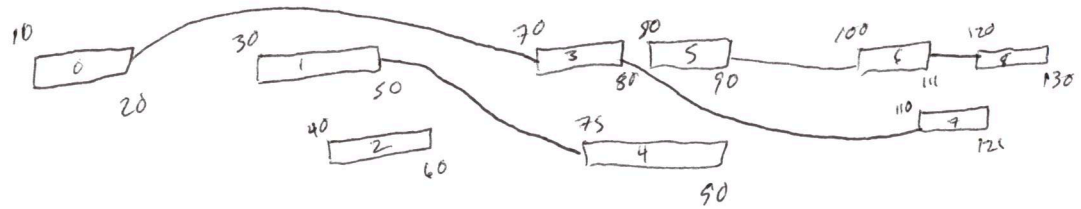
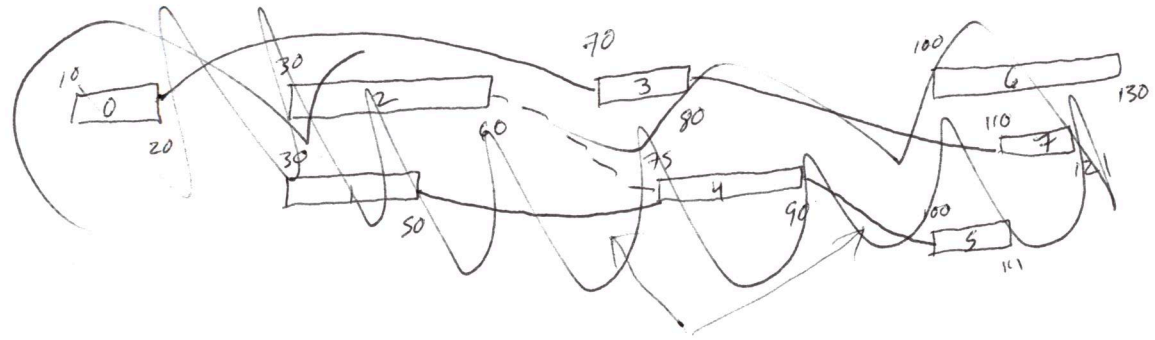
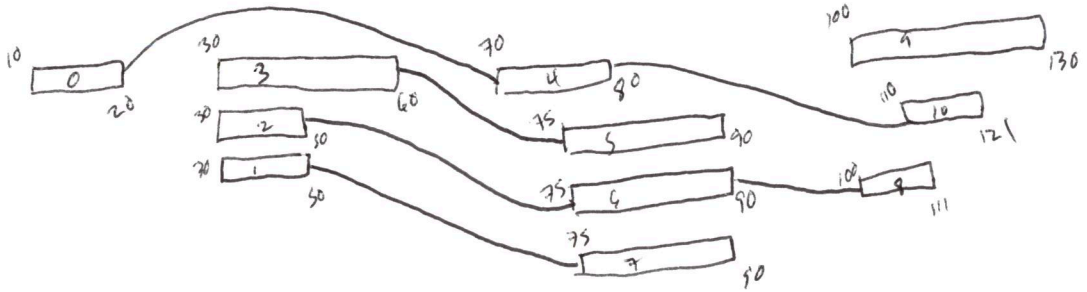
A:



B:



2 isoforms



transcripts: exons

A.1: 0, 3, 7

A.2 1, 4

A.3 2

A.4 5, 6, 8

B.1 ~~0, 3, 7~~ 0, 4, 10

B.2: 1, ~~8~~ 7

B.3: ~~1, 4, 5~~ 2, 6, 8

B.4: ~~2, 4~~ 3, 5

B.5 ~~8~~ 9

get exact matches

input (orfs_1, orfs_2, exons_1), exons_2

~

get exons of the orfs

find all exact matches ~ this is a list of

(a_info, b_info)

{orf_num, exon_num } { orf_num, exon_num

for each orf,
check if all exons are covered

sort ~~by~~

group by ~~a_info, b_info~~
a_orf, b_orf

1, 1

q: o_1a, e_1a, o_1b_e1b
o_1a, e_2a, o1b-e2b

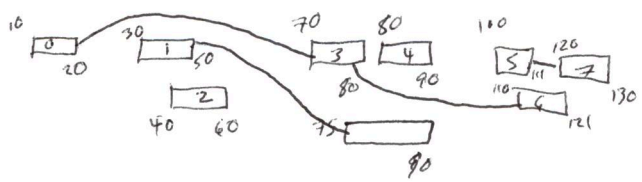
group by orf_num, orf_num

o_1a,

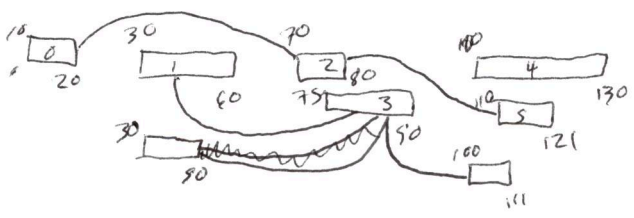
given

A.

A:

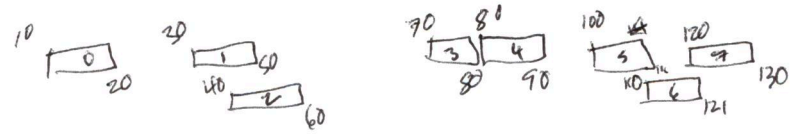


B:

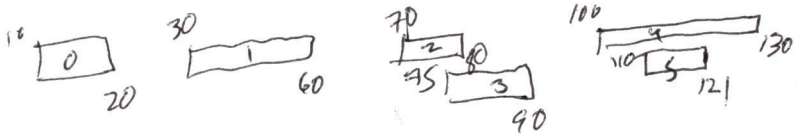


Find exact matches

A:



B:



a, b

matches: (0,0) (3,2) (6,5)

a_cache, b_cache

always favor B

~~add~~ do

process next A interval:

check if A interval exactly matches anything in B_cache

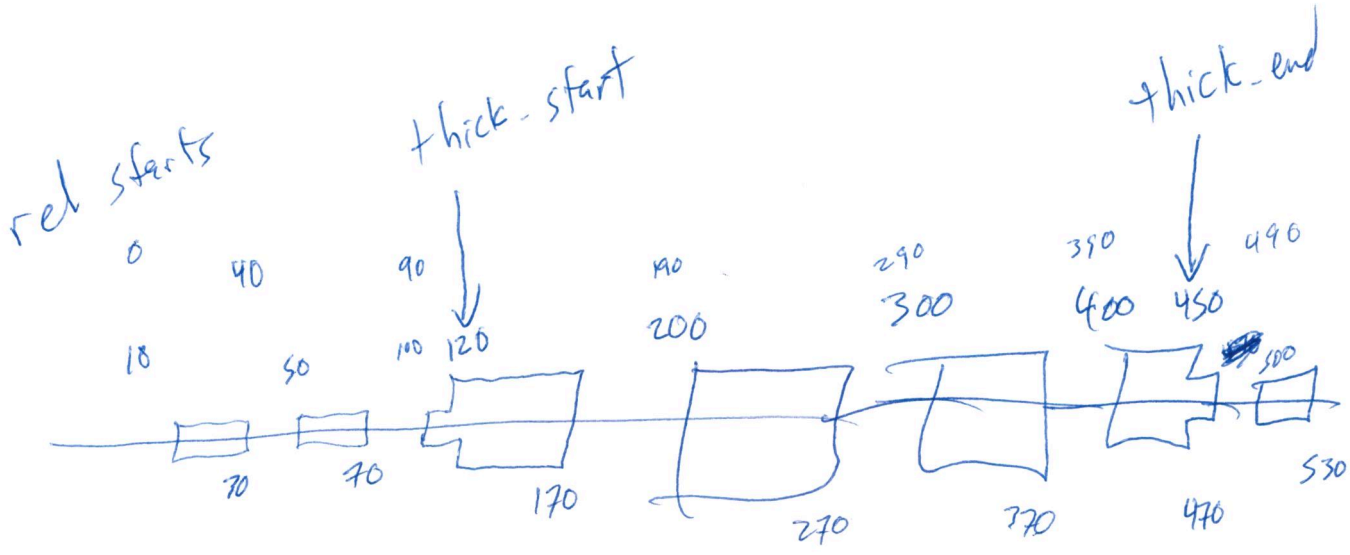
if so, add to matches

remove anything in B cache sf.

B.start < a(start)

test

start thick_start rel_starts lengths



lengths 20 20 70 70 70 70 30

rel starts

expected output 0 80 180 280

lengths

50 70 70 50

return before thick

expected output

rel starts 0 40 90 start 10

lengths 20 20 20 end 120

after thick

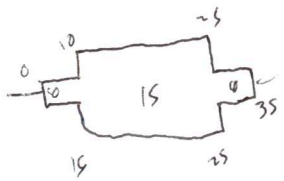
rel starts 0, 50

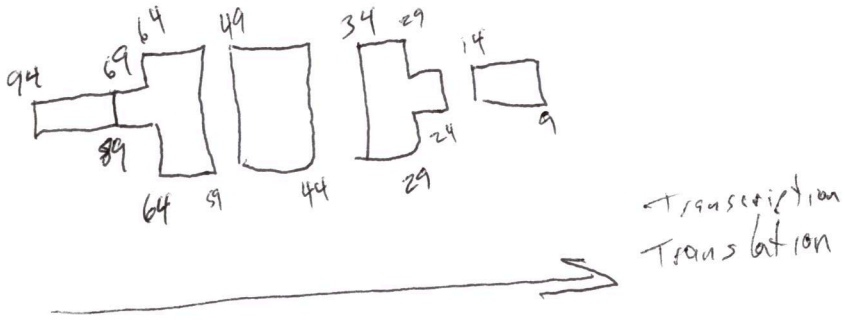
lengths 20, 30

start 450

end 530

genome coordinates





Sequence

but... a relative pos is actually from the End!

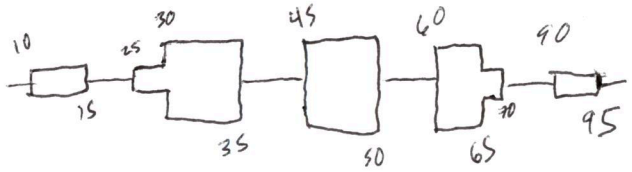
So, to convert rel pos to gen pos, first obtain the genomic start and I just extract it as thick ✓

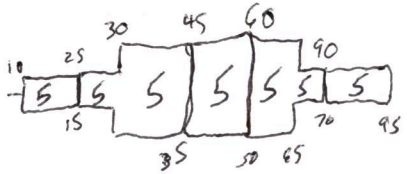
If on the reverse strand... the rel coordinates are still fine

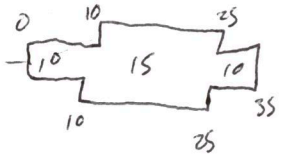
But, for rev strand we have to adjust rel pos first

given rel pos, find its genomic pos

Transcript







Frame

0 30s 45s End [65,68)

1

2

S (10,13) E (25,28) → can

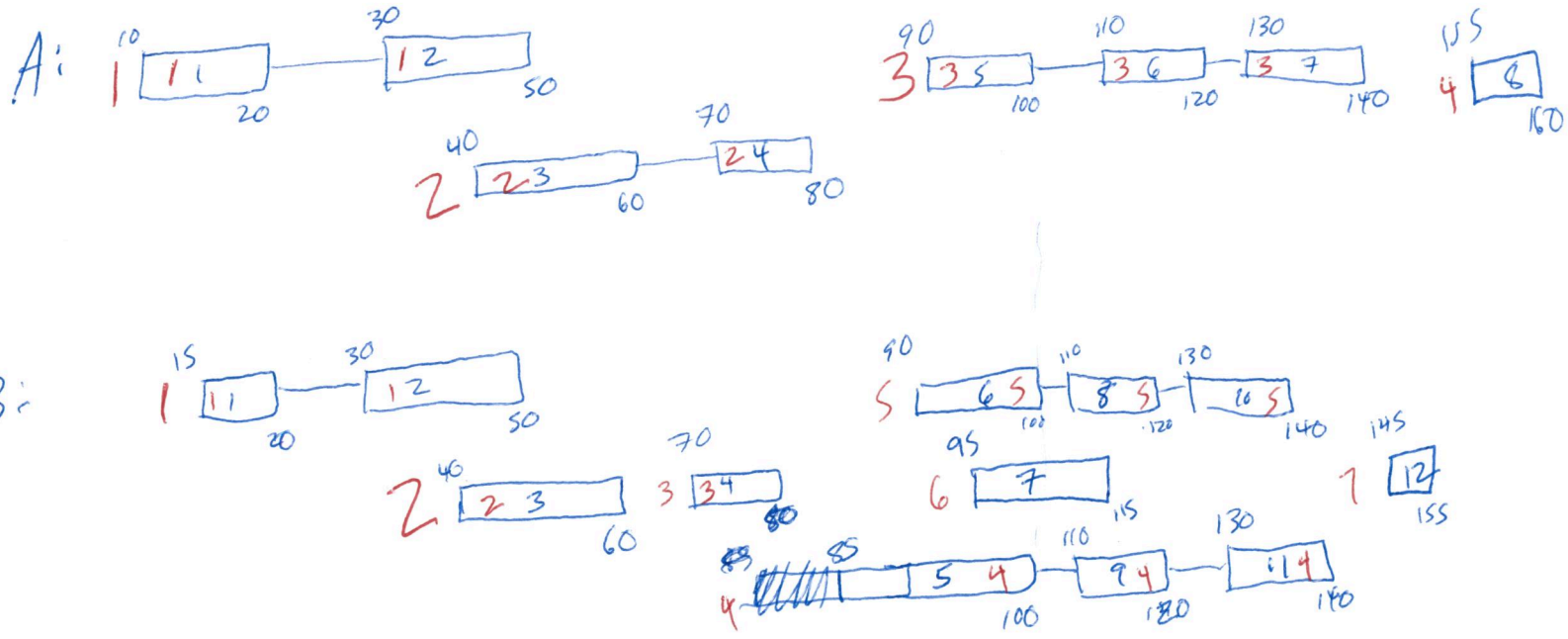
ATG [16,19) [25,28)

TGA

TAA [8,11) ATG [24,27) TAG [30,33)

[2,5)

rel_pos	gen_pos
2	12
8	24
10	30
16	46
24	64
25	65
29	69
30	90



Expected Output

(1,1) : 5
(2,2) : 20
(2,3) : 10
(3,2) : 10
(3,3) : 20
(4,4) : 10

(5,5) : 10
(5,6) : 10
(5,7) : 5
(6,7) : 5
(6,8) : 10
(6,9) : 10
(7,10) : 10
(7,11) : 10

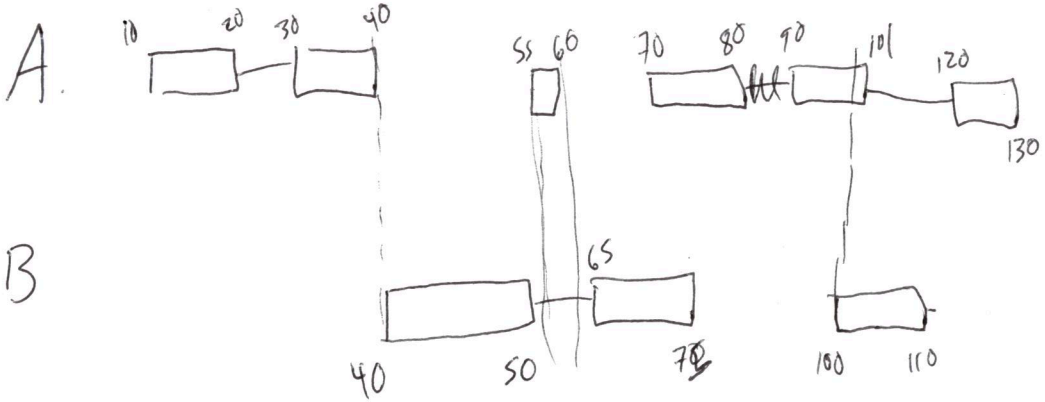
Expected transcript

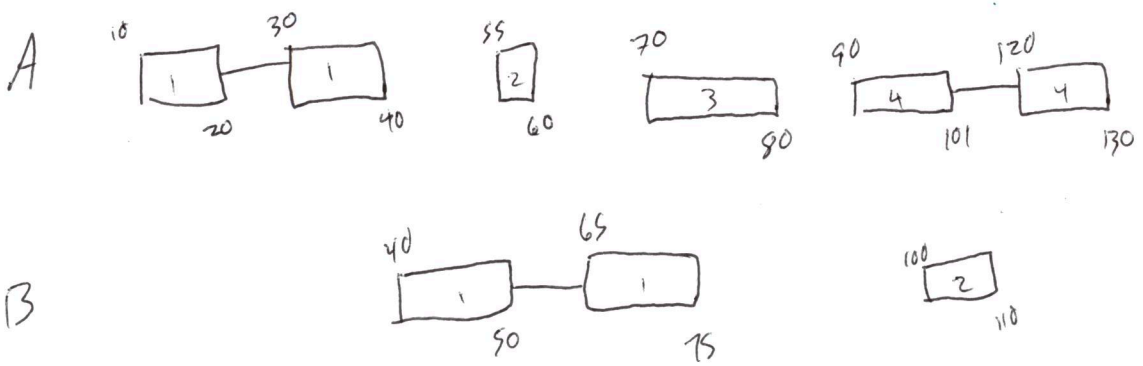
Over laps

(1,1) : 25
(1,2) : 10
(2,1) : 10
(2,2) : 20
(2,3) : 10
(3,4) : 30
(3,5) : 30
(3,6) : 10

Expected fractions

	A	B
(1,1) :	~~.333~~	1.0
(1,2)	~~.33~~	.5
(2,1)	.33	.4
(2,2)	.67	1.0
(2,3)	.33	1.0
(3,4)	1.0	.857
(3,5)	1.0	1.0
(3,6)	.33	.5





expected output:

1, 2

min_overlap = 2 (20-90)

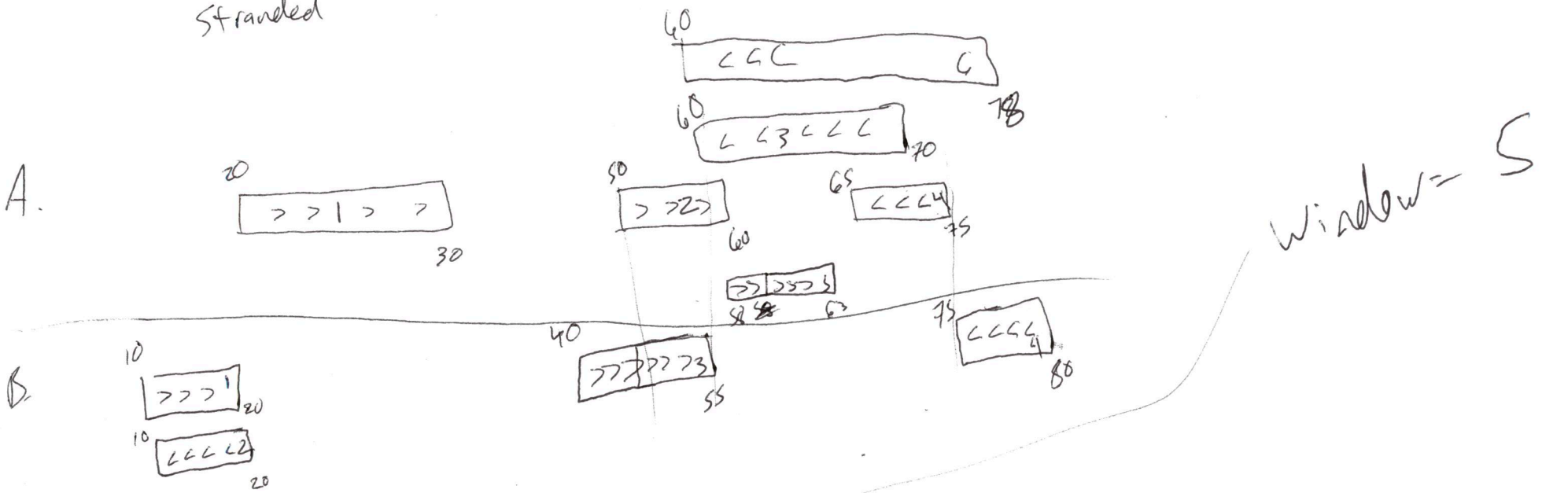
expected: 1, 2, 4

min_b_overlap = 0.3

expected: 1, 2, 3, 4

Find intersections upstream (only)

stranded



expected output [allow overlaps]

A B

(1 1)

~~(3, 4)~~

(4,4)

(5,7)

allow overlaps

for

(1,1)

(2,3)

(4,4)

(5,7)

(6,9)

1. create new regions from A which contain the "upstream" region

2. find overlaps w/ B

3. for each overlap ensure the B feature does not overrun

for each ~~positive~~ forward overlap

B.end > ? a.start

↳ overlap

reverse

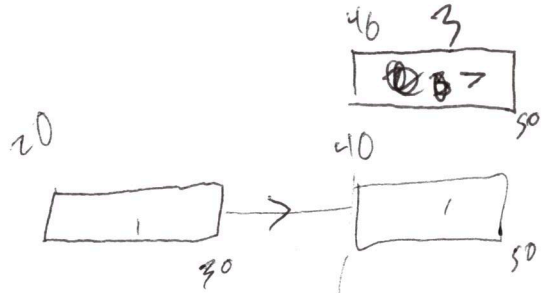
B

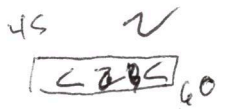
a.end > b.start

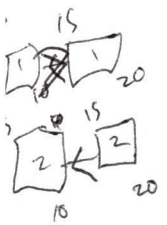
↳ overlap

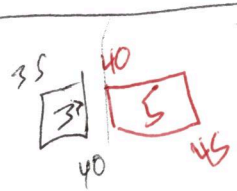
No.

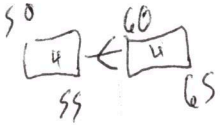
A.











Expected output

allow overlap

1,1

2,4

3,3

No overlap

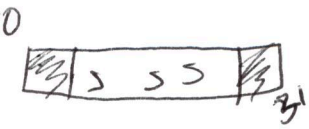
1,1

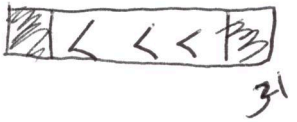
3,3

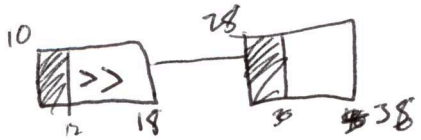
- dbtype prot
- in <file>
- parse_seqids
- logfile

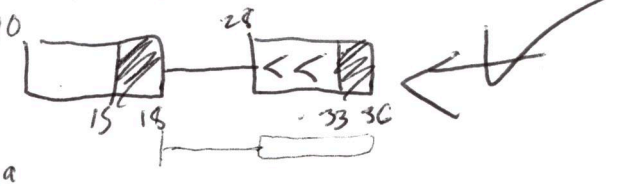
Q

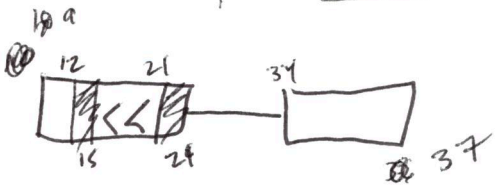
Cases:

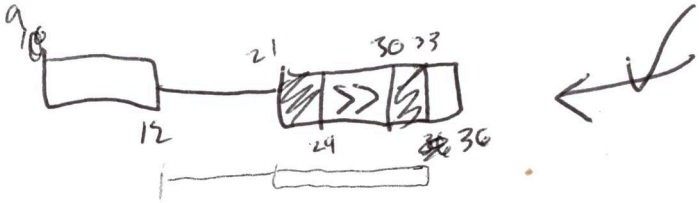
1. 

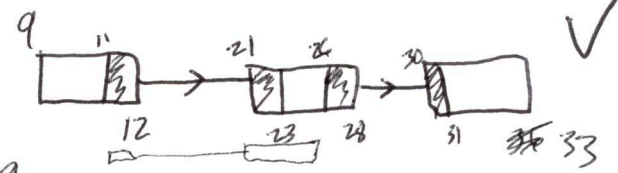
2. 

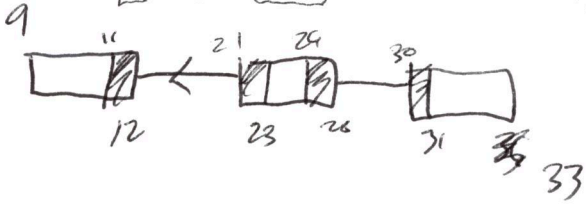
3. 

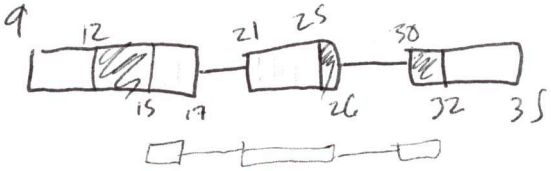
4. 

5. 

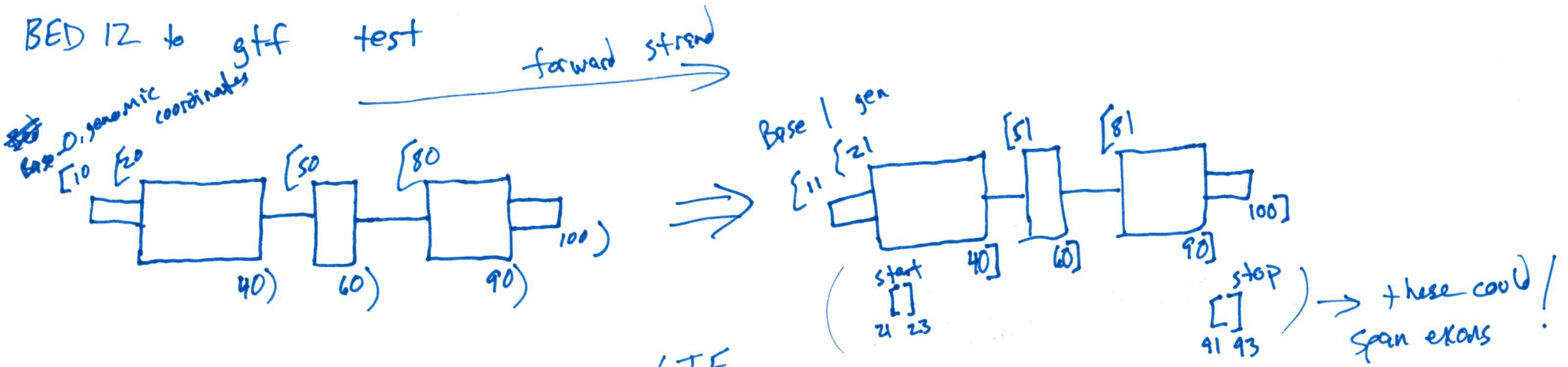
6. 

7. 

8. 



BED 12 to gtf test



BED

start 10

end 100

thick_start 20

thick_end 90

num_exons : 3

exon_lengths : 30, 10, 20

exon_rel_offsets : 0, 40, 70

GTF

exon_starts	exon_ends	exon_frame
11	40	.
51	60	.
81	100	.

CDS_starts	CDS_ends	CDS_frame
21	40	0
51	60	2
81	90	0