

1. `<start>` -> require "ifj21" `<program>`
2. `<program>` -> `<fnc_dec>` `<program>`
3. `<program>` -> `<fnc_call>` `<program>`
4. `<program>` -> `<fnc_def>` `<program>`
5. `<program>` ->  $\epsilon$
6. `<fnc_dec>` -> global id\_fnc : function(`<params_dec>`) `<return_type>`
7. `<params_dec>` ->  $\epsilon$
8. `<params_dec>` -> `<data_type>` `<params_dec2>`
9. `<params_dec2>` ->  $\epsilon$
10. `<params_dec2>` -> , `<data_type>` `<params_dec2>`
11. `<return_type>` -> : `<data_type>`
12. `<return_type>` ->  $\epsilon$
13. `<data_type>` -> integer
14. `<data_type>` -> number
15. `<data_type>` -> string
16. `<fnc_call>` -> id\_fnc (`<value>`)
17. `<value>` ->  $\epsilon$
18. `<value>` -> `<value_last>` `<value2>`
19. `<value2>` -> , `<value_last>` `<value2>`
20. `<value2>` ->  $\epsilon$
21. `<value_last>` -> id\_var
22. `<value_last>` -> integer\_value
23. `<value_last>` -> number\_value
24. `<value_last>` -> string\_value
25. `<fnc_def>` -> `<fnc_head>` `<fnc_def2>` end
26. `<fnc_head>` -> function id\_fnc (`<params_def>`)
27. `<fnc_def2>` -> `<fnc_body>` `<return_void>`
28. `<fnc_def2>` -> : `<data_type>` `<fnc_body>` `<return>`
29. `<params_def>` ->  $\epsilon$
30. `<params_def>` -> `<var_def>` `<params_def2>`
31. `<params_def2>` ->  $\epsilon$
32. `<params_def2>` -> , `<var_def>` `<params_def2>`
33. `<var_def>` -> id\_var : `<data_type>`

34. `<return>` -> return `<expr>`

35. `<return_void>` -> return

36. `<return_void>` ->  $\epsilon$

37. `<fnc_body>` -> `<if>``<fnc_body>`

38. `<fnc_body>` -> `<loop>``<fnc_body>`

39. `<fnc_body>` -> `<statement>``<fnc_body>`

40. `<fnc_body>` ->  $\epsilon$

41. `<statement>` -> `<var_dec>`

42. `<statement>` -> id\_var = `<var_assign>`

43. `<statement>` -> `<fnc_call>`

44. `<var_dec>` -> local `<var_def>` `<var_dec_init>`

45. `<var_dec_init>` -> = `<var_dec_init2>`

46. `<var_dec_init>` ->  $\epsilon$

47. `<var_dec_init2>` -> `<expr>`

48. `<var_dec_init2>` -> `<fnc_call>`

49. `<var_assign>` -> `<expr>`

50. `<var_assign>` -> `<fnc_call>`

51. `<if>` -> if `<expr>` then `<statements>` else `<statements>` end

52. `<loop>` -> while `<expr>` do `<statements>` end

53. `<statements>` ->  $\epsilon$

54. `<statements>` -> `<statement>` `<statements>`

POZN.: Slova a znaky psaná černě jsou neterminály.

POZN.2: Znak mezery v pravidlech je irrelevantní a nic neznačí, slouží pouze k zpřehlednění.

POZN.3: Neterminál `<expr>` není rozšířen a zpracuje ho analýza shora dolů.