**Question 1**

The big O asympototic runtime complexity of cutAndSplice when using StringStrand is . This is because:

The length of fragments is and the length of splicee is , so the total length would be these two combined, which is . Then looking at the append() method in StringStrand, the runtime complexity can be calculated as in general terms, when is the number of times the append is called and is the length of the input string. Since is the number of fragments, we can substitute this into . Since is the total length of the string, we can substitute this into . Hence, the final big O asymptotic runtime complexity is .

The empirical data from running this code is as follows:

dna length = 320,160

cutting at enzyme gaattc

----------------------------------------------------------------------

Class dna,N splicee,S recomb time(ms) breaks,b

----------------------------------------------------------------------

StringStra: 320,160 10,000 769,890 29 45

StringStra: 320,160 20,000 1,219,890 32 45

StringStra: 320,160 40,000 2,119,890 30 45

StringStra: 320,160 80,000 3,919,890 47 45

StringStra: 320,160 160,000 7,519,890 81 45

StringStra: 320,160 320,000 14,719,890 156 45

StringStra: 320,160 640,000 29,119,890 308 45

StringStra: 320,160 1,280,000 57,919,890 650 45

StringStra: 320,160 10,000 769,890 28 45

StringStra: 640,320 10,000 1,539,780 37 90

StringStra: 1,280,640 10,000 3,079,560 137 180

StringStra: 2,561,280 10,000 6,159,120 544 360

StringStra: 5,122,560 10,000 12,318,240 2,137 720

**Question 2**

The big O asymptotic runtime complexity of cutAndSplice when using StringBuilderStrand is . This is because:

The length of fragments is and the length of splicee is , so the total length would be these two combined, which is . Then looking at the append() method in StringBuilderStrand, the runtime complexity is linear this time because the for loop simply runs for times in general terms, when is the number of fragments. This runtime complexity should then be multiplied by , when is the length of the input string. Since is the total length of the string, we can substitute this into . Hence, multiplying these two, the final big O asymptotic runtime complexity is , which can be written as .

The empirical data from running this code is as follows:

dna length = 320,160

cutting at enzyme gaattc

----------------------------------------------------------------------

Class dna,N splicee,S recomb time(ms) breaks,b

----------------------------------------------------------------------

StringStra: 320,160 10,000 769,890 24 45

StringStra: 320,160 20,000 1,219,890 29 45

StringStra: 320,160 40,000 2,119,890 28 45

StringStra: 320,160 80,000 3,919,890 43 45

StringStra: 320,160 160,000 7,519,890 86 45

StringStra: 320,160 320,000 14,719,890 160 45

StringStra: 320,160 640,000 29,119,890 297 45

StringStra: 320,160 1,280,000 57,919,890 610 45

StringStra: 320,160 10,000 769,890 12 45

StringStra: 640,320 10,000 1,539,780 39 90

StringStra: 1,280,640 10,000 3,079,560 146 180

StringStra: 2,561,280 10,000 6,159,120 551 360

StringStra: 5,122,560 10,000 12,318,240 2,089 720

**Question 3**

Since each character of the String takes 1 byte of memory to store, the total memory required to store the result of the cutAndSplice operation on the StringStrand object would be equal to the number of characters. The number of characters in StringStrand object is because the for loop in cutAndSplice is iterated times, and the splicee length is added each time, leading to . is added to this because it is the total number of bytes from search.split. Therefore, bytes of memory would be needed to store the result on the StringStrand object. Using a StringBuilderStrand,

**Question 4**

toString() is still constant in LinkStrand

**Question 5**