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

[Julia's](https://mindmajix.com/julia-tutorial" \o "Julia Online Tutorial" \t "https://mindmajix.com/_blank) 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.

|  |
| --- |
| If you want to enrich your career and become a professional in Julia, then enroll in "[Julia Training](https://mindmajix.com/julia-training" \o "Julia Online Training" \t "https://mindmajix.com/_blank)" - This course will help you to achieve excellence in this domain. |

### **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](https://julialang.org/" \t "https://mindmajix.com/_blank)

[](https://bit.ly/3if9dmk)

### **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.

**What Is Julia?**

The Julia is *open source* programming language was designed from the beginning for high-performance.

Compiled, not interpreted, for speed. Julia is [just-in-time (JIT) compiled](https://www.code-sample.com/2019/02/julia-interview-questions-and-answers.html" \t "https://www.code-sample.com/2019/02/_blank) using the LLVM compiler framework. Julia can approach or match the speed of C.

Julia is dynamic typing with static type benefits, feels like a scripting language, and easy to use.

Julia has a rich language of descriptive data-types, and type declarations can be used to clarify and solidify programs.

The Julia created in 2009 by a four-person team - J[eff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B. Shah](https://www.code-sample.com/2019/02/julia-interview-questions-and-answers.html" \t "https://www.code-sample.com/2019/02/_blank).

The Julia was designed for scientific and numerical computation. The Julia first appeared 2012; 7 years ago and the stable release 1.1.0 in 21 January 2019.

Julia’s syntax is very similar to **[Python](https://www.code-sample.com/2017/12/python-interview-questions-and-answers.html" \t "https://www.code-sample.com/2019/02/_blank)**, but also expressive and powerful. **[Python](https://www.code-sample.com/2017/12/python-interview-questions-and-answers.html" \t "https://www.code-sample.com/2019/02/_blank)**, C, and FORTRAN libraries are just a call away.

**Why is Julia a language?**

1.      Julia is a new and open source programming language

2.      Easy to use

3.      Julia has attracted some high-profile clients

4.      Dynamic typing with static type benefits

5.      Julia feels like a scripting language

**What are the features of Julia?**

According to the official website, the main features of the language are:

1.      Very good performance, approaching that of statically-typed languages like C

2.      Dynamic type system: types for documentation, optimization, and dispatch

3.      Multiple dispatches: providing ability to define function behavior across many combinations of argument types

4.      Call C functions directly: no wrappers or special APIs

5.      Call Python functions: use the PyCall package[a]

6.      A built-in package manager

7.      Lisp-like macros and other meta-programming facilities

8.      Powerful shell-like abilities to manage other processes

9.      Designed for parallel and distributed computing

10. Coroutines: lightweight green threading

11. User-defined types are as fast and compact as built-ins

12. Elegant and extensible conversions and promotions for numeric and other types

13. Efficient support for Unicode, including but not limited to UTF-8

14. Automatic generation of efficient, specialized code for different argument types

**How to install Julia?**

Julia installation is straightforward, several ways for you to run, whether using pre-compiled binaries or compiling from source. Download and install Julia by following the instructions at *[https://julialang.org/downloads/](https://julialang.org/downloads/" \t "https://www.code-sample.com/2019/02/_blank)*

**Is Julia a good programming language?**

1.      Yes! Julia is high-level, high-performance dynamic programming language for technical computing and has similar syntax as **Python**.

2.      Easy to use

3.      Julia has attracted some high-profile clients

4.      Julia feels like a scripting language

**Is Julia a functional language?**

No! Julia is not a functional programming language. Julia offers features that are present in functional programming languages and can be used to a certain degree for coding in a functional programming style.

**Is Julia a compiled language?**

Yes! Julia is compiled, like C.

**Is Julia object oriented?**

No! Julia is not object-oriented because there is no inheritance.

**Is Julia a scripting language?**

Yes! Julia is a scripting language because **[Julia](https://www.code-sample.com/2019/02/julia-interview-questions-and-answers.html" \t "https://www.code-sample.com/2019/02/_blank)**is an interpreted, embeddable language that features just-in-time (JIT) compilation.

**Is Julia statically typed?**

No! Julia is always dynamically typed and type annotations do not make it static.

**Is Julia replacing python?**

I can’t say! Market trend of data science projects are truly based on python.

Question 1. How Do I Delete An Object In Memory?

Answer :

Julia does not have an analog of MATLAB's clean feature; as soon as a name is defined in a Julia session (technically, in module Main), it is constantly gift.

If memory utilization is your challenge, you may always replace items with ones that devour less reminiscence. For example, if A is a gigabyte-sized array which you not need, you can unfastened the memory with A = nothing. The reminiscence could be launched the following time the rubbish collector runs; you may force this to appear with gc(). Moreover, an attempt to use A will possibly result in an blunders, due to the fact maximum techniques are not defined on kind Nothing.

Question 2. How Can I Modify The Declaration Of A Type In My Session?

Answer :

Perhaps you've described a type and then recognise you want to feature a brand new area. If you do this at the REPL, you get the error:

ERROR: invalid redefinition of steady MyType

Types in module Main can not be redefined.

While this may be inconvenient when you are growing new code, there may be an extraordinary workaround. Modules may be replaced by way of redefining them, and so in case you wrap all of your new code interior a module you could redefine sorts and constants. You cannot import the type names into Main after which count on in order to redefine them there, but you may use the module name to solve the scope. In other words, whilst growing you would possibly use a workflow some thing like this:

consist of("mynewcode.Jl")              # this defines a module MyModule

obj1 = MyModule.ObjConstructor(a, b)

obj2 = MyModule.Somefunction(obj1)

# Got an mistakes. Change some thing in "mynewcode.Jl"

encompass("mynewcode.Jl")              # reload the module

obj1 = MyModule.ObjConstructor(a, b) # antique items are no longer legitimate, should reconstruct

obj2 = MyModule.Somefunction(obj1)   # this time it worked!

Obj3 = MyModule.Someotherfunction(obj2, c)

...

Linux Interview Questions  
Question three. How Do I Check If The Current File Is Being Run As The Main Script?

Answer :

When a document is run as the main script the usage of julia document.Jl one might need to set off more functionality like command line argument coping with. A manner to determine that a record is administered on this fashion is to check if abspath(PROGRAM\_FILE) == @\_\_FILE\_\_ is true.

Question four. Can I Use Using Or Import Inside A Function?

Answer :

No, you aren't allowed to have a the use of or import statement interior a function. If you need to import a module but most effective use its symbols internal a selected function or set of functions, you have  alternatives:

Use import:

import Foo

feature bar(...)

# ... Discuss with Foo symbols via Foo.Baz ...

Cease

This hundreds the module Foo and defines a variable Foo that refers back to the module, however does not import any of the other symbols from the module into the modern-day namespace. You discuss with the Foo symbols by their qualified names Foo.Bar etc.

Wrap your feature in a module:

module Bar

export bar

using Foo

characteristic bar(...)

# ... Check with Foo.Baz as certainly baz ....

Cease

give up

using Bar

This imports all the symbols from Foo, but handiest in the module Bar.

Linux Tutorial  
Question 5. What Does "kind-stable" Mean?

Answer :

It way that the sort of the output is predictable from the types of the inputs. In unique, it approach that the sort of the output can not vary depending at the values of the inputs.

The following code is not kind-solid:

julia> function risky(flag::Bool)

           if flag

               go back 1

           else

               return 1.0

           give up

       cease

volatile (normal feature with 1 method)

It returns both an Int or a Float64 depending at the cost of its argument. Since Julia can't predict the go back kind of this function at compile-time, any computation that uses it'll ought to shield in opposition to both kinds likely happening, making technology of speedy machine code tough.

Computer Science Engineering Interview Questions  
Question 6. Why Does Julia Give A Domainerror For Certain Seemingly-realistic Operations?

Answer :

Certain operations make mathematical feel but result in errors:

julia> sqrt(-2.0)

ERROR: DomainError with -2.Zero:

sqrt will most effective return a complex result if called with a complicated argument. Try sqrt(Complex(x)).

Stacktrace:

[...]

This behavior is an inconvenient effect of the requirement for kind-balance. In the case of sqrt, most users need sqrt(2.0) to present a real number, and would be unhappy if it produced the complicated number one.4142135623730951 + 0.0im. One may want to write the sqrt characteristic to replace to a complex-valued output best while exceeded a negative wide variety (which is what sqrt does in a few different languages), however then the result might no longer be type-solid and the sqrt feature might have terrible performance.

In these and other cases, you may get the result you need by way of selecting an input kind that conveys your willingness to just accept an output kind in which the result may be represented:

julia> sqrt(-2.0+0im)

zero.Zero + 1.4142135623730951im.

Question 7. Do I Want To Use A Release, Beta, Or Nightly Version Of Julia?

Answer :

You may additionally prefer the release model of Julia in case you are looking for a solid code base. Releases usually arise each 6 months, providing you with a strong platform for writing code.

You may decide upon the beta version of Julia if you don't mind being slightly in the back of the latest bugfixes and adjustments, but find the marginally quicker price of adjustments more attractive. Additionally, those binaries are tested before they are posted to make certain they're absolutely useful.

You may also decide upon the nightly version of Julia in case you need to take advantage of the modern updates to the language, and don't mind if the model to be had nowadays occasionally would not surely paintings.

Finally, you could also take into account building Julia from supply for your self. This alternative is especially for the ones folks that are comfortable on the command line, or interested in studying. If this describes you, you may additionally be inquisitive about analyzing our pointers for contributing.

F Sharp (programming language) Tutorial Mac OS X Deployment Interview Questions  
Question eight. Why Some Programmers Avoid Global Variables While Using Julia? What Is The Better Alternative According To You?

Answer :

Although the global variables are right sufficient to be depended on, the most important difficulty with them is their fee and types both adjustments often. Of direction, this leads to code optimization troubles. Later these issues make it very difficult for the customers to check the code and make it OS unbiased. Thus, the better choice is to use the nearby variables. Julia supports almost each local variable and let you come with similar exceptional even though all the variables used are local than global.

Question 9. Although Julia Is A High-stage Language, Give One Reason Due To Which Even Beginners Can Handle Tasks With It?

Answer :

the fact is Julia has been geared up with a very big range of compiler techniques. This makes it viable for the programmers to installation diverse techniques to keep up the pace definitely. Many responsibilities which are simply hard may be made simple with this approach.

Windows Administration Interview Questions  
Question 10. What Are The Good Things You Have Personally Noticed About The Julia Language?

Answer :

Julia may be considered for developing packages that are overall performance extensive   
The outputs can be made like minded with all the platforms  
Julia enables programmers to combine utility developed with different platforms in it  
It is flexible programming language.  
R Programming language Tutorial  
Question eleven. What Are The Major Applications Of Julia Where It Is Widely Preferred?

Answer :

Julia is extensively desired in both numerical computing and medical computing especially due to its overall performance. It can produce comparable results with short codes and hence make sure of quality consequences. A lot of duties related to each scientific and numerical computing can effortlessly be treated.

F Sharp (programming language) Interview Questions  
Question 12. How You Will Measure The Performance In Julia And What Are The Problems That Can Declare Their Presence While Doing So?

Answer :

One of the pleasant things approximately Julia is it has a totally simple tool available for this and the identical is @time. With this tool, overall performance can surely be measured as it should be and with out compromising with something. It is feasible for the programmers to even assemble this tool for measuring portions that are not feasible with commonplace techniques.

Linux Interview Questions  
Question thirteen. Does Julia Compiler Is Similar To Python?

Answer :

No, it's miles truely distinct and on occasion this makes many programmers to suppose Julia is a totally complex language which isn't always true.

Computer Programming Tutorial  
Question 14. Name One Approach That Simply Enables You To Run The Julia Code At A Faster Speed?

Answer :

Well, by fending off the use of worldwide variables, it's been seen that programmers can make sure of speed. However, it isn't always constantly vital to paintings with all of the programmers. It in reality relies upon at the enjoy and programming skills of a expert that how he/she can beautify the code velocity.

Question 15. In Julia, What Is The Restriction On A Code Which Is Benchmarked?

Answer :

In Julia if a code is benchmarked or may be very vital, the identical have to be assigned inside a function. If now not, there are numerous compatibility problems that may declare their presence. Also, benchmarked codes on occasion want to be stored in a separate container. It is possible to call them with a unmarried command from the same.

R Programming language Interview Questions  
Question 16. In Julia, Do You Find Any Other Function Or Tool Similar To The @time? Which One You Prefer And Why?

Answer :

toc () and tic() are the alternative features that you will put identical to @time. However, they may be now not preferred by using many programmers. The biggest cause for that is memory allocation problem. Both these capabilities consume greater reminiscence and might therefore have an effect on the overall performance up to a certain extent. With greater reminiscence, compatibility and performance associated difficulty can generally be seen.

D Programming Language Tutorial  
Question 17. In Julia, Is There Any Default Approach That Can Help Programmers To Enhance The Performance?

Answer :

The reality is Julia has been prepared with a completely big range of supporting tools that the programmers can install after they need. In actual experience, those tools are excellent in coping with this mission. One of the excellent to be had tools is Profiling. With this device, it's far feasible for the programmers to reveal the best in their code at the equal time they're writing it. All the bugs can for this reason at once be managed. Some programmers say it consumes time but actually it doesn’t.

Computer Programming Interview Questions  
Question 18. In Julia, Is It Possible To Use The Variables Which Are Not Constant?

Answer :

Yes, it's miles allowed. However, programmers want to make it positive that they've annotated their kind earlier than virtually the usage of them in the machine.

Computer Science Engineering Interview Questions  
Question 19. What Is The Project You Are Working With Is Too Complex And You Need Are Facing The Performance Related Issues And Errors?

Answer :

To eliminate such an trouble, customers can proceed with ProfileView package with the help of which the whole thing associated with complexity can be controlled in a reliable manner. Another package deal that may be considered is Lint. Programmers can always make sure of errors removal on the proper time with this package deal. The fine component is the use of those packages isn't always at all a large deal and there is no need for the programmers to have pinnacle-notch skills in packaging.

Question 20. Is It Possible That You Can Use Type Declarations Anywhere You Want?

Answer :

Yes, it isn't viable however it needs a variety of efforts. Some programmers even name it a downside within the Julia language. However, the fact is, Julia is a high level programming language and most effective experienced programmers are in a role to use them anywhere they want.

D Programming Language Interview Questions  
Question 21. Can Compiler Itself Generate Code With High Performance In Julia? Why Or Why Not?

Answer :

it isn't always feasible in case of Julia. This is due to the truth that complier makes use of object types and not the values assigned in relation to producing a code. This sometime complements the overall period of the code.

Question 22. In Julia, Suppose You Mistakenly Build A Complex Code For A Simple Task. Can You Make Changes In It Or You Need To Develop The Same Again?

Answer :

Making adjustments in a code for slicing down its typical period and compatibility isn't at all a huge deal. It depends on the capabilities of a programmer whether he/she prefer making changes or begin developing the whole code once more. Sometimes amendment consumes extra time and efforts in Julia and therefore it's miles a wise option to continue with writing code.

Question 23. Suggest One Unique Feature In Julia Which Makes It Totally Different Than Other Programming Languages?

Answer :

In Julia, it is feasible for the programmers to assign a couple of assignment to the code they expand. In other phrases a code can do specific features. However, this needs a variety of superior programming capabilities. Most of the programmers decide on wrapping their code in a new feature to hold up the tempo in this count number. This is one of the important elements which have enhanced the overall utility of Julia. Many programmers who're skilled now select Julia due to no different purpose than this.

Advanced C++ Interview Questions  
Question 24. Is Fortress Dynamically Typed Or Statically Typed?

Answer :

It is statically typed.

Mac OS X Deployment Interview Questions  
Question 25. What Are The Top Features You Find In Julia?

Answer :

Multiple Dispatch  
Good Performance  
Optional Typing  
This is exactly what that makes Julia great in managing numerous responsibilities and the nice component is it's been recognized as one of the quality programming languages most effective due to these simple functions.

Question 26. Compare Julia With Matlab?

Answer :

This MATLAB incorporates a totally massive range of modules while Julia doesn’t. This makes MATLAB to have extra and actually, superior applications. Also, MATLAB is widely desired in digital and electrical software maximum of the time (even though it has other programs too), Julia is a computer programming language. Although MATLAB supports preferred programming, it's far genuinely based on mathematical programming. Julia can also take care of mathematical operations but not upto the volume MATLAB can.

C and C++ Interview Questions  
Question 27. Do You Find Any Disadvantage In Julia While Using It?

Answer :

Julia comes with a few cons too. The biggest one is the restrained library and the equal is written in Julia simplest. This every so often creates compatibility troubles. Also, due to limited nature, programmers which can be new should take additional aid again and again. The production and predefining of objects is a totally daunting venture in Julia besides for some simple operations. In addition to this, defining of capabilities is also confined.

Windows Administration Interview Questions  
Question 28. How Calling The C Functions In The Julia Is Different From Other Languages?

Answer :

Julia is having one of the fundamental blessings over other languages in relation to calling the C language. The truth is C functions can without delay be called in Julia and without defining them in advance. There is virtually no strict upper restrict on this and C functions may be managed in an green manner.

Question 29. How Can Process Management Be Made Easier In Julia?

Answer :

It has some properly-defined functions for this venture. Also, there are a number of the exceptional and effective talents for handling the tactics.

Basic C Interview Questions  
Question 30. How Can Programming Be Made Easier In Julia?

Answer :

There are a lot of Meta-programming centers this programming language has been prepared with. Users can easily maintain up the pace in each element and there are facilities associated with making programming clean particularly for the beginners.

Question 31. What Is Distributed Computing? Is It Possible To Use Julia On This Model?

Answer :

It is essentially an technique wherein many computer systems work on a single hassle. Julia can effortlessly be deployed in this version.

Question 32. What Is The Present Scope Of Julia Being An Open Source Programming Language?

Answer :

Julia is an open supply language because of which it feasible for the programmers to have custom designed results. Many programmers have made changes into this language up to an excellent quantity to get the pleasant possible effects. Actually, open source approach makes it more bendy and with custom experiments, customers are not able to discover it in a better way.

Dart programming language Interview Questions  
Question 33. How Can Packages Be Managed In Julia According To You?

Answer :

Julia has its personal package deal supervisor and the identical contributes lots to this. All the packets and similar principles can be controlled reliably with them.

F Sharp (programming language) Interview Questions  
Question 34. What Does Clos Abbreviate For?

Answer :

It stands for Common LISP Object device.

## ****Data Structures Concept in Julia Programming Language****

Like every other programming language, Julia also has data structure concepts. Let’s learn about some of these concepts that are used for data analysis.

1. ****Vector(Array) –****A vector is a one-dimensional array which is similar to a normal array. In array, we use numbers followed by a comma as separator similarly in Julia also the vector(array) follows same.

    Let’s have a look on a piece of code.

In Julia, the index starts at ‘1’. In the above code snippet, it begins with ‘0’ since its python.

1. ****Matrix Operations****

             A matrix is another data structure that is widely used in linear algebra. We know that matrix is of a multidimensional array. Let’s see dome operation of a matrix in Julia,

****A = [1 2 3; 4 5 6; 7 8 9]   # semi-column is used to change rows****

****When we print, it looks like:      1    2     3****

****4    5     6****

****7    8     9****

****In order access element, say A [1,2] = 2****

Now for transpose of a matrix, ****A’**** then the following result will look like:

****A’ =   1   4   7****

****2   5   8****

****3   6   9****

1. ****Dictionary****

                  Another data structure is the dictionary, which is an unordered key-value pair, and the keys are always unique.

             Let’s have a look on the dictionary implementation,

****D = Dict (“string1” => “Hello”, “length” => 5) #create dictionary****

                            It  will get result :   string => Hello

                                                           Length => 5

Suppose in-order to access the dictionary we will access the key of dictionary then the value will give us as result

****D[“length]****

****o/p: 5****

to get count of dictionary use object. Count i.e  ****D.count****

****Operations of Dictionary:****

1. ****Creation        =****           ****Dict(“a” => 1, “b” => 2)****
2. ****Addition        =            d[“c”] = 3****
3. ****Removal        =            delete !(d, “b”)****
4. ****Lookup          =            get(d,”a”, 1)****
5. ****Update           =              d[“a”] = 10****

****Strings****

Next data structure is strings , strings are generally written within the quotes as {“ ”} i.e inverted commas. Similar to the python in Julia also once string is created it cannot be changes as they are immutable.

                Lets have a look,

****Text = “Hello world”****

****print(Text[1]) # will gives first character of string as H****

****Print(Text.length) # will gives the length of string 11****

There are three key phases of data structures that are used in data analysis

1. ****Data Exploration****

It’s all about finding the data more than what we have

1. ****Data Munging****

Cleaning the data and use that data for making better statistical models

1. ****Predictive Modelling****

                       Final thing is run the algorithm and have fun

****Loops, Conditions In Julia****

Like other programming languages Julia also uses the loops and conditional statements

For loop

While Loop

If condition

These are most commonly used loops and condition statement in Julia as well as other programming languages

****If and else****

In Julia we need not to worry about spaces, identation, semicolon, brackets etc instead just add end at the end of statement or condition. Lets have the syntax for if and else

Syntax:     if condition

                             Statement

                 else

                               Statement

                 end

****if elseif and else****

It also follows same as if else block follows. Let’s have look on syntax

   Syntax:            if condition

                                        Statement

                           elseif

                                         Statement

                           else

                                          Statement

                           End

Lets take an example to the above we discuused

                                         If  x > 0

                                                   “Positive”

                                         else if x < 0

                                                      “Negative”

                                          else

                                                     “Whole Number”

Lets talk about loops in Julia.

## ****For Loop****

The only difference to the loop for with other languages for loop is, in Julia for loop will have start and end counter.

Julia> for i in 0: 10: 100

                  Print(i)

           end

will gives result as:  0 10 20 30 40 50 60 70 80 90 100

Julia> for a in [“red”, “green”, “yellow”]

                Print(a, “ “)

           end

Will give result as : red green yellow

Julia> for a in Dict(“name” => “orange”, “size” => 6)

                Print(a)

           end

Name => orange   Size -=> 6

Similarly we can also iterate through 2D array, lets have look on that

A = reshape(1:50, (3, 3))

for I in A

   Print(I, “ “)

end

 The result will be as  1 2 3 4 5 6 7 8 9 …………..50

We can also use inside of functions

   function()

             for condition

                       Statement

              end

        return

 end

We know that scope of an variable inside a method or function will exists until its life span is not yet done once method or function ends and comes out then the variable scope is zero or dead

       Function()

         K = 2

             for I in 1:10 :50

                   K = k\*i

             end

        return

     end

****if we want to persist the variable to be exist in the function or method then use keyword “global” before variable name.****

****continue and break**** are the condition statements used in between the loops

for I 10:5:20

     print(i)

    continue

end

****comprehensions****

similar to python Julia also supports comprehensions

Julia> s = set([a for a in 1: 8])

Set([6,4,5,7,1,3,2,8])

Julia> [(a,b) for a in 1:5, c in 1:2]

(1,1)     (1,2)

(2,1)     (2,2)

(3,1)     (3,2)

(4,1)     (4,2)

(5,1)     (5,2)

****Generator Expressions****

 Like comprehensions generating expressions can also be used to produce result using iterable variable.

  Let’s have  a look on the example

****Julia> sum( x^2 for x in 1:10)****

****385****

****Nested Loops****

 Nested loops in Julia is quite different as of writing loop inside another loop is known to be as nested loops. But, in Julia we need not make duplicate loops instead we can use

 @show(var1, var2) variables with comma separated

    Have a loop on the piece of code for better understanding

            for a in 1 : 10, y in 1: 10

                 @show (x,y)

  Result will be:

           (x,y) = (1,1)

           (x,y)  = (1,2)

          (x,y) = (1,3)

          (x,y) = (1,4)

        (x,y) = (1,5)

        (x,y) = (1,6)

        (x,y) = (1,7)

        (x,y) = (1,8)

        ……………

       (x,y) = (10,10)

****@show is an macro that prints the names and values****

****@time will gives the complexity of loops****

Julia>     x = rand(1000);

Julia>     function sum()

                        A = 0.0

                  For I in x

                         A + = i

                  End

            Return A

         End

Julia>      @time sum()

                   0.017705 seconds (15. 28k allocations: 694. 484 kiB)

                   496.84883432553846

****While Loop****

Same as for loop while as performs only when condition is true. The following syntax is

                                     While condition

                                                     Statements

                                      End

Let’s have an example

Julia> x = 0

           0

Julia> while x < 3

                 Print(x)

                 global x+ = 1

           end

result: 0 1 2

And finally  Exceptions  with loops, like other programming language Julia also have try, catch blocks.

   Julia> s = “apple”

              try

                    S[1] = “a”

               catch e

                    Print(“caught an error: $e”)

                End

## ****Basics Of Julia For Data Analysis****

Till today many of us familiar with python or R language in the field of machine learning, data science. All those are good in their performances and predicting fasten results. Whereas Julia is such a language that can computate the large amount of data and give results in fraction of seconds.

It is very similar to the languages like python or R with respect to syntax. There won’t be no time taking for one to use Julia on data analysis. Moreever  a lot of time is spent by data scientists in-order to transform the data into good format . For that purpose Julia will provides an extensive library in dealing with the raw data and to make into good format of data I,e structured data format . There are basic steps to be followed in data analysis

1. Always explore the given data sets or data tables and apply statistical methods to find patterns in numbers.
2. Second thing is plot the data for visualization.

As in Machine Learning the data has to convert into data frames similarly using Julia we can do that. The following package provide by the Julia on Data Frames is DataFrames.jl that will converts the data into matrix format with extensions like .csv, .xlsx etc

Julia> Pkg.add(“DataFrames.jl”)

Let’s take an example to demonstrate dataframes in Julia

****Using DataFrames****

****#read the dataset****

****df = readtable(“demo.csv”, separator=’,’)****

—we have loaded the dataset into df variable and then we can print the dataset—-

****Df****

Look at the demo dataset , this is just the view of dataset its not the dataframe view.

****Dataframe functions****likefinding size , column names, to know the first n rows of dataframe set

****size(df)**** = given rows and columns (mXn)

output: [ 3, 3]

****Names(df)**** = column names

Output: [‘Aanthony’, ‘Ball’, ‘Call’]

****head(df)**** = say we give head(5) will results first five rows

output: first five rows

****Numerical Data****like describe() function which gives basic statistical data analysis such as mean, mode, sum, avg

****Categorial Data**** countmap() is function that maps the values to the no. of occurrence in the dataset.

****Dealing with Missing Data****

This is very important concept because entire game runs on this data only as of when there is loss of data obviously the predicted result will generates differ accuracy. So, in-order to maintain a good accuracy we should handle the missing data from the dataset

****showcols()**** = to check for missing values in variables

And we can replace the empty values with some related values , lets say

****df.replace(df[‘Anthony’] == “ “ , : “some data to replace”)****

****Visualization****part that generalizes the entire data and their relation among them.

         Above chart says that  rainfall over a period of  time interval keeps on increasing  [cm’s]

****Point to remember****

         Histogram charts should always be divide into bins i.e more bins more data analyzed

         Data Analysis is not limited to data visualization after modelling also data analysis is done.

## ****Exploratory Data Analysis With Julia****

     Exploratory Data Analysis is used in understanding data in terms of data features, variables and their relationship among them. Always the main step to be do is understand the data set properly. There are some methods to be followed

****Methods to be followed on given dataset (explore)****

1. Statistical Methods or Functions
2. Visual Plot Techniques

****To the data table apply some statistics****

****Step1:****installing Data Frame Package

    Using Julia over the data table or data set a data structure concept called Data Frames is   used. As of data frame can handle multiple operations like speed , accuracy and compatibility

   Data frames to be used in Julia should be installed first

   The following command is used to install the data frame

            Using Pkg

            Pkg.add(“DataFrames”)

****Step2:****Next download the data set

****Step3:****Then install necessary packages, CSV packages, Data Frame etc

            using DataFrames

            using CSV

a = CSV.read(“sample.csv”)

****Step4:**** Then have data exploration

                Data exploration has to be done over the data set because it gives the relations   among data variables, what are the functions ,column names, lists etc

            using DataFrames

            using CSV

            a = CSV.read(“sample.csv”);

            size(a)

            names(a)

            head(a, 10)

****Describe Function****

   Describe function is used to give mean, mode, meadian, some basic statistical data over the data set

****Mean:****Mean givesthe average of dataset or datatable.

****Mode:****Mode will gives the observed value of dataset or datatable

****Median****: Median will gives middest value of datatable or dataset.

                        using DataFrames

                        using CSV

                        a = CSV.read(“sample.csv”);

                        describe(a)

                        describe(a, :all, cols = :SepalLength)

****Apply visual plot techniques over the data set****

Visual plotting in Julia can be achieved using plot libraries like Plots, StatPlots and Pyplot

****Plots :****it’s an high level plotting package which interfaces with other plotting packages called

‘****back-ends’****. Actually they behave like graphic engine that provides graphics

****StatPlots:****Itsan plotting package including with the Plots package especially these StatPlots are used for some statistics

****Pyplot:****Itsanpackage with Matplotlib which is library of python.

These libraries can be installed as follows:

Pkg.add(“Plots”)

Pkg.add(Statplots”)

Pkg.add(“Pyplot”)

****Distribution Analysis****

Here, in distribution Analysis Julia is performed using various plots such as histograms, scatterplot, boxplot

                                     using DataFrames

                                     using CSV

                                     a = CSV.read(“sample.csv”);

                                     using Plots

                                      Plots.histogram(a[:SepalLength], bins = 50, xlabel = “Sepallength”,

                                                                                 Labels = “length in cm”)

Similarly we can plot graph using different formats like histogram etc

## ****Using R and Python Libraries in Julia****

    Julia programming language is such a powerful language with many libraries and packages included as well as it also provides outside libraries to be accesses.

   You may get doubt like if Julia is has such powerful libraries then why is needed to access from other languages especially Python and R because even the libraries are there but they might be very young to be used that’s the reason Julia provides ways to access libraries from R and python.

 To call python libraries in the Julia PyCall is the package that will enables to call python libraries from Julia code

    Julia> Pkg.add(“PyCall”).

****PyCall**** provides many good functionality that helps in manipulating python in Julia using type ****PyObject****

 The following are the steps to be followed in order to call python packages

   Step1: using Pkg

   Step2: Pkg.add(“PyCall”)

   Step3: using PyCall

   Step4: @pyimport python\_library\_name

   Lets see basic programe on how to import math package of python into Julia

                  using Pkg

                  Pkg.add(“PyCall”)

                  using PyCall

                  @pyimport math

                  Print(math.cos(90))

Second example to import Numpy package into Julia language

                  using Pkg

                 Pkg.add(“PyCall”)

                 using PyCall

                 @pyimport numpy

                 A = numpy.array([2,1,4,3,

                                                         5,7,6,8])

                 Print(A)

  Output:

             [2, 1, 4, 3, 5, 7, 6, 8]

## ****Using Pandas With Julia****

 If you are familiar with the library pandas in python then it is same as Julia also. Using Pandas we can filter the data or analyze the data lot more. Especially converting the data into dataframes which is package of pandas library .

         DataFrames will helps to visualize the data into multidimensional array i.e matrix format

                         Julia> Pkg.add(“Pandas”)

           Lets see an example using pandas with Julia

                     using pandas

                     df = read\_csv(“job.csv”)

df = DataFrame(Dict(:company => [“google”, “Apple”, “Microsoft”], :job=>[“sales executive”,

                                         “business manager”, “business manager”, “computer manager”],

                                         :degree=>[“bachelors”, “masters”], :salary=>[0,1]))

typeof(df)

head(df)   # will gives first five rows of data

describe(df)

If  df[“job”] == “computer manager”

              df[“job”] = “manager”

end

df.mean(“salary”, axis = 1)

 So, there are many operations which are basics of pandas and are used on data set as cleaning procedure .

  Cleaning includes like removing null values, missing values replacement and modifying the data which is in appropriate .

 Pandas is most powerful library not only in python but also in Julia .

## ****Introduction To DataFrames.jl****

As we all know that Julia has the library that handles the data transformation like python and R does i.e DataFrames. This approach although looks similar to python or R but it differs during API call. For complex data tables DataFramesMeta concept is used

   Lets see how to install and import the library

* To install library use command  ****Pkg.add(DataFrames****)
* To load the library use command ****using DataFrame****

After doing above steps the next is to load the data set . The following way to read the data table is.

                                  using  CSV

                                  Datatable = CSV.readtable(“sample.csv”)

****Fruits           Sweet           Sour****

****Apple             80%             10%****

****Orange          90%              10%****

****Pineapple      100%             0%****

After loading CSV file check for the missing values suppose if the column has missing values in the top most rows due to using type-auto recognization then there are chances of having error rate. Manually we have to remove the error tendancy from the data set.

         To find missing value

Types = Dict(“Florida” => Union{Missing, Int64})

If we want to edit the values of imported dataframes then don not forget to use copycols = true

* Use the package from the stream HTTP:

                 Using DataFrame , HTTP, CSV

                   Resp  = HTTP.request(“GET”, [https://somesite@domain.com?accesstyep =       “Download](https://somesite@domain.com/?accesstyep =       ))

                     df = CSV.read(IOBuffer(String(resp.body))

* Again create df from scratch

             Df = DataFrame(

              Color = [“red”, “yellow”, “orange”, “white”]

              Shape = [ “circle”, “rhombus”, “vertical”]

              Border = [“line”, “dotted”, “line”]

               Area = [1.1,1.2,1.3,2.5])

* There are many possibilities with df like convert matrix form data to vector form :

       For example:

 df = DataFrame([[mat[ : , i]…] for I in a : size(mat, 2)], Symbol.(headerstrs))

Using dataframes package we can do a lot mpre with the data set or data table. Always the given dataset has to be converted into data frames i.e matrix conversions so that one can analyze the data properly and handle it regarding null values, missing values..

****Get Some Insights of Data****

* first(df, size)
* show(df, allrows=true, allcolls = true)
* last(df, size)
* describe(df)
* unique(df.fieldName)
* names(df)
* size(df)
* to iterate over each column [for  a in eachcol(df)]
* to iterate over each row  [for a in eachrow(df)]

****Filter****

In-order to refer to some columns there are two ways in data frame like referencing the stored values into the object or copying them into another new object

1. Myobject = df[ !, [cFruits]]  {store values in object}
2. newObject = df[ :, [cFruits(s)]  { Copying entire into new object }

You know we can also query using data frames let’s see how we can do

                   dfresult1 = @from I in df begin

                                           @where i.col > 1

                                            @select {aNewColName = i.col1, i.col3}

                                             @collect DataFrame

                                    end

                  dfresult2 = @from I in df begin

                                         @where i.value != 1 && i.cat1 in [“red”, “yellow”]

                                          @ select i

                                           @collect DataFrame

                                    end

****Replace Data****

We can replace the values of column with other data  that to dictionary based values

      df.col1 = map(key ->mydict[key], df.col1)

Can be concate the values of column using dot operation     df.a = df.b

 Appending rows : push! (df, [1 2 3])

 Delete rows: deleterows !(df, rowIdx)

****Change the structure of data or holding object****

Here dataframe can be used to change name of column, data type of column , delete column, rename column or else replacing position of columns. Type casting which can be help to convert any kind of data type

   From int to float: df.a = convert(Array{Float32, 1}, df.a)

****Sorting****sort ! (df, cols = (:col2, :col1), rev = (false, false))

So, Dataframes is most powerful library or package for data handling . It will handle missing values which cause a lot error tendancy . we can split the datasets and re combine them together and apply some statistical operations like aggregate functions,

## ****Visualization in Julia Using Plots.jl****

 This is another way to explore the data and analysis i.e by doing visualization using various kinds of plot formats.

In Julia we can even plot the graph for the data using library. But, Julia does not provide direct library of its own instead it provides to use libraries of your own choice in Julia programs.

To have this functionality we need install some packages:

                     Julia> Pkg.add(“Plots.jl”)

                     Julia> Pkg.add(“StatPlots.jl”)

                     Julia> Pkg.add(“PyPlot.jl”)

This Plots.jl is act as interface to any plotting library such that using libraries in Julia we can plot data .

StatPlots.jl is supporting package for Plots.jl

PyPlot.jl will act as Matplotlib of python

Now, let’s see some data visualization plots using pyplot.jl and also we can get information about data table more using plots.

               Using CSV

              S = CSV.readtable(‘Venice.csv’)

              using Plots, StatPlots

             pyplot()   #set backend as matplotlib package i.e matplotlib.pyplot

             Plots.histogram(dropna(train[: ApplicationTax]), bins = 50, xlabel = “ApplicationTax”,              labels = “Frequency”)        # plot histogram

If you observe the plot we have different values with depriciation in the plot , so that is the reason why we need the bins as 50 or relevant to that

In other scenario we can look at box plots to understand the distributions of bins in the above graph clearly.

   Lets see another way of visualizing the plot:

               Plots.boxplot(dropna(train[: ApplicationTax]), xlabel = “”ApplicationTax”)

If u look the plot below it tells us the preence of extreme values . This can be attributed to the Tax in the society. And also we can segregate the part based on their profession  in the society

Plots.boxplot(train[: Education], train[: ApllicationTax], label = “ApplicationTax”)

****ApplicationTax****

   Now, if u see there is no  difference between the Tax of the persons and also the Profession

    of persons based on which the tax is paid i.e high or low tax .

   Lets have look on other charts like ****line chart****, pie chart for rain data in a year/month

                  using CSV

                  a = CSV.read(“sample.csv”)

                  plot(a.month, a.max)

This graph will says that a month with maximum rain

Next, we will see ****scatter chart**** by using same data i.e rain data in a year/month

                    Scatter(a.Rain, label = “y1”)

This chart says that the rainfall is vary’s on every year i.e increase as the year goes on increase

Similarly lets look on the ****pie chart**** also with same rain data in a year/month

W = 1:5; y = rand(5); #plotting data

Pie(x,y)

The pie chart gives an analyzation of more area with rainfall followed by average and less rainfall per year or month.

## ****Histogram Chart****

Histogram(a.Rain, label = “Rainfall”)

We can easily find by histogram chart the rainfall is varies in a year with unequal distribution of rainfall.

The graphs and charts can be used for visualizing or seeing the trends.

So, I hope we learnt topic in Julia i.e plots. so far we completed all the basic charts that are used in Julia with plot library.

## ****Data Munging In Julia****

 While we did analysis of data there are some problems that we encountered i.e missing values, null values all these problem has to be remove under data analysis step. To do so, data munging is a technique or process to handle the missing values in data table or data set i.e converting the raw data into some format that can be utilized for data analysis . It is also known as ****Data Wrangling****

It is one of the most important component in data science .

The following packages that are required:

****RDataset****this packagae will load the data set generally used in R language since julia can also be access the libraries or packages of other languages like R it can be installed as follows

 Julia> Pkg.add(‘RDatasets’)

As we know that inorder to convert into multidimensional array format to a data set in python or R we use data frames . similarly here in julia ****DataFrames**** and ****DataFramesMeta****  will provide the functionality

     Julia> Pkg.add(‘DataFrames’)

     Julia> Pkg.add(‘DataFramesMeta’)

     Let’s load the data set

     It contains columns

     company

     job

     degree

     salary

So, the analysis of this data set is if an employee having bachelors degree he or she can be promoted or salary can be increased and condition applys i.e varies with company.

        using RDatasets

        sal = dataset(“datasets”, “sample”)

        head(sal)

****it gives the same dataset as we saw in the above figure****

****Using groupby():****

The groupby function will group the data in all the columns to a given value . It splits the datagrame and those split dataframes are again split into subsets then the function is used. The indices for data set starts from indices 1 when we use the groupby()

   The following syntax:

        groupby(a, :col\_names, sort = false, skipmissing = false)

     Parameters are

     a : dataframe

    :col\_names:  column names on which data set is split

    sort: to return the data set in sorted manner by default it is false

    skipmissing: it will decides whether to skip the missing values or not , by default false

                 using RDatasets

                  sal = dataset(“datasets”, “sample”)

                  groupby(sal)

****by() function****

This by() function will performs ****split-apply****method which means it will split the column and then apply the by() function . The syntax as follows:

                 by(a, :col\_names, function, sort = false)

The Parameters :

              a: dataframe

              col\_names: the split of columns

              function: function applied on each column

              sort: the dataframe to be return sort order by default it is false

 lets split the dataframe and show the column who are eligible for salary promotion

                              using RDatasets

                              using Statistics

                             sal = dataset(“datasets”, “sample”)

                             by(sal, [:job, :degree]) do a DataFrame(Mean\_of\_Salary = mean(a[:Salary]),

                                                       Variance\_of\_Salary = var(a[:Salary])

                            End

       \* Mean of Column Salary

****aggregate() function****

aggregate function will also follows split- apply method . columns are split and then the function is applied to the specified column .

                    aggregate(a, :col\_names, function)

The Parameters are:

                             a: dataframe

                             col\_names: the split of columns

                             function: function applied on each column

          using RDatasets

          sal = dataset(“datasets”, “Sample”)

         aggregate(sal, :job, degree)

****Missing****

     In Julia the missing values are represented using special name i.e ****missing****which is instance for the type Missing.

          Julia> missing

                    missing

let’s see for the type of of missing

          Julia> typeof(missing)

                    Missing

Missing type will allows  users to create Vectors and DataFrame column with missing values.

Let we see an example :

   Julia> x = [0, 1, missing]

                 3-element Array{Union{Missing, Int64}, 1}:

                  0

                  1

                        Missing

   Julia>  eltype(x)

              Union{Missing, Int64}

   Julia> Union{Missing, Int}

               Union{Missing, Int64}

 Julia> eltype(x) == Union{Missing, Int}

            True

While performing some operations missing values can be excluded using a technique called as

 “skipmissing”

Julia>  skipmissing(x)

           Base.Skipmissing{Array{Union{Union{Missing, Int64}, 1}}(Union{Missing, Int64}[0,1,missing].

Lets take an scenario i.e I want to find the average of all missing values.

Julia> avg(skipmissing(x))

       4

 Julia> collect(skipmissing(x))

           2-element Array{Int64, 1}

****Coalesce is the  function which is used to replace null value with some other values.****

 Julia> coalesce(x, 0)

    3-element Array{Int64, 1}

    1

    2

    0

  Similarly we may also have missing values or null values in rows . For that we can use         dropmissing and dropmissing! to remove the missing values .

Julia> df = DataFrame(I = 1:4,

                                         P = [missing, 3, missing, 2,1]

                                         Q = [missing, missing, “c”,“d”,”e”])

4X3 DataFrame

Row | I                x                        y

        |  Int64        Int 64                String?

    1 |    1           missing              missing

    2 |    2              3                     missing

    3 |    3           missing                 c

    4 |    4              2                        d

   Julia> dropmissing(df)

              2X3 DataFrame

             Row | I             x                     y

                     |  Int64     Int64             String

             ————————————————

              1 |      4           2                     d

              2|       5           1                     e

 One more point i.e Missings.jl  package provide the few functions inorder to work with missing values.

             Julia> using Missing

            Julia>  Missings.replace(x,1)

                  Missings.EachReplaceMissing{Array{Union{Misssing, Int64}, 1}, Int64}(Union{Missing, Int64}[1,2,missing], 1)

These are some basic functions used to handle the data while analyzing i.e mainly to remove null and missing values from the data set. This is what data munging.

## ****Building a Predictive ML Model****

Till now, we have saw how the data set  should be handle , how to overcome the problems especially like missing values in the data set or null values and more-ever visualizing the data using library plot.pl, StatPlots.

  Now, we will see how to build an Machine Learning Model using Julia programming language.

In python scikitlearn is the package or library that will provides all the necessary models , similarly in Julia Scikitlearn package will provides.

                   Julia> Pkg.add(“Scikitlearn.jl”)

This package will act as interface to the python’s Scikitlearn package

*“ Since Julia can access Packages of Python”*

****Label Encoder****

   In python labelencoder() is the package that can be found from ****Scikitlearn.Preprocessing****which will converts data into numerical format data [0,1,2…………….]

 In Julia also we will convert the data into numerical format. The one who are familiar with python they can understand why label encoder is used.(it becomes easy to access any column of data with numerical values).

          Lets encode sample data

          using ScikitLearn

         @sk\_import preprocessing: LabelEncoder

          encoder = LabelEncoder()

           data = [“apple”, “orange”, “papaya”]

           for col in data

                  train[data] = fit\_transform! (encoder, train[data])

          end

Now, we will define generic classification function which takes model as input and gives us the accuracy and cross-validation scores.

          using ScikitLearn : fit!, predict, @sk\_import, fit\_transform!

         @sk\_import preprocessing : LabelEncoder

         @sk\_import model\_selection : cross\_val\_score

         @sk\_import metrics: accuracy\_score

         @sk\_import linear\_model: LogisticRegression

         @sk\_import ensemble: RandomForestClassifier

         @sk\_import tree: DecisionTreeClassifier

        function classification\_Model(model, predictions)

                         p = convert(Array, train[:13])

                         q = convert(Array, train[predictions])

                         r = convert(Array, test[predictions])

          # check for fitness of model

                   fit! (model, p, 1)

          #predicitons on training data set

                   Predictions = predict(model, p)

          #accuracy

                   Accuracy = accuracy(Predictions, q)

         #cross\_validation

               Cross\_score = cross\_val\_score(model, p, q, cv = 5)

          #print cross score

                  print(“cross score: “, mean(Cross\_score))

             fit!(model, p, q)

            Out = predict(model, r)

           Return Out

   End

## ****Logistic Regression****

                Using logistic regression we are going to calculate the accuracy and cross validation scores like what we have done in the above classification\_Model function.

   LogisticRegression in Julia is similar to Python. Logistic Regression in Machine Learning is an classification algorithm which is used to predict the probability of dependent categorial value. The dependent values will be either in 0 or 1.

       Logistic Regression can be classifies into two classifications

1. Binary Classification
2. Multiclass Classification

        Lets see the logistic regression plot in visual

    Mathematical Equation For Logistic Regression :  ****1/ 1+ e^-x (or) 1/ 1 + e^-z****

lets make use of model and determine the accuracy for the persons obesity

                        model = LogisticRegression()

                        predict\_value = [:Obesity]                                => this code snippet add as

                        classification\_Model(model, predict\_value)          continuation to above code

The result will be :

                       Accuracy: 80.9% Cross-Validation Score: 80%

The accuracy and cross\_score are good but if you need more  accuracy then change the column or variables and apply model again.

                      Predict\_value = [:Obesity, :Age, :Weight]

                      Classification\_Model(model,predict\_value)

The result wil be :

                     Accuracy: 88% Cross-Validation-Score: 87.9%

This how  logistic regression classifies. Generally problems which are not ended at particular limit instead they tend to change frequently for those problems Logistic Regression Model is used to solve.

## ****Decision Tree****

     Decision Tree is another Model under Classification. Decision Tree works on parent child scenario, always the child node will be consider as the result node vice-versa parent node is consider as root node which takes decisions. The working process of decision tree

* Decision tree selects best attribute using Attribute Selection Measure
* Selected attribute will be consider as root node
* Then again it divides into sub nodes until it reaches to leaf node

The mathematic equations or formulae used in decision tree are:

* ****Information Gain (ig)**** = ****-p/s log(p/s) – n/s log(p/s)****
* ****Gini Index = ig – Entropy****

****Information Gain:****

         This will gives  us the information regarding an attribute i.e how important an attribute to the data set as of attribute posses feature od vectors through which we can identify the relations of parent and child nodes.

****Entropy****

Entropy , we can get this from information gain such that information gain will gives us the

       entire relation of data set whereas the entropy will tells us the impurities from the data set.

                          The higher entropy the more information gain.

       Let’s say two classes and we want to find the which class belongs to same category

       Suppose class A belongs to some x category and B also same category x then it is not

       a good entropy as 0. if it is like 50 – 50 % then it is good entropy and data set is good as 1

****Gini Index****

****Gini Index**** will gives the pure impurity which means it will calculate the probability of s

             Selected attribute if all are linked to same attribute then that attribute is pure attribute  or

             Belongs to same classs.

         Decision tree gives higher accuracy than logistic regression , since decision tree  follows the parent and child concept by taking exact decision.

    Let’s see the implementation part for decision tree by considering an example.

     We are going to calculate the results i.e accuracy and cross-validation-score of student using decision tree classifier algorithm. Now, the attributes for student are Name and age

   Conside Name and Age columns possess some 10 rows of random data and we used decision tree classifier algorithm, which it should its gives best accuracy and cross-validation-score.

                            model = DecisionTreeClassifier()

                            predict\_value = [:Student, :Name, :Age]

                            classification\_Model(model, predict\_value)

 The result will be as:

                            Accuracy: 81.95%  Cross-Validation Score: 75.6%

We can again increase the accuracy to more extent by changing the input columns so that maximum accuracy can be obtained.

                                “***Always find maximum accuracy and score”***

                        Predict\_value = [:Student, :Name, :Class, :Age]

                        Classification\_Model(model, predict\_value)

The result will be as:

                            Accuracy: 85.78% Cross-Validation Score: 80.7%

## ****Random Forest****

Random Forest, it is an another algorithm that is  capable of performing both regression as well as classification tasks with a technique called “Bootstrap” and “Aggregation” known as bagging.

Random Forest having multiple decision trees as its learning models  then it performs random row sampling and feature sampling to the dataset by applying a model. This is called as Bootstrap.

  Let’s see the approach or process involved to use  random forest algorithm

* We should design a relevant question to the given information or data set
* And one more thing to make sure is convert all the data to accessible format or else convert into that format
* Develop a machine learning model
* categorize data set into training data and test data
* Apply model and find the accuracy or score for the testing data
* Repeatedly change the values so that accuracy will reach to max

Let’s see the implementation part of Random Forest

     We are going to calculate the results i.e accuracy and cross-validation-score of bank customers using RandomForestClassifier algorithm to segregate customers based on loan status. Now, the attributes for customer are Name , Age, Sex, Loan.

   Conside Name , Age, Sex, Loan columns possess n rows of random data and we used RandomForestClassifier algorithm, which it should its gives best accuracy and cross-validation-score.

                model = RandomForestClassifier(n\_value = 100)

                predictions = [:Name, :Age, :Sex, :Loan]

                classification\_Model(model, prediction)

Accuracy : 100% Cross-Validation Score : 80%

Here, we got 100% accuracy for the training data set. This is the problem overfitting and can be resolved in two ways

1. Reducing the number of predictions
2. Tuning the model parameters

 model = RandomForestClassifier(n\_value = 100, min\_samples\_split = 50, max\_depth = 20,

                                                                                                                                n\_jobs  = 1 )

classification\_Model(model, predictions)

The result will be :

                                 Accuracy : 83% Cross-Validation Score : 80%

Here if you see even though accuracy is reduced the score is increased which means the model is doing well  Random Forest will use multiple decision trees which in return gives different predictions.

As possible as avoid complex modelling technique as black box without understanding the concepts.

## ****Using ggplot2 in Julia****

ggplot2 is an data visualization package used in statistical programming language R. ggplot will breaks the data into semantic components such as scales and layers.

Since, Julia can access the libraries of python and R so ggplot2 can be installed with Julia and include.

  Lets see how to load R package into Julia

                  Using RCall

                 @rlibrary ggplot2

There might be question araise like having most powerful Julia with all packages include why to use R packages for data visualization ?

 Plots.jl is powerful package but unfortunately its interface is similar to R language . If user wants to visualize the plot then it is very difficult to remember all the commands as there are more to remember .

So that’s the reason why Julia uses R packages for data visualization even python libraries too.

  Lets consider an example with this scenario:

****Using Julia plot.jl  package****

             plot(plot\_data\_1, a = “a”, b = “b”, Geom.line,

                     layer(Geom.line, a = “a”, b = “text” , Theme(default\_color = “red”)),

                    layer(Geom.line, a = “a”, b = “a\_mc”, Theme(default\_color = “blue”)),

                    layer(Geom.line, a = “a”, b = “a\_mf”, Theme(default\_color = “orange”)),

}

****Using R ggplot package****

ggplot(plot\_data\_1, aes(a = “a”, b = “b”)) +

            geom\_line(color = “red”) +

            geom\_line(aes(b = :a\_mc), color = “green”) +

            geom\_line(aes(b =:a\_mf), color = “violet”)

if u observe above piece of code using ggplot which is very simper when compared to Julia plots.jl . The user wont get frustrated on using R package as it is simpler than Julia package

 The above code might be have some issues since, Gadfly do not follow grammer of graphics strictly like font size, data visualizing pattern, color pattern on the line  etc.

  By considering all these we can say at the end of day that packages of Julia are bit complex than the packages of R or python . R packages gives good interoperability and difficulty problems can be solved easily.

      The package ggplot in Julia installed as follows:

Julia> Pkg.add(“RDatasets.jl”)

Julia> Pkg.add(“RCall.jl”)

Lets look on the plot visualized using ggplot library

  using Rcall, RDatasets

  val = datasets(“datasets”, “demo”)

  library(ggplot2)

  ggplot($demo, aes(p =”ASD” , q =”AOSI Total Score(Month 12)” )) + geom\_print()

****Thoughts of Conclusion****

Finally Julia is such powerful language that provides accessability packages related to python and R by PyCall and RCall . Julia is ideal in its nature and its syntax too compared to python particularly when writing highly function code .

Julia is better programming language we can say . Strong reason might be its best for numerical computation

<https://julialang.org/learning/tutorials/>

<https://www.tutorialspoint.com/julia/index.htm>

https://www.freecodecamp.org/news/learn-julia-programming-language/