* Polynomials
  + Tests:
    1. Using Newtons difference method to figure out the possible degree k of a fitting polynomial. Then try to fit a polynomial to the first k+1 elements of the sequence. Calculate the error between the true values of the sequence and the “predicted” values from the polynomial. When this error is 0, we conclude this test is passed, otherwise it is failed. Special labels (3) are assigned if the length of the sequence is too short for a prediction on datapoints that were not used for fitting.
    2. Checking if the sequences has the keyword “fini” or “dead”. If so, this test is passed otherwise it is failed.
  + Conclusion:
    1. When Test 2 is passed we give a final rating of 0
    2. When Test 1 yields the special label 3 (sequence too short) we give a final rating of 1
    3. When Test 1 is passed we give a final rating of 4
    4. When Test 1 is failed we give a final rating of 0
* Exponentials
  + Tests:
    1. First when the sequence has keywords “fini” or “dead” we assign the special label 5 to this test. Otherwise we calculate the quotients of subsequent values in the sequence. This yields the new sequence “q”. When the sequence contains any zero-value, except at the first position, this test yields the special label “6” (because we can’t divide by 0). During this calculation an overflow error can occur. Such cases get the special label “7” in this test. q is then checked for monotonicity (increasing or decreasing). We then check the last 30 elements of q are all the same but not equal to 1 (label 1). Otherwise we check whether the last 30 elements of q are all within 10e-5 of the last element, in which case we conclude that it is possible that the sequence converges slowly. Thus, giving the label 2. Sequence that converge to 1 get the label 8. Sequences that are too short (i.e. don’t have 30 elements) get the label 3. Everything else is labelled with a 0.
    2. When the name of the sequence contains a string pattern of the form “a(n) = \*^(\*n\*).” (where \* marks anything) the test is passed. Otherwise it is failed.
    3. When the formulas of the sequence contain a string pattern of the form “a(n) = \*^(\*n\*).” (where \* marks anything) the test is passed. Otherwise it is failed. When no formulas are in the database the special label 4 is assigned.
    4. When the mathematica\_programs of the sequence contain a string pattern of the form "Table[\*^(\*n\*)]" (where \* marks anything) the test is passed. Otherwise it is failed. When no mathematica\_programs are in the database the special label 4 is assigned.
  + Conclusion:
    1. We assign a rating depending on the label of test 1: Labels 0,5 and 8 get a rating of 0. Labels 6 and 7 get a rating of 1. Label 2 gets a result of 2. Label 1 gets a result of 3.
    2. If test 2 is passed, we increase the result from step 1 by 2.
    3. If test 3 is passed, we increase the result from step 2 by 1.
    4. If test 4 is passed, we increase the result from step 3 by 1. (to the limit of 4)
    5. Reduce values bigger than 4 to the limit of 4
* Periodic
  + Tests:
    1. We check whether the values in the sequence are periodic. The test is passed when a period of length smaller than 10000 is found which repeats at least 3 times in the given data. Otherwise, the test is failed.
    2. From test 1 we store the number of repetitions that were found in the given data.
  + Conclusion:
    1. We assign a final rating of 4 if a period is found with at least 30 repetitions.
    2. We assign a final rating of 3 if a period is found with 3 or more repetitions (and less than 30).
    3. We assign a final rating of 1 the sequence does not have enough values given. To reach a conclusion.
    4. Otherwise, we assign a final value of 0.
* Decimal Expansion
  + Tests:
    1. Check whether the sequence name contains the substring: “decimal expansion”. The test is passed if this is the case.
    2. The test is passed when the sequence contains the keyword “cons”. From OEIS: “A [decimal expansion](https://oeis.org/wiki/Category:Decimal_expansions) of a number (occasionally some other base)”
    3. The test is passed when the sequence contains the keyword “base”. From OEIS: “Sequence is [dependent on base](https://oeis.org/wiki/Category:Base-dependent_integer_sequences" \o "Category:Base-dependent integer sequences) used.”
    4. The test is passed when the comments of the sequence contain the substring “decimal expansion”.
  + Conclusion:
    1. Set the initial result to 0 and increase depending on passed tests:
    2. Passing test 1 increases by 4
    3. Passing test 2 increases by 1 (this low value is somehow not very intuitive. However, when checking the usage of the keyword “cons” in the database it seems that it is used quite loosely. For example [A000007](https://oeis.org/A000007) or [A000012](https://oeis.org/A000012) are marked as “cons” but are – in my opinion – not anymore a decimal expansion of a number than any other sequence).
    4. test 3 increases by 1
    5. Passing test 4 increases by 2
    6. When the final result is bigger than 4 reduce it to 4.
* Expansion:
  + Tests:
    1. Passed when the sequence name contains the substring “expansion”.
    2. Passed when the sequence name can be matched by the regular expression: “base\d+” (the word base followed by at least one digit)
    3. Passed when the mathematica\_programs of the sequence contain the substring “RealDigits”.
    4. The test is passed when the sequence contains the keyword “base”. From OEIS: “Sequence is [dependent on base](https://oeis.org/wiki/Category:Base-dependent_integer_sequences" \o "Category:Base-dependent integer sequences) used.”
    5. The test is passed when the sequence contains the keyword “cons”. From OEIS: “A [decimal expansion](https://oeis.org/wiki/Category:Decimal_expansions) of a number (occasionally some other base)”
  + Conclusion:
    1. Set the initial result to 0 and increase depending on passed tests:
    2. Passing test 1 AND test 2 increases by 4.
    3. Passing test 1 and not test 2 increases by 3.
    4. Passing test 4 increases by 1
    5. Passing test 5 increases by 1
    6. Passing test 3 increases by 2
    7. When the final result is bigger than 4 reduce it to 4.
* Fibonacci-Like
  + Tests:
    1. The test is passed when the name of the sequence contains a Fibonacci-like formula as a substring. Those are substrings like: f(n) = f(n-1) + 3\*f(n-2). More accurately anything that is matched by the regular expression:  
         
       regex = c + r"\(n\)=" + R + r"[\+\-]" + R + r"([\+\-]" + R + r")\*"  
       where R = r"[0-9]\*\\*?" + c + r"\(n[\+\-][0-9]+\)"  
       and c is any letter in the alphabet (in the above example “f” would be the letter which would create a match).
    2. The test is passed when the formulas of the sequence contain a Fibonacci-like formula as a substring. See test 1 for more details.
  + Conclusion:
    1. We give a final rating of 4 when test 1 is passed.
    2. We give a final rating of 3 when only test 2 is passed.
* Are palindromes.
  + Tests:
    1. The test is passed when all elements in the sequence are palindromes. Palindromes are numbers that are the same when read from left to right (as usual) and when read from right to left, for example “12321”.
  + Conclusion:
    1. When test 1 is passed a final result of 4 is set.
    2. Otherwise a final result of 0 is given.
* Related to palindromes
  + Tests:
    1. The test is passed when the name of the sequence contains the substring “palindrome”.
    2. The test is passed when the comments of the sequence contains the substring “palindrome”.
    3. The test is passed when the mathematica\_programs of the sequence contains the substring “palindrome”.
    4. The test is passed when the keywords of the sequence contain “base”. This is because obviously the property of being palindromic is not invariant under base changes (except for the numbers 0 and 1).
  + Conclusion:
    1. Set the initial result to 0.
    2. Increase by 4 when test 1 is passed.
    3. Increase by 2 when test 2 is passed.   
       Increase by an additional 1 when additionally, to test 2, test 4 is also passed.
    4. Increase by 2 when test 3 is passed.   
       Increase by an additional 1 when additionally, to test 3, test 4 is also passed.
    5. When the value is bigger than 4 reduce it to 4 to get the final result.
* Are primes
  + Tests:
    1. The test is passed when all elements in the sequence are prime. This however is limited by runtime problems for big numbers. Therefore only number smaller than 2^64 (about 10^19) are checked, when numbers bigger than 2^64 are encountered the special label 8 is set as the result of the test.
  + Conclusion:
    1. A final result of 4 is given when test 1 is passed.
    2. A final result of 2 is given when test 1 yields a label 8.
    3. Otherwise, a final result of 0 is given.
* Related to primes
  + Tests:
    1. The test is passed when the name of the sequence contains the substring “prime”.
    2. The test is passed when the mathematica\_programs of the sequence contains the substring “prime”.
    3. The test is passed when the comments of the sequence contain the substring “prime”.
    4. The test is passed when the formulas of the sequence contain the substring “prime”.
    5. The test is passed when the maple\_programs of the sequence contain the substring “prime”.
    6. The test is passed when the other\_programs of the sequence contain the substring “prime”.
  + Conclusion:
    1. We set the result initially to 0
    2. Passing test 1 increases the result by 4
    3. Passing any of the tests 2-6 increases the result by 1 each.
    4. When the result is bigger than 4 it is reduced to 4 to get the final result.
* Are continued fraction expansions
  + Tests:
    1. The test is passed if the keywords of the sequence contain “cofr”. From OEIS:   
       “A [continued fraction expansion](https://oeis.org/wiki/Category:Continued_fraction_expansions) of a number.”
    2. The test is passed if the name of the sequence contains the substring “continued fraction” but not as part of the substring “continued fraction convergents”.
  + Conclusion:
    1. Passing test 1 (or both) yields a final result of 4.
    2. Passing test 2 but not test 1 yields a final result of 3.
    3. Passing no test yields a final result of 0.
* Related to continued fraction expansions
  + Tests:
    1. The test is passed if the name of the sequence contains the substring “continued fraction” possibly as a substring of “continued fraction convergents”.
    2. The test is passed if the mathemtaica\_programs of the sequence contain the substring “continued fraction”.
  + Conclusion:
    1. Passing test 1 (or both) yields a final result of 4.
    2. Passing test 2 yields a final result of 3.
    3. Passing no tests yields a final result of 0.
* Are Tables/2 Dimensional shapes
  + Tests:
    1. The test is passed if the keywords of the sequence contain “tabl”. From OEIS: “A regular array of numbers, such as [Pascal's triangle](https://oeis.org/wiki/Pascal%27s_triangle), made into a sequence by reading it row by row.”
  + Conclusion:
    1. When test 1 is passed, the final result is set to 4.
    2. Otherwise, the final result is set to 0.
* Related to Tables/2 Dimensional shapes
  + Tests:
    1. The test is passed if the name of the sequence contains one of the substrings “triangular array”, “square array” or “rectangular array”.
    2. The test is passed if the comments of the sequence contains one of the substrings “triangular array”, “square array” or “rectangular array”.
    3. The test is passed if the formulas of the sequence contains one of the substrings “triangular array”, “square array” or “rectangular array”.
  + Conclusion:
    1. Set the initial result to 0.
    2. Passing test 1 increases the result by 4.
    3. Passing test 2 increases the result by 2.
    4. Passing test 3 increases the result by 3.
    5. If the result is bigger than 4, reduce it to 4 to get the final result.
* Rounding
  + Tests:
    1. The test is passed if the name of the sequence contains any of the substrings "rounded", "rounding", "ceiling" or "floor".
    2. The test is passed if the mathematica\_programs of the sequence contains any of the substrings "round", "ceil" or "floor".
    3. The test is passed if the formulas of the sequence contains any of the sequences "round", "ceil" or "floor".
  + Conclusion:
    1. Set the initial result to 0.
    2. Passing test 1 increases the result by 3.
    3. Passing any one of test 2 or 3 increases the result to 4.
* Roots
  + Tests:
    1. The test is passed if the name of the sequence contains the substring “root” but not as part of “rooted” (which belongs the sequences related to trees).
    2. The test is passed if the mathematica\_programs contain the substring “root” but not as part of “rooted”.
    3. The test is passed if the formulas of the substring contain the substring “root” but not as part of “rooted”.
    4. The test is passed if the name contains a substring that gets matched by the following regular expression:  
       (( )|(^))(([a-z]+)|(\d+th))(( )|(-))root  
       Those are strings like “fifth-root”or “7th root” .
    5. The test is passed if the name contains a substring that gets matched by the following regular expression:  
       ((\^\((n\/\d+)\))|(\^\((\d+\/\d+)\)))  
       Those are strings like “^(5/3)” or “^(n/7)”.
    6. The test is passed if the mathematica\_programs of the sequence contain a substring that is matched by the regular expression in test 5.
    7. The test is passed if the formulas of the sequence contain a substring that is matched by the regular expression in test 5.
  + Conclusion:
    1. Set the result initially to 0.
    2. If any of the above tests are passed set the result to 1.
    3. Additionally, increase the result by 1 for each test passed (except test 4, which can only be passed if also test 1 is passed).
    4. If the result is larger than 4, reduce it to 4 to get the final result.
* Sum of Sequences
  + Tests:
    1. The test is passed if the name of the sequence contains a substring that gets matched by the regular expression:  
       "a\(n\)=(?:\d+\\*)\*A\d{6}\(n(?:[\+\-]\d+)?\)([\+\-](?:\d+\\*)\*A\d{6}\(n(?:[\+\-]\d+)?\))+"  
       This matches strings like:   
       "a(n)=15\*A052856(n+5)-423\*A052856(n-21)+11\*A123456(n)"
    2. The test is passed if the formulas of the sequences contains a substring that gets matched by the regular expression from test 1.
  + Conclusion:
    1. If any of the two tests is passed we set a final result to 4.
    2. Otherwise, we set a final result of 0.
* Sum/Product/Difference/Quotient of a Sequence with a constant number
  + Tests:
    1. The test is passed if the name of the sequence c ontains a substring that gets matched by the regular expression:  
       "a\(n\)=(?:\d+\\*)\*A\d{6}\(n(?:[\+\-]\d+)?\)[\+\-\/\\*]\d+"  
       This matches strings like:  
       "A123456(n+1)-12" or "a(n)=A123456(n)/2"
    2. The test is passed if the formula of the sequence contains a substring that gets matched by the regular expression of test 1.
  + Conclusion:
    1. If any of the two tests is passed we the final result to 4.
    2. Otherwise, we set the final result to 0.