Julia Interview Questions and Answers

1: What makes Julia a better choice over other programming languages?

Julia's Language is better than several other languages due to many reasons. The very first reason is it's a high-level language that is capable to address any needs programmers have. The vast support available makes sure that error-free outcomes can be generated in a very reliable manner. In addition to this, the final outputs of this language are simple to test and thus trust. Also, it is compatible with almost every Operating system. There are no strict upper limits on its use and this is exactly what makes it one of the best options to consider.

2: Does Julia support web applications also?

Yes, this language vast support for the same. Almost every kind of web application can be developed with this language. The biggest benefit is a very large number of operators that can be deployed for this purpose in a very reliable manner. Even if the applications need customized touch, it is possible for the users to keep up the pace. Also, there are already a very large number of web applications that are based on Julia currently being active.

3: Why some programmers avoid global variables while using Julia? What is the better alternative according to you?

Although the global variables are good enough to be trusted, the biggest issue with them is their value and types both changes frequently. Of course, this leads to code optimization problems. Later these issues make it very difficult for the users to test the code and make it OS independent. Thus, the better option is to use the local variables. Julia supports almost every local variable and lets you come with similar quality even if all the variables used are local than global.

4: Although Julia is a high-level language, give one reason due to which even beginners can handle tasks with it.

Well, the fact is Julia has been equipped with a very large number of compiler techniques. This makes it possible for the programmers to deploy various techniques to keep up the pace simply. Many tasks that are actually difficult can be made simple with this approach.

5: What are the good things you have personally noticed about the Julia Language?

- Julia can be considered for developing applications that are performance intensive
- The outputs can be made compatible with all the platforms
- Julia enables programmers to integrate application developed with other platforms in it
- It is a flexible programming language

6: What are the major applications of Julia where it is widely preferred?

Julia is widely preferred in both numerical computing and scientific computing mainly because of its performance. It can produce similar outcomes with shortcodes and thus make sure of the best outcomes. A lot of tasks related to both scientific and numerical computing can easily be handled.

7: How you will measure the performance in Julia and what are the problems that can declare their presence while doing so?

One of the best things about Julia is it has a very simple tool available for this and the same is @time. With this tool, performance can simply be measured accurately and without compromising with anything. It is possible for the programmers to even compile this tool for measuring quantities that are not possible with common methods.

8: Does Julia's compiler is similar to Python?

No, it is actually different and sometimes this makes many programmers think Julia is a very complex language which is not true.

9: Name one approach that simply enables you to run the Julia code at a faster speed

Well, by avoiding the use of global variables, it has been seen that programmers can make sure of speed. However, it is not always necessary to work with all the programmers. It actually depends on the experience and programming skills of a professional that how he/she can enhance the code speed.

10: In Julia, what is the restriction on a code which is benchmarked?

In Julia, if a code is benchmarked or is very critical, the same should be assigned within a function. If not, there are various compatibility problems that

can declare their presence. Also, benchmarked codes sometimes need to be kept in a separate container. It is possible to call them with a single command from the same.

11: In Julia, do you find any other function or tool similar to the @time? Which one you prefer and why?

toc() and tic() are the other functions that one can put equal to @time. However, they are not preferred by many programmers. The biggest reason for this is the memory allocation problem. Both these functions consume more memory and can thus affect the performance up to a certain extent. With more memory, compatibility and the performance-related issue can commonly be seen.

12: In Julia, is there any default approach that can help programmers to enhance the performance?

The fact is Julia has been equipped with a very large number of supporting tools that the programmers can deploy when they want. In a true sense, these tools are best in handling this task. One of the best available tools is Profiling. With this tool, it is possible for the programmers to monitor the quality of their code at the same time they are writing it. All the bugs can thus immediately be managed. Some programmers say it consumes time but actually, it doesn't.

13: In Julia, is it possible to use the variables which are not constant?

Yes, it is allowed. However, programmers need to make it sure that they have annotated their type before actually using them in the machine.

14: What is the project you are working with that is too complex and you need to be facing performance-related issues and errors?

To eliminate such an issue, users can proceed with the ProfileView package with the help of which everything related to complexity can be managed in a reliable manner. Another package that can be considered is Lint. Programmers can always make sure of error elimination at the right time with this package. The best part is using these packages is not at all a big deal and there is no need for the programmers to have top-notch skills in packaging.

15: Is it possible that you can use type declarations anywhere you want?

Yes, it is not possible but it needs a lot of effort. Some programmers even call it a drawback in the Julia language. However, the fact is, Julia is a high-level programming language and only experienced programmers are in a position to use them anywhere they want.

16: Can the compiler itself generate code with high performance in Julia? Why or why not?

Well, it is not possible in the case of Julia. This is due to the fact that the compiler makes use of object types and not the values assigned when it comes to generating a code. This sometimes enhances the overall length of the code.

17: In Julia, suppose you mistakenly build a complex code for a simple task. Can you make changes to it or you need to develop the same again?

Making changes in a code for cutting down its overall length and compatibility is not at all a big deal. It depends on the skills of a programmer whether he/she prefers making changes or start developing the entire code again. Sometimes modification consumes more time and effort in Julia and therefore it is a wise option to proceed with writing code.

18: Suggest one unique feature in Julia which makes it totally different than other programming languages?

In Julia, it is possible for the programmers to assign more than one task to the code they develop. In other words, a code can do different functions. However, this needs a lot of advanced programming skills. Most of programmers prefer wrapping their code in a new function to keep up the pace in this matter. This is one of the major factors that have enhanced the overall application of Julia. Many programmers who are experienced now prefer Julia because of no other reason than this.

19: Is Fortress dynamically typed or statically typed?

It is statically typed

20: What are the top features you find in Julia?

- Multiple Dispatch
- Good Performance
- Optional Typing
- This is exactly what makes Julia best in handling several tasks and the best part is it has been recognized as one of the best programming languages only because of these basic features.

21: Compare Julia with MATLAB

This is actually a tricky question. Avoid giving detailed answers and the same could be like this MATLAB contains a very large number of modules whereas

Julia doesn't. This makes MATLAB have more and in fact, advanced applications. Also, MATLAB is widely preferred in electronic and electrical applications most of the time (although it has other applications too), Julia is a computer programming language. Although MATLAB supports general programming, it is actually based on mathematical programming. Julia too can handle mathematical operations but not up to the extent MATLAB can.

22: Do you find any disadvantage in Julia while using it?

Julia comes with some cons too. The biggest one is the limited library and the same is written in Julia only. This sometimes creates compatibility issues. Also, because of its limited nature, programmers which are new have to take additional support again and again. The construction and predefining of objects is a very daunting task in Julia except for some basic operations. In addition to this, defining of functions is also limited.

23: How calling the C functions in the Julia is different from other languages?

Julia is having one of the major advantages over other languages when it comes to calling the C language. The fact is C functions can directly be called in Julia and without defining them in advance. There is actually no strict upper limit on this and C functions can be managed in an efficient manner.

24: How can process management be made easier in Julia?

It has some well-defined functions for this task. Also, there are some of the best and powerful capabilities for managing the processes.

25: How can programming be made easier in Julia?

There are a lot of Meta-programming facilities this programming language has been equipped with. Users can easily keep up the pace in every aspect and there are facilities related to making programming easy especially for beginners.

26: What is distributed computing? Is it possible to use Julia on this model?

It is basically an approach in which many computers work on a single problem. Julia can easily be deployed on this model.

27: What is the present scope of Julia being an open-source programming language?

Julia is an open-source language due to which it possible for the programmers to have customized results. Many programmers have made changes to this

language up to a good extent to get the best possible outcomes. Actually, the open-source approach makes it more flexible and with custom experiments, users are unable to explore it in a better way.

28: How can packages be managed in Julia according to you?

Julia has its own package manager and the same contributes a lot to this. All the packets and similar concepts can be managed reliably with them.

29: What does CLOS abbreviate for?

It stands for Common LISP Object system.

30: Which programming language you find similar to Julia and why

Well, the answer to this question depends on the languages a programmer has worked with. Basically, as per reviews, most programmers find it similar (not up to a large extent) to C. Many things in Julia are actually an advanced version of the same. However, in a true sense, everything in Julia is designed on its own.