# Short Questions

1. What is Ruby and what makes it a scripting language?
2. Explain the structure and execution of Ruby programs.
3. How does RubyGems facilitate package management in Ruby?
4. Describe the use of Ruby in web development.
5. What is the role of CGI scripts in web development using Ruby?
6. How are cookies utilized in Ruby for web applications?
7. What factors influence the choice of webservers when using Ruby?
8. Explain the concept of SOAP and its relevance in web services with Ruby.
9. How does RubyTk contribute to building graphical user interfaces?
10. What are widgets in RubyTk and how are they used in GUI design?
11. Describe how events are bound in RubyTk.
12. Explain the purpose and usage of the Canvas widget in RubyTk.
13. How can you implement scrolling in a RubyTk application?
14. What are the key differences between Ruby and Rails?
15. How does Rails simplify web application development in Ruby?
16. What is the significance of MVC architecture in Rails?
17. Explain the concept of ActiveRecord in Ruby on Rails.
18. How do you create a new Rails application?
19. What is the purpose of the "rails generate" command in Rails?
20. How does Rails handle database migrations?
21. Describe the use of routes in a Ruby on Rails application.
22. What is RESTful routing in Ruby on Rails?
23. How can you create a new controller in a Rails application?
24. What is the purpose of layouts in Rails views?
25. Explain how to use partials in Rails views.
26. What is the role of helpers in Ruby on Rails?
27. How do you perform validation in Rails models?
28. Describe the use of callbacks in Rails models.
29. What is the purpose of before_action and after_action in Rails controllers?
30. Explain how to handle authentication in a Ruby on Rails application.
31. What is the asset pipeline in Ruby on Rails and why is it important?
32. How does Rails support internationalization and localization?
33. What are the advantages of using a Ruby on Rails framework for web development?
34. What is a gem in the context of Ruby development?
35. How do you install a gem using RubyGems?
36. Explain the difference between local and global gem installation.
37. What is the purpose of the Gemfile in a Ruby project?
38. How can you list installed gems on your system?

39. Describe the process of updating a gem to its latest version.
40. What is a shebang line in Ruby scripts and why is it important?
41. How do you handle command-line arguments in a Ruby script?
42. Explain the purpose of the 'require' statement in Ruby.
43. What is the 'load' method used for in Ruby scripts?
44. How can you define and use modules in Ruby?
45. Describe the concept of method chaining in Ruby.
46. What is the 'yield' keyword used for in Ruby?
47. How do you raise and handle exceptions in Ruby programs?
48. What is a lambda function in Ruby and how is it defined?
49. Explain the use of regular expressions in Ruby.
50. How can you create and use custom libraries in Ruby programs?

51. How can Ruby objects be extended using C?
52. Explain the concept of extending Ruby with C using an example.
53. What is the Jukebox extension in the context of Ruby?
54. How does memory allocation work in Ruby when extending it with C?
55. Describe the Ruby Type System and its significance in C extensions.
56. What are the steps involved in embedding Ruby into other languages?
57. How can you embed a Ruby interpreter into a C/C++ program?
58. What is the purpose of the 'ruby.h' header file when embedding Ruby?
59. Explain the role of the 'ruby_init()' function in embedding Ruby.
60. How can you execute Ruby code from within a C/C++ program?
61. What is the difference between embedding Ruby and extending Ruby with C?
62. Describe the process of passing data between Ruby and C in an embedded scenario.
63. How can you evaluate Ruby expressions dynamically within a C program?
64. What is the significance of the 'rb_eval_string()' function in embedding Ruby?
65. How can you call Ruby methods from C code?
66. Explain the concept of Ruby callbacks in an embedded environment.
67. What is the Global Interpreter Lock (GIL) in Ruby, and how does it affect embedding?
68. How can you handle exceptions raised in Ruby code within a C program?
69. Describe the concept of multithreading when embedding Ruby.
70. What are some potential use cases for embedding Ruby in other languages?
71. How does embedding Ruby enhance the functionality of a C/C++ application?
72. What are some alternative scripting languages that can be embedded in C/C++?

73. How do you manage memory and resources when embedding Ruby into other languages?
74. Explain the role of Ruby's Garbage Collector in an embedded scenario.
75. What is the purpose of the 'Data_Wrap_Struct' function in C extensions?
76. How can you create Ruby classes and objects from C code?
77. Describe the process of defining methods for Ruby classes in C extensions.
78. What is the Ruby Data Object (RDO) and when is it used?
79. How can you perform type checking and conversions in C extensions?
80. Explain the significance of the 'rb_define_module()' function in C extensions.
81. How do you handle exceptions in C extensions when calling Ruby methods?
82. What are some best practices for documenting and testing C extensions in Ruby?
83. How can you make use of Ruby's dynamic typing system in C extensions?
84. Describe the process of releasing memory when using C extensions.
85. What are the advantages and disadvantages of extending Ruby with C?
86. How can you integrate Ruby code seamlessly into a C/C++ application?
87. What are the potential challenges of embedding a scripting language like Ruby?
88. How does the embedding of Ruby affect the performance of a C/C++ program?
89. Explain how you can pass data between Ruby and other languages in an embedded context.
90. What is the role of the 'rb_require()' function in C extensions?
91. How can you debug issues in Ruby code when embedded in C/C++ applications?
92. Describe the steps to load Ruby scripts from within a C/C++ program.
93. What are some security considerations when embedding Ruby in other languages?
94. How can you handle Ruby gems and external libraries within an embedded environment?
95. Explain the concept of Ruby's virtual machine and its role in embedding.
96. What is the purpose of the 'ruby_cleanup()' function in an embedded scenario?
97. How do you handle multi-threading synchronization when using embedded Ruby?
98. What is the significance of the 'rb_protect()' function in C extensions?
99. How can you ensure compatibility with different Ruby versions in C extensions?

100. What resources and documentation are available for learning more about embedding and extending Ruby?

101. What distinguishes scripts from programs, and how do they differ in execution?
102. Discuss the historical origin of scripting and its evolution over time.
103. What are the key characteristics of scripting languages?
104. How does scripting play a significant role in modern computing environments?
105. Explain the importance of scripting in automating tasks and simplifying processes.
106. Describe the primary uses of scripting languages in various domains.
107. What are the applications of scripting languages in web development?
108. How does web scripting contribute to dynamic web page generation?
109. Provide an overview of the diversity of scripting languages available today.
110. What are the core concepts of Perl, and how does it fit into the scripting landscape?
111. Define variables in Perl and discuss their scope and naming conventions.
112. Explain the concept of scalar expressions in Perl with examples.
113. Describe the control structures available in Perl for flow control.
114. What are arrays in Perl, and how are they used to store data?
115. Differentiate between arrays, lists, and hashes in Perl.
116. How are strings represented and manipulated in Perl?
117. Explain the importance of pattern matching and regular expressions in Perl.
118. Provide examples of common regular expression patterns in Perl.
119. What is the role of subroutines in Perl, and how are they defined?
120. Discuss the benefits of modularizing code using subroutines in Perl.
121. How do you pass arguments to Perl subroutines, and how are they accessed?
122. Explain the concept of scoping in Perl variables.
123. What is lexical scoping, and how is it implemented in Perl?
124. Describe the difference between global and lexical variables in Perl.
125. How do you declare and work with global variables in Perl?