evaluates the right-hand side in the list context.
- **Boolean Context-** It is a place where an expression is evaluated to check whether it’s true or false. It is “true” when then the scalar value is not the null string. Boolean doesn’t let any conversions to happen since it is a don’t-care context.
- **Void Context:** This context neither cares for the type of the return value nor does it want a return value. It is no different from an ordinary scalar context from the standpoint of how functions work.
- **Interpolative Context:** This context only happens inside a quote or things that act like a quote.

Q8. What is the use of –w, -t and strict in PERL?

-w gives us the warnings about the possible interpretation errors in the script.

–t is used for switching on taint checking. It checks the origin of the variables where outside variables are not allowed in subshell executions and system calls.

Strict calls the strict pragma and is used to check on the usage of variables, references. It is also used to force a check on the definition.

Q9. What are the different types of Perl operators?

Perl language supports several operator types. The important ones are-

- **Arithmetic operators:** These include addition(+), subtraction(-), multiplication(*), division(/),

modulus(%), etc.

- **Equality operators** also known as relational operators, these include == (equal to), != (not equal to), > (greater than), < (less than), etc.
- **Assignment operators** These operators assign specific values and include =, +=, -+, *=, /=, etc.
- **Logical operators** these include &&, or, ||, not, etc.
- **Quote-like operators** In this, {} depicts any kind of delimiters. These include q{}, qq{}, qx{}, etc.

Q10. Explain the meaning of subroutine.

Subroutines accept an argument, perform the necessary operation and return the value. It is a named block of code, and often the term subroutine and function are used interchangeably. Subroutine takes the PROTOTYPE as the prototype of arguments and ATTRIBUTES as the attributes of the argument. Its syntax is: sub NAME or sub NAME PROTOTYPE ATTRIBUTES when the prototypes and attributes are optional.

Q11. Explain the meaning of Perl one-liner.

Perl one-liners are one line command programs that are used for success of any operation. They may include more than one Perl statements, and one advantage to using it is that the program can be typed and executed from the command line instantly. Example:
The #run program, but with warnings

Perl –w my_file

The #run program under debugger

Perl –d my_file

Q12. Explain Lists and iValue.

Lists are special types of arrays that hold a series of values. The list can be explicitly generated by the user, or it can be a value returned by a function. Users can generate it using parenthesis and comma to separate the values. If you want to store the result of any expression, then iValues are used. These are scalar values which are found on the left-side of the expression. It represents the data space in the memory.

Q13. Explain grooving and shortening of arrays and splicing of arrays.

- **Grooving and shortening of arrays:** This can be done by giving a non-existent index to which Perl will automatically adjust the array size as needed.
- **Splicing of arrays:** Instead of just extracting another array, splicing just copies and replaces elements from the array using the position specified in the splice function. Syntax: ARRAY, OFFSET, LENGTH, LIST.

Q14. Explain Goto label, Goto name, and Goto expr.

- **Goto label:** at this label, execution stops at the current point and resumes at the point where the label is specified. You cannot jump to a point inside a subroutine using Goto Label.
- **Goto name:** it replaces the subroutine that is currently executing with a call to a particular subroutine instead. It automatically calls a different subroutine dynamically selects alternative routines.
- **Goto expr:** it is an extension of Goto label.

Q15. Explain chomp, chop, CPAN, TK.

- **Chomp:** It removes the last character from the expression. It may remove each element of the list if it matches the value $/. It is considered to be safer than chop as it removes only when there is a match.
- **Chop-** It removes last character and each element.
- **CPAN-** Comprehensive Perl Archive Network(CPAN) is a large collection of Perl software.

Q16. Elaborate on Perl bite-wise operators.

Perl bit-wise operators work on bits and perform bit-by-bit operations. It includes-

- **AND(&) :** this operator compares two bits and copies the bit to result if it exists in both operands.
- **OR(|) :** it compares two bits and copies the bit if it exists in either of the operands.
- **XOR(^) :** it copies the bit if it is there in on operand and not other.
- **~ :** this complement operator is uniary.

Q17. what are the two ways to get private values inside a subroutine?

Ans. Local operator: this operator can operate on global variables.

My operator: this operator is used to define or create a new variable. Variables created by this operator are always private.

Q18. Explain the meaning of closure in Perl.

The closure is a block of code that is used to capture the environment where it is defined, and it captures lexical variables that the block consists of.

Q19. Enlist the advantages of using C over Perl.

C has more development tools, and it also executes faster than Perl. In C, you don’t have to hide your code if you don’t want others to use them. However, in case of Perl, you have to hide your Perl code to prevent others from using them.

Q20. How interpreter is used in Perl?

The interpreter compiles the Perl program internally into a parse tree. Any word after a profound symbol will be ignored. Once compiled, the interpreter will execute it immediately.