A. Some challenges include substring out of range, realizing that digits in a string are not the same as integers, checking for control may reach end of non-void function, and testing border cases and determining ranges for the iterations in loops.

B. hasCorrectSyntax

String song takes user input

String can be empty, or it must end with a ‘/’

Wrong syntax if user input has invalid characters.

After a letter, there can only be up to two digits in a row or slashes that follow.

Only letters and slashes may follow slashes

Digits may not start off in a string

Max of 2 digits in a row, and the next character must be a slash.

translateSong

return 1 if song does not have correct syntax

return 2 and set badBeat to the beat of the song, or position of the string, if the digit is in the form 00, 01, 0, or 1

if there are 2 digits, digit value is (first digit – ‘0’)\*10 + (second digit –‘0’)

if just 1 digit, digit value is (first digit – ‘0’)

return 3 if digits follow a letter, and if the following dashes are less than the digit value and the character after the dashes is another letter; then set badBeat to string number where the letter shows

return 4 if digits follow a letter, and if the next characters are only slashes and there are less slashes than the digit value

if slash follows letter without digit, add lower case of the letter to the empty string

if there are at least as many immediate slashes as digit value, keep adding upper case of letter to empty string until total added equals digit value

if song is translatable, set instructions equal to instructions, and do not change badBeat.

C.

/3g/, return 1, (testing syntax)

G20/, return 4, badBeat=2 (testing premature cases in which first digit was greater than 1)

g10/, return 4, badBeat=2 (premature for cases where first digit was 1)

1/, return 1, (testing syntax)

g05///, return 4, badBeat = 4 (testing premature cases with 2 digits less than 10 and proper badBeat)

G01/, return 2, badBeat=1(testing for sustained beats less than 2)

g01/, return 2, badBeat=2(testing whether upper or lower case mattered)

///g1/, return 2, badBeat= 4 (testing if putting slashes in front mattered)

g5//, return 4, badBeat=3 (testing single digits without 0 in front)

g05//g/////////, return 3, badBeat=3 (testing large song with beat during sustained note)

g09////g/, return 3, badBeat=5 (testing small songs but with sustained notes and another beat)

g11////g//////////////////, return 3, badBeat=5(testing double digits with 1 in front)

g40////, return 4, badBeat = 5 (testing double digit, first digit bigger than 1, with premature ending)

g20////g/, return 3, badBeat = 5(double digit, premature case)

g5/g/, return 3, badBeat = 2 (single digit w/o 0, sustained beat with another beat)

g4//, return 4, badBeat = 3(single digit w/o 0, premature case)

g7//g/, return 3, badBeat = 3(single digit w/o 0, sustained beat with another)

//, instructions = xx, (just dashes to test if instructions printed properly)

g03/////, instructions = GGGxx (double digit less than 10, translatable)

g/g/, instructions = gg (translatable singles)

g//g/, instructions = gxg (translatable single with extra slashes)

g/G/g/b/, instructions = gggb (translatable single with an upper case)

g10///////////, instructions = GGGGGGGGGGx (double digit with 1 in front, translatable)

g4/////, instructions = GGGGx (single digit, translatable)

r3//y/b0//o2/, badBeat = 3, return = 3 (multiple errors)

r3//b0/, badBeat 3, return = 3(multiple errors)

/g3///, instructions = xGGG (check if slashes changed anything)

g0/, return 2, badBeat = 1 (testing sustained beat less than 2)

/gg/, return 1, (syntax)

///////g3, return 1, (syntax)

g/, instructions = g and B/, instructions = b (testing single beat cases with upper and lower cases)

/, instructions = x (testing single beat with only slash)

All cases for FAQ#4 passed. Tested because Smallberg gave examples.