Chapter 4

Regular Languages and Expressions

A. **Construct a regular expression (RegEx) for each of the following (assuming that the shortest acceptable strings are long enough to accommodate all the requirements).**

1. Binary numbers that are twice an odd number (i.e., binary representations of 2*n* where *n* is an odd number).

   Solution: Anything followed by a 1 (to make the number odd) and then a 0 (to make it twice the odd number):

   $$(0 + 1)^*10$$

2. Binary strings containing the sequence 011.

   Solution: Please see also the VIDEO SOLUTION (which shows an equivalent NFA in addition to the RegEx). Anything can be present on either side of 011:

   $$(0 + 1)^*011(0 + 1)^*$$

3. Binary strings containing at least one 00 and at least one 11.

   Solution: 00 can come first and then 11 or 11 first and then 00:

   $$(0 + 1)^*00(0 + 1)^*11(0 + 1)^* + (0 + 1)^*11(0 + 1)^*00(0 + 1)^*$$

4. Binary strings with at least two occurrences of at least two consecutive 1 s, the two occurrences not being adjacent (i.e., 011011 is acceptable but 011111 is not).

   Solution: There must be at least one 0 between the two occurrences of 11:

   $$(0+1)^*11(0 + 1)^*0(0 + 1)^*11(0 + 1)^*$$

5. Strings over {*a*, *b*, *c*} in which the fourth symbol from the beginning is a *c*.

   Solution: The first three symbols can be anything:

   $$(a + b + c)(a + b + c)(a + b + c)c(a + b + c)^*$$

6. Strings over {*a*, *b*} with an even number of *a* s.

   Solution: Please see also the VIDEO SOLUTION which shows a slightly different but equivalent RegEx. Only *b* s or pairs of *a* s with any number of *b* s in between:

   $$b^* + (b^*ab^*ab^*)^*$$

7. Strings over {*a*, *b*} with at most three *a* s.

Solution: Strings may have no a, one a, two a s or three a s:

$$b^* + b^*ab^* + b^*ab^*ab^* + b^*ab^*ab^*ab^*$$

8. Strings over $\{a, b\}$ of the form $a^{2n}b^{2m}$ where n and m are positive integers.

   Solution: One or more pairs of a s followed by one or more pairs of b s:

   $$aa(aa)^*bb(bb)^*$$

9. Strings over $\{a, b\}$ whose length is divisible by 2 but not by 3.

   Solution: Strings of length 2, 4, 8, 10, 14, 16, etc. are acceptable. These can be obtained by taking strings of length 2 or 4 and concatenating strings of length 6 any number of times:

   $$((a + b)(a + b) + (a + b)(a + b)(a + b)(a + b))((a + b)(a + b)(a + b)(a + b)(a + b)(a + b))^*$$

10. Binary strings such that if they are of even length, then the number they represent is also even and if the length is odd, then the number is also odd.

    Solution: First we obtain strings of even length and concatenate any one symbol to get strings of odd length; then we append a 0 to get strings of even length that also represent even numbers. In the second option, we concatenate a 1 to a string of even length to get strings of odd length that also represent odd numbers:

    $$((0 + 1)(0 + 1))^*(0 + 1)0 + ((0 + 1)(0 + 1))^*1$$

11. Strings over $\{a, b, c\}$ with a single c before which an even number of a s and any number of b s are present in any order; after the c, there must be an odd number of a s and any number of b s in any order.

    Solution: Note that to obtain an odd number of a s in the second part, we must first obtain an even number of a s and then introduce the odd one. We must allow for the possibility of any number of b s anywhere:

    $$(b + (b^*ab^*ab^*))^*c(b + (b^*ab^*ab^*))^*ab^*$$

12. Strings of the form $a^nb^mc^k$, where $n + m$ is odd and k is even.

    Solution: Either n is even and m is odd or n is odd and m is even.

    $$((aa)^*(bb)^*b + (aa)^*a(bb)^*)(cc)^*$$

13. Strings of the form $a^nb^mc^k$, where $n \leq 4$, $m \geq 2$, $k \leq 2$.

    Solution: Note that bbb^* below means two b s followed by zero or more additional b s.

    $$(\lambda + a + aa + aaa + aaaa)bbb^*(\lambda + c + cc)$$

14. Strings in the infinite sequence: aba, a^5, aba^7, a^{11}, aba^{13}, a^{17}, ….

    Solution: Please see also the VIDEO SOLUTION.

    $$(aba + aaaaa)(aaaaaa)^*$$

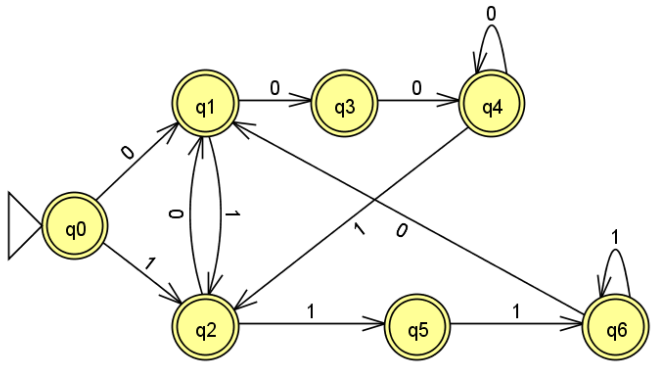
15. Strings over $\{a, b, c\}$ in which at least one a appears in the first three symbols and at least one b in the last four symbols.

    Solution: The first symbols can be a or the second or the third. Similarly for b.

    $$(a(a + b + c)(a + b + c) + (a + b + c)a(a + b + c) + (a + b + c)(a + b + c)a)(a + b + c)^*$$
    $$(b(a + b + c)(a + b + c)(a + b + c) + (a + b + c)b(a + b + c)(a + b + c) +$$
    $$(a + b + c)(a + b + c)b(a + b + c) + (a + b + c)(a + b + c)(a + b + c)b)$$

16. Binary strings in which any run is of length at least 3. A run is a sequence of two or more occurrences of the same symbol. For example, 00011110000 has a run of 0 s of length 3 followed by a run of 1 s of length 4 and a run of 0 s of length 4.

    Solution: There are many possibilities to consider: starting with 0 or 1, ending with 0 or 1, etc. and 01 or 10 and other possible sequences with no runs in between. It is rather difficult to construct this RegEx directly. Instead, let us construct a DFA first:



    The RegEx obtained by converting from the equivalent DFA is(!):

    λ+0+(1+01)(01)*(λ+0)+00+(1+01)(01)*00+(00+(1+01)(01)*00)0(0+1(01)*000)*(λ+1(01)*(λ+0)+1(01)*00)+(1+01)(01)*1+(00+(1+01)(01)*00)0(0+1(01)*000)*1(01)*1+((1+01)(01)*1+(00+(1+01)(01)*00)0(0+1(01)*000)*1(01)*1)1(1+(01(01)*1+(00+01(01)*00)0(0+1(01)*000)*1(01)*1)1)*(λ+0+01(01)*(λ+0)+00+01(01)*00+(00+01(01)*00)0(0+1(01)*000)*(λ+1(01)*(λ+0)+1(01)*00)+01(01)*1+(00+01(01)*00)0(0+1(01)*000)*1(01)*1)

17. All valid dates in the year 2012 expressed in the dd/mm format.

    Solution: Minor changes from Example 4.18 to change the year to 2012 and to include February 29:

(0(1-9)|(1-2)(0-9)|30)/(0(4|6|9)|11)|

(0(1-9)|(1-2)(0-9)|3(0-1))/(0(1|3|5|7|8)|1(0|2))|

(0(1-9)|1(0-9)|2(0-9))/02

18. All valid telephone numbers: just a 7-digit number (e.g., 26721983), or a 10-digit number (e.g., 9900011111), or a 3-digit area code followed by a space and a 7-digit number (e.g., 080 26721983), or a plus sign followed by a 2-digit country code, a space and either a 10-digit number (e.g., +91 9900011111) or a 3-digit area code, a space and a 7-digit number (e.g., +91 80 26721983). Note that of these, only the three-digit area code can begin with a 0 in its first digit.

    Solution: Please note minor corrections to the problem specification: 8 digits (not 7 as mentioned in the problem) and two more optional digits for the first two kinds, and so on. Also, area codes are 2 digits, not 3 as mentioned. Country codes can begin with a 0.

    (1-9)(0-9)(0-9)(0-9)(0-9)(0-9)(0-9)(0-9)(λ + (0-9)(0-9)) +
    (1-9)(0-9)\ (1-9)(0-9)(0-9)(0-9)(0-9)(0-9)(0-9)(0-9) +
    \+(0-9)(0-9)\ (1-9)(0-9)(0-9)(0-9)(0-9)(0-9)(0-9)(0-9)(0-9)(0-9) +
    \+(0-9)(0-9)\ (1-9)(0-9)\ (1-9)(0-9)(0-9)(0-9)(0-9)(0-9)(0-9)

19. All valid names of people: a first name and an optional last name and any number of middle names or middle initials (e.g., James Bond or James H. H. E. Bond); or any number of initials followed by a single name (e.g., J. H. H. E. Bond). First name, middle name and last name are all in init-caps (i.e., only the first letter is capitalized); initials are a capital letter followed by a period.

    Solution: First type for first name and other optional parts; second type for optional initials and a single name.

    (*A*-*Z*)(*a*-*z*)*(\ (*A*-*Z*)(\. + (*a*-*z*)(*a*-*z*)*))*(λ + \ (*A*-*Z*)(*a*-*z*)*) +
    ((*A*-*Z*)\.\ )*(*A*-*Z*)(*a*-*z*)*

20. All valid monetary amounts: starting with the Rupee symbol with an optional decimal point and a two-digit fractional amount; if it is larger than three digits, then commas separate billions, millions and thousands (e.g., Rs. 101.99 or Rs. 10,000 or Rs. 1,670,439,444.49).

    Solution: Note that the Rupee symbol is shown as R below. Amounts can have just one digit, two digits, three digits or more. If there are more than 3 digits, commas are introduced as shown below.

    R\ ((1-9) + (1-9)(0-9) + (1-9)(0-9)(0-9))(,(0-9)(0-9)(0-9))*(λ + \.(0-9)(0-9))

21. All valid GPS coordinates: latitude and longitude expressed in degrees and minutes in the DDD MM.MMMM format with N, S, E, W to indicate North or South of the equator for latitudes and East or West of the Greenwich Meridian for longitudes. Note that latitudes range from 0 to 90 and longitudes from 0 to 180. Leading 0 s may be omitted in both, for example, 12 30.0123 N 77 66.5843 E.

Solution: Note that we cannot allow fractions greater than zero when the whole degree is 90 for latitude or 180 for longitude!

((((0-9) + (1-8)(0-9))\ (0-9)(0-9)\.(0-9)(0-9)(0-9)(0-9)) + 90 00.0000)\ (*N* + *S*)
((((0-9) + (1-9)(0-9) + 1(0-7)(0-9))\ (0-9)(0-9)\.(0-9)(0-9)(0-9)(0-9)) + 180 00.0000)
\ (*E* + *W*)

B. For the following, describe the language of the RegEx as concisely as possible.

22. 1(0 + 1)(0 + 1)(0 + 1)(0 + 1)*0

Solution: Binary strings that begin with 1 and end with 0 and are of length at least five OR even numbers greater than or equal to 16.

23. $(b + \lambda)(a\ (a + \lambda)^*(b + \lambda))^*(a + \lambda)^*$

Solution: Note that (*a* + *λ*)* is the same as *a**. Strings can start with *a* or *b*. Any occurrence of *b* except the first one must be preceded by at least one *a*. Strings can also end with *a* or *b*. As such, this is the language of all strings with no two consecutive *b* s.

24. $(ab + ba)^*$

Solution: Strings of even length with no run of length more than two.

25. $(aa)^*(bb)^*(b + \lambda)$

Solution: Even number of *a* s followed by odd or even number of *b* s.

26. (*A*-*Z*)(*a*-*z*)*(*a* + *e* + *i* + *o* + *u*)

Solution: Words that begin with a capital letter and end with a vowel.

27. (*a*-*z*)(*a*-*z*|0-9)*

Solution: Strings beginning with a lowercase letter followed by any number of lowercase letters or digits (i.e., alphanumeric strings in which the first letter must be an alphabet).

28. (1-9)(0-9)(0-9)?\.(1-9)(0-9)(0-9)?\.(1-9)(0-9)?\.(1-9)(0-9)?

Solution: Please note that each backslash is to be followed by a period which is missing in the print. These strings denote IP addresses.

C. Simplify the following RegEx's:

29. $(\alpha + \beta)^* \beta\alpha(\beta + \alpha)^* + (\beta + \alpha)^* \beta\beta(\alpha + \beta)^*$

Solution:

$(\alpha + \beta)^* \beta(\alpha + \beta)(\alpha + \beta)^*$

30. $00^*(0 + \lambda)(1 + \lambda)(1 + \lambda)(1 + \lambda)^*$

Solution:

00*1*

31. $(00 + 11 + 01 + 10)^*$

Solution:

((0 + 1)(0 + 1))*

32. $abc + a(c + b)(c + b) + (b + c)a(c + b) + (c + b)(c + b)a + (b + c)(b + c)(b + c)$

Solution:

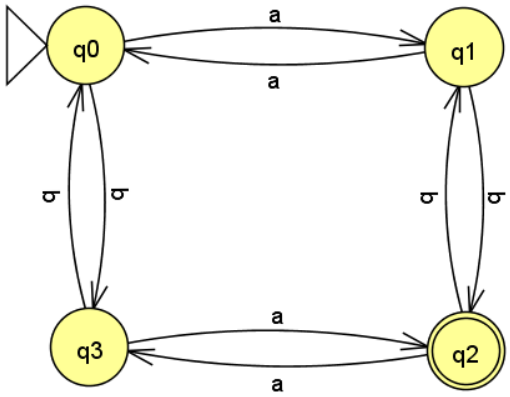
$a(b + c)(b + c) + (b + c)(a + b + c)(b + c) + (b + c)(b + c)a$

D. **For the following, construct a finite automaton (DFA or NFA) first and then convert it to a RegEx:**

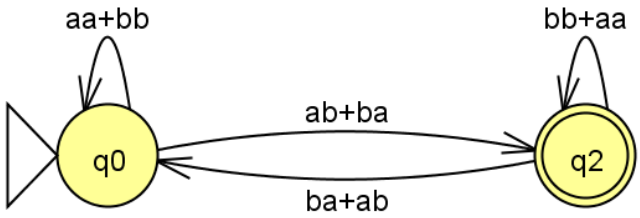
33. Strings over {*a*, *b*} with an odd number of *a* s and an odd number of *b* s.

Solution: RegEx obtained from the following DFA:

$((aa + bb)^*(ab + ba)(bb + aa)^*(ba + ab))^*(aa + bb)^*(ab + ba)(bb + aa)^*$



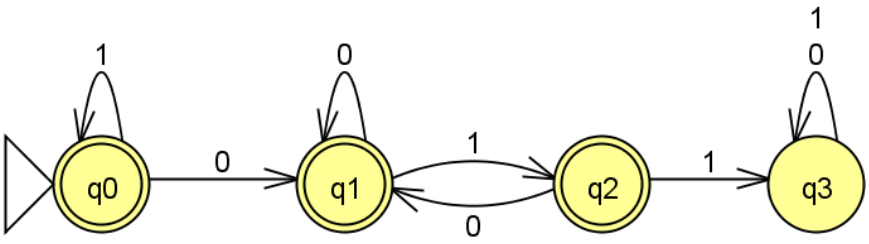
The generalized transition graph obtained in converting to RegEx is:



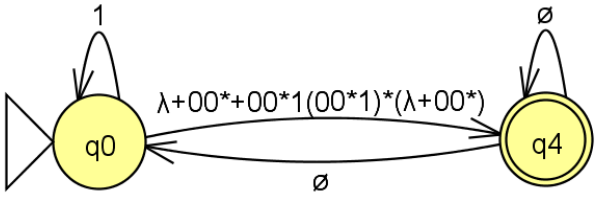
34. Binary strings not containing the keyword 011.

Solution: RegEx obtained from the following DFA:

$$1^*(\lambda + 00^* + 00^*1(00^*1)^*(\lambda + 00^*))$$



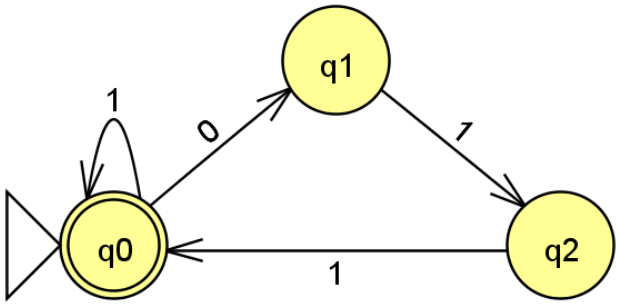
with a corresponding generalized transition graph:



35. Binary strings in which every 0 is followed by 11.

Solution: Please see also the VIDEO SOLUTION.

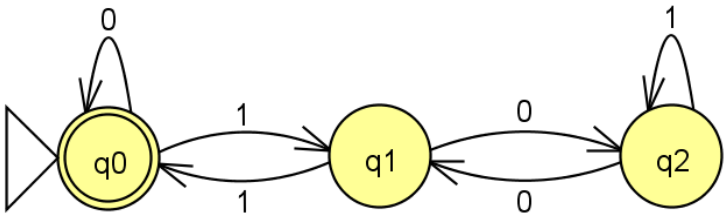
$$(1 + 011)^*$$



36. Binary strings representing positive integers divisible by 3.

Solution:

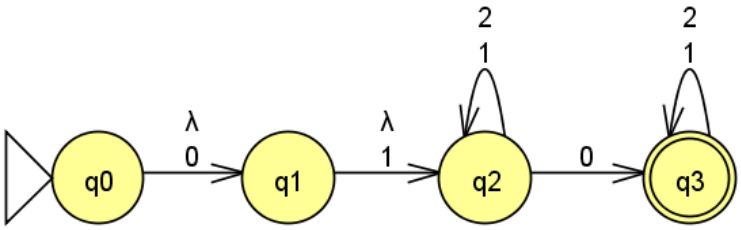
$$(0 + 11 + 10(1 + 00)^*01)^*$$



E. Convert the given RegEx to an equivalent NFA:

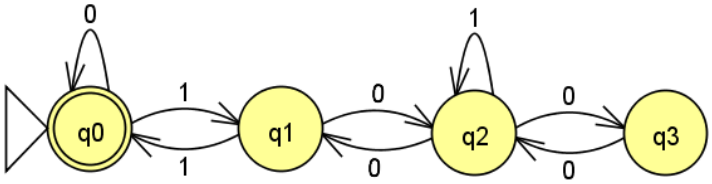
37. $(0 + \lambda)(1 + \lambda)(1 + 2)^*0(2 + 1)^*$

Solution:



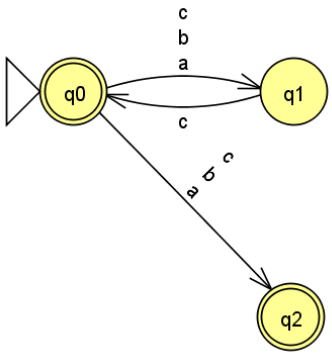
38. $(0 + 11 + 10(1 + 00)^*01)^*$

Solution:



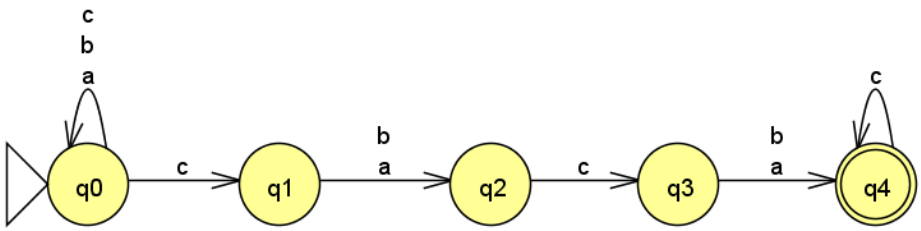
39. $((a + b + c)c)^*(a + b + c + \lambda)$

Solution:



40. $(a + b + c)^*c(a + b)c(a + b)(c + \lambda)^*$

Solution:



F. Convert the given DFA/NFA to an equivalent RegEx.

41. DFA shown in Fig. 2.5 for even binary numbers without leadings 0 s.

Solution:

$$0 + 11*0(0 + 11*0)*$$

42. DFA shown in Fig. 2.12 for same symbols in all odd positions.

Solution:

$$a((a + b)a)*(a + b + \lambda) + b((a + b)b)*(a + b + \lambda)$$

or, equivalently (as given by JFLAP),

$$a + a(a + b)(a(a + b))*(\lambda + a) + b + b(b + a)(b(b + a))*(\lambda + b)$$

43. NFA shown in Fig. 3.5 to search for a keyword.

Solution:

$$(0 + 1)*101(0 + 1)*$$

44. NFA shown in Fig. 3.9 for certain binary strings.

Solution:

$$((0 + 01)1)*01$$

45. NFA shown in Fig. 3.11 for binary numbers divisible by either 4 or 6.

Solution:

$$0 + 1(0 + 1)*00 + 1(01*0 + 10*1)*1(0 + 1(01*0)1)*0$$

or, equivalently (as given by JFLAP),

$$0 + 1(0 + 1)*00 + (11 + 10(1 + 00)*01)(0 + 11 + 10(1 + 00)*01)*0$$

46. DFA shown in Fig. 3.15 for strings with 0, 1 and 2 in that order.

Solution:

$$0*1*2*$$

or, equivalently (but less minimally, as given by JFLAP),

$$0^*(\lambda + 11^* + (2 + 11^*2)2^*)$$

G. Are the following pairs of RegEx's equivalent (i.e., do they represent the same set of strings)?

47. $(0 + \lambda)(11^*0)^*(1 + \lambda)$ and $(1 + \lambda)(011^*)^*(0 + \lambda)$

Solution: Yes, both can start with 0 or 1; both can end with 0 or 1; both do not allow consecutive 0 s. Both represent the language of all binary strings without consecutive 0 s (except 11* which neither of them include). The first RegEx excludes consecutive 0 s by requiring every 0 except the first one to be preceded by a 1; the second RegEx achieves the same result by requiring every 0 except the last one to be followed by a 1.

48. $(0^*1^*)^*$ and $(0 + 1)^*$

Solution: Yes. In both cases we can repeat something zero or more times; what we repeat, in both cases, can be either 0 or 1 each time.

49. $(1 + \lambda)\ (00^*1)^*0^*$ and $(0 + \lambda)\ (11^*0)1^*$

Solution: No. The first one matches 00, for example, which the second one does not match.

50. $(0 + 1)^*(0 + \lambda)$ and $(1 + \lambda)(1 + 0)^*(0 + 1 + \lambda)$

Solution: Yes, both are just (0 + 1)*

51. Can a RegEx with only union and concatenation operators (i.e., with no * closure) represent an infinite number of strings? Explain.

Answer: No, it cannot, because such a RegEx is itself, by definition, of finite length.

H. Debug and correct the following incorrect RegEx's:

52. Binary strings with a 0 in every third position: 110(0 + 1)*.

Solution: There can never be a * closure when counting positions is important. The correct solution must also consider strings whose length is not divisible by 3 (hence the last two parts below):

$$((0 + 1)(0 + 1)0)^*(0 + 1 + \lambda)(0 + 1 + \lambda)$$

53. Binary strings alternating 0 s and 1 s starting with either a 0 or a 1: (01 + 10)*.

Solution: The given RegEx is incorrect since it allows 0110, for example. The error is the possibility of mixing of the two branches for 01 and 10. Instead, we can use only 01 and add corrections for beginning and ending parts:

$$(1 + \lambda)(01)^*(0 + \lambda)$$

54. Student letter grades represented as strings made up of the symbols {*A*, *B*, *C*, *D*, *F*} with no more than two *F* grades:

$$(A + B + C + D)^*F(A + B + C + D)^*F$$

Solution: This assumes that every student must obtain two F grades! The correct RegEx is:

$$(A + B + C + D)^*(F + \lambda)(A + B + C + D)^*(F + \lambda)(A + B + C + D)^*$$

55. Time in 24 hour hh:mm:ss format:

$$(1 + 2 + 0)(0\text{-}9):(0\text{-}5)(0\text{-}9):(0\text{-}5)(0\text{-}9)$$

Solution: This allows hours to run till 29! The correct RegEx is:

$$((2(0\text{-}3)) + (1 + 0)(0\text{-}9)):(0\text{-}5)(0\text{-}9):(0\text{-}5)(0\text{-}9)$$

56. Radio stations from AM 535.0 kHz to AM 1605.0 kHz and FM 88.0 MHz to FM 108.0 MHz:

$$(A + F)M\ (1 + \lambda)(0\text{-}9)(0\text{-}9)(0\text{-}9).(0\text{-}9)\ (k + M)Hz$$

Solution: This allows too many invalid strings by trying to combine the parts for AM and FM. It is best to keep them in separate branches:

$$AM\backslash\ (((53(5\text{-}9) + 5(4\text{-}9)(0\text{-}9) + (6\text{-}9)(0\text{-}9)(0\text{-}9) + 1(0\text{-}5)(0\text{-}9)(0\text{-}9) + 160(0\text{-}4)).(0\text{-}9)) + 1605.0)\backslash\ kHz +$$
$$FM\backslash\ (((8(8\text{-}9) + 9(0\text{-}9) + 10(0\text{-}7)).(0\text{-}9)) + 108.0)\backslash\ MHz$$