Homework 2
Index Structures

Due: *Friday, February 20, in-class*

Submission. I encourage everyone to use text-processing software for the solutions and use PowerPoint, xfig or other graphical tools to draw pictures/diagrams required in this homework. (I will accept handwritten solutions, but I discourage them. This policy is mainly to ensure expedient and error-free grading).

Problems

Consider a database relation R with an integer key X, which needs to be indexed. X has no duplicate keys. We consider the following two file organizations:

Data File A. Data file A is a sequential file with X as the search key. Each disk page can store four (4) records from the relation R on it.

Data File B. Data file B is a heap file. Each disk page can store three (3) records from the relation R on it.

Consider the following sequence of insertion and deletion operations for the relation R. For simplicity, `Insert(10)` means "insert a record with the value of X attribute set to 10".

```
Create(R);

Insert(100);
Insert(10);
Insert(65);
Insert(15);
Insert(80);

Insert(20);
Insert(35);
```

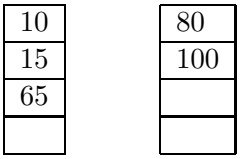
```
Delete(65);
Delete(80);
Insert(75);

Insert(70);
Insert(120);
Delete(15);
Delete(10);
Insert(35);


Insert(60);
Insert(55);
Delete(75);
Insert(160);
Delete(120);

Insert(75);
Insert(10);
Insert(40);
Delete(110);
Insert(95);
```

Problem 1

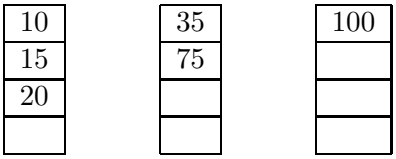
For each data file structure (A and B), show the state of the data files after each five `Insert`/`Delete` operations. For each disk page, show only pageID and any pointers you may have in the header (nothing else is needed for this exercise). You can omit the header page - just show the pages with the data.

File Structure A

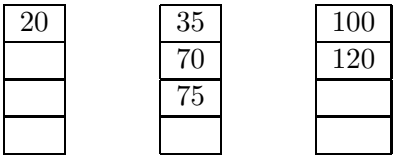
Checkpoint 1: Page 1 split after Insert(80)



Checkpoint 2: Page 1 split after Insert(35)



Checkpoint 3:



Checkpoint 4: Page 2 split after Insert(55)

20

35
55
60

70

100
160

Checkpoint 5:

10
20

35
40
55
60

70
75
95

100
160

File structure B

We assume that FreeSpace list is a stack (first deleted - last restored) and that the most recently added page is at the bottom of any non-trivial FreeSpace stack.

Checkpoint 1:

100
10
65

15
80

Checkpoint 2:

100
10

15
75
20

35

Checkpoint 3:

100
70

75
20

35
120

Checkpoint 4:

100
60
70

55
160
20

35

Checkpoint 5:

100
60
70

55
160
20

35
75
10

40
95

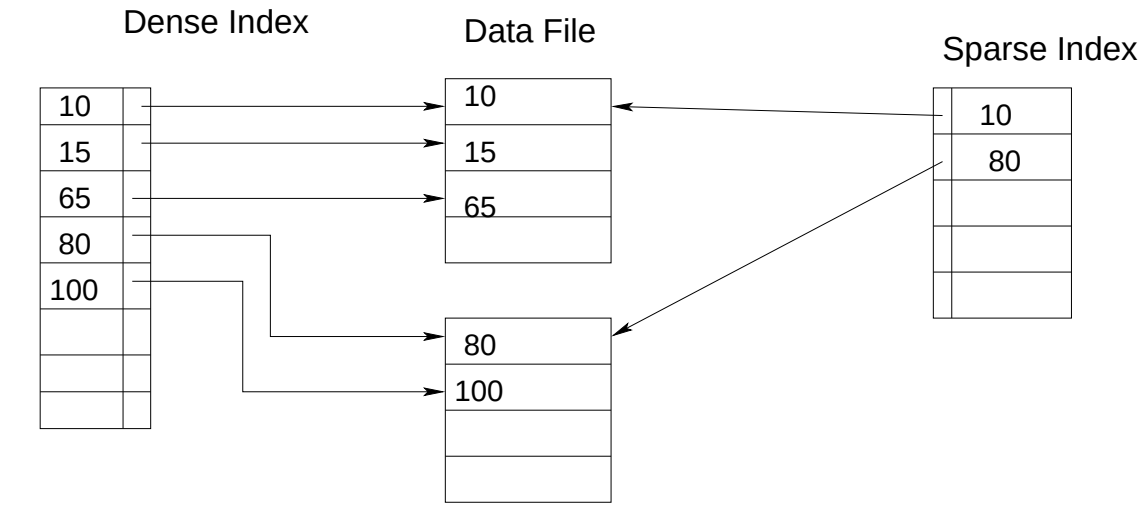
Problem 2

For Data File A, show the state of the following index files after every five `Insert`/`Delete` operations:

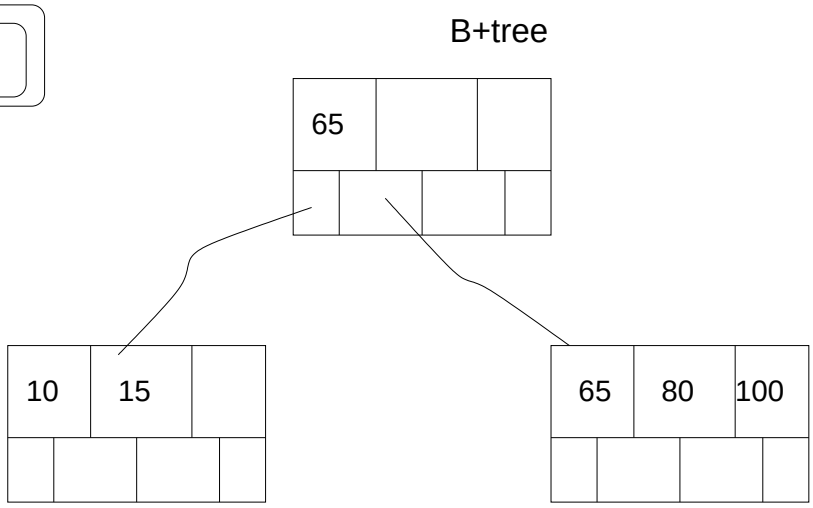
1. Simple Dense index (8 index records per page) on X;

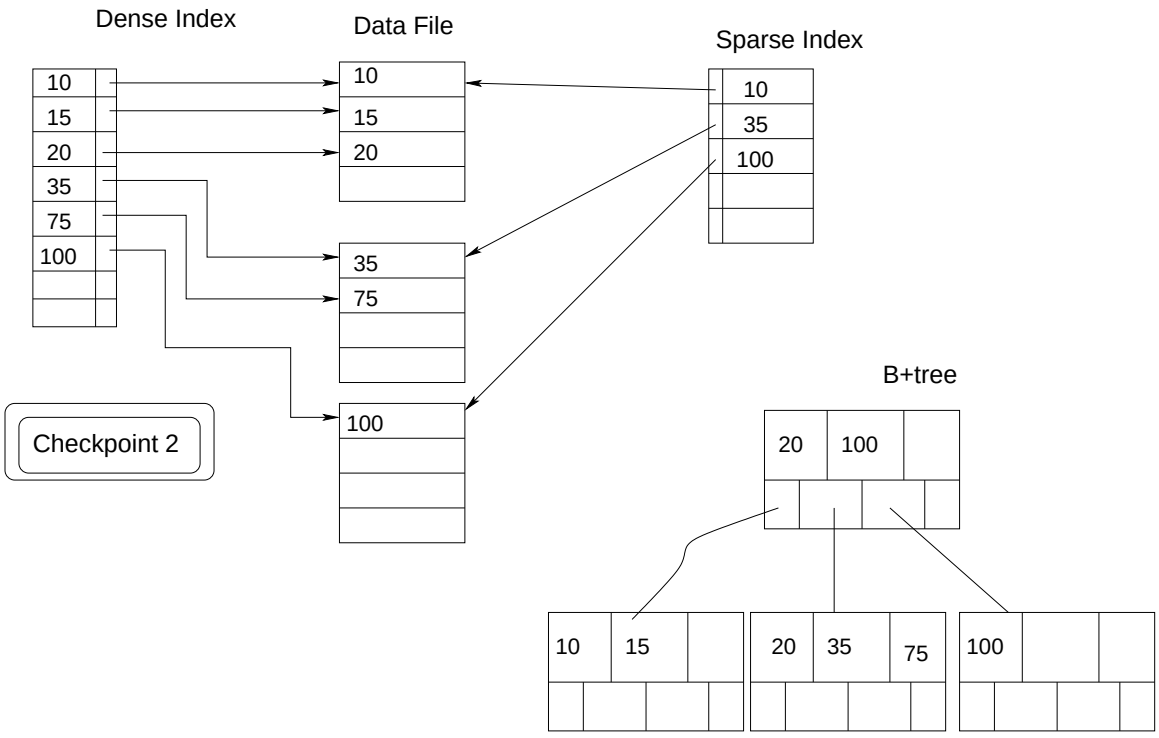
2. Simple Sparse (5 index records per page) index on X.

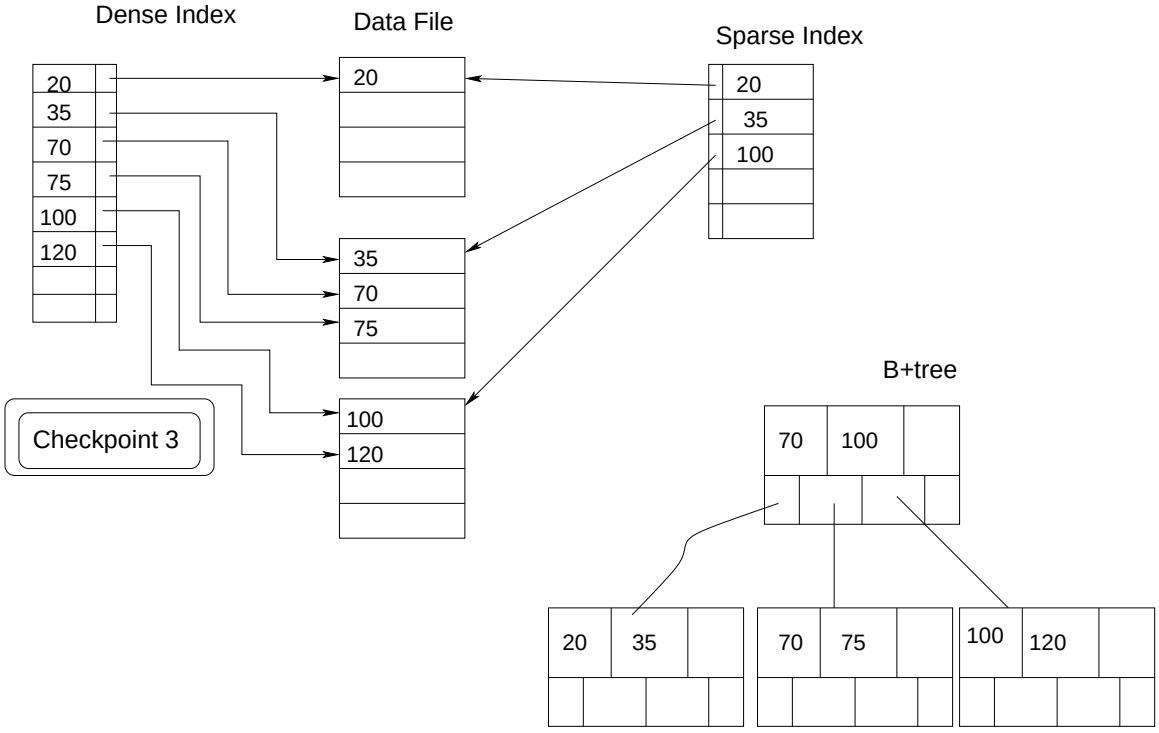
3. Dense B+-tree (3 key values per page) index on X;

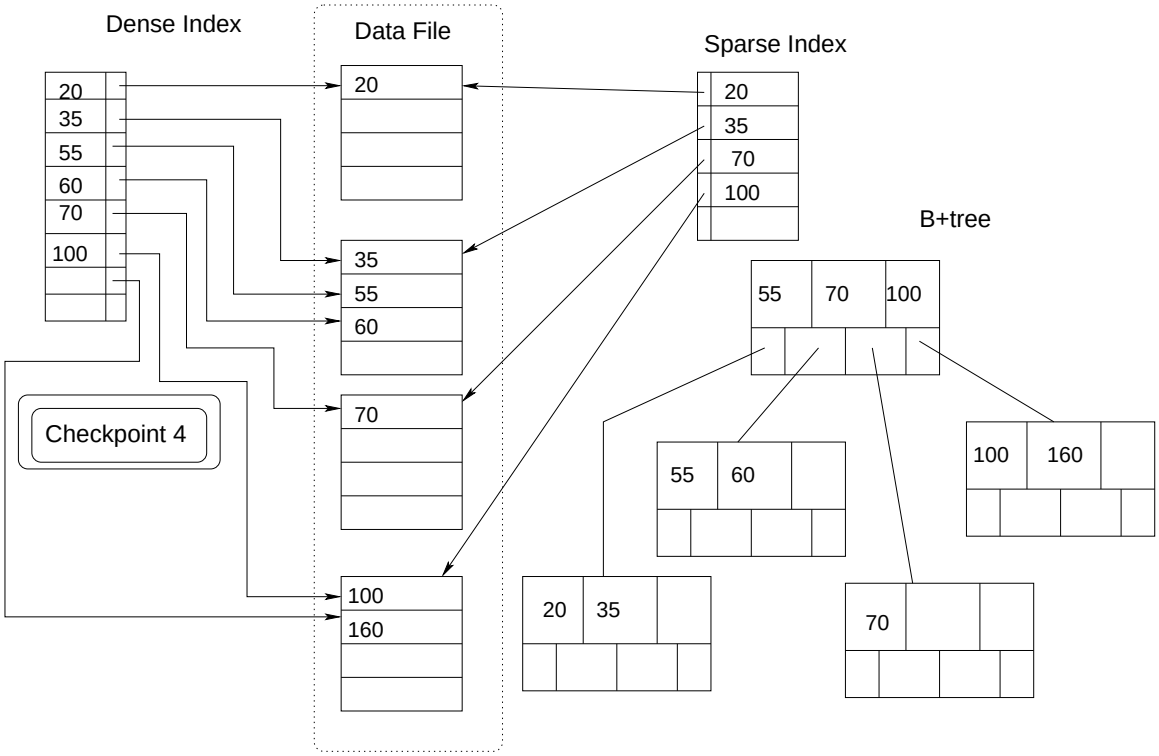


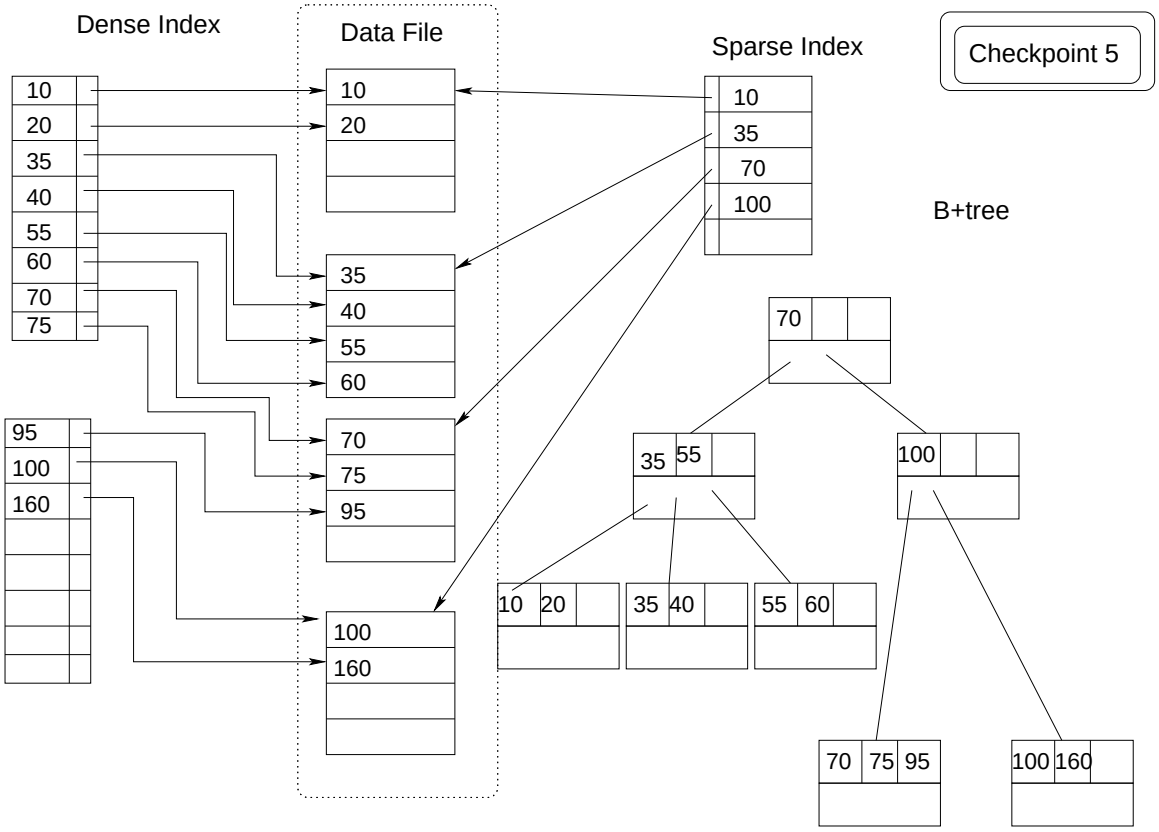
Checkpoint 1




Dense Index
Data File
Sparse Index
10
15
20
35
75
100
Checkpoint 2
B+tree
20
100
10
15
20
35
75
100


Dense Index
Data File
Sparse Index
20
35
70
75
100
120
Checkpoint 3
B+tree
70
100
20
35
70
75
100
120



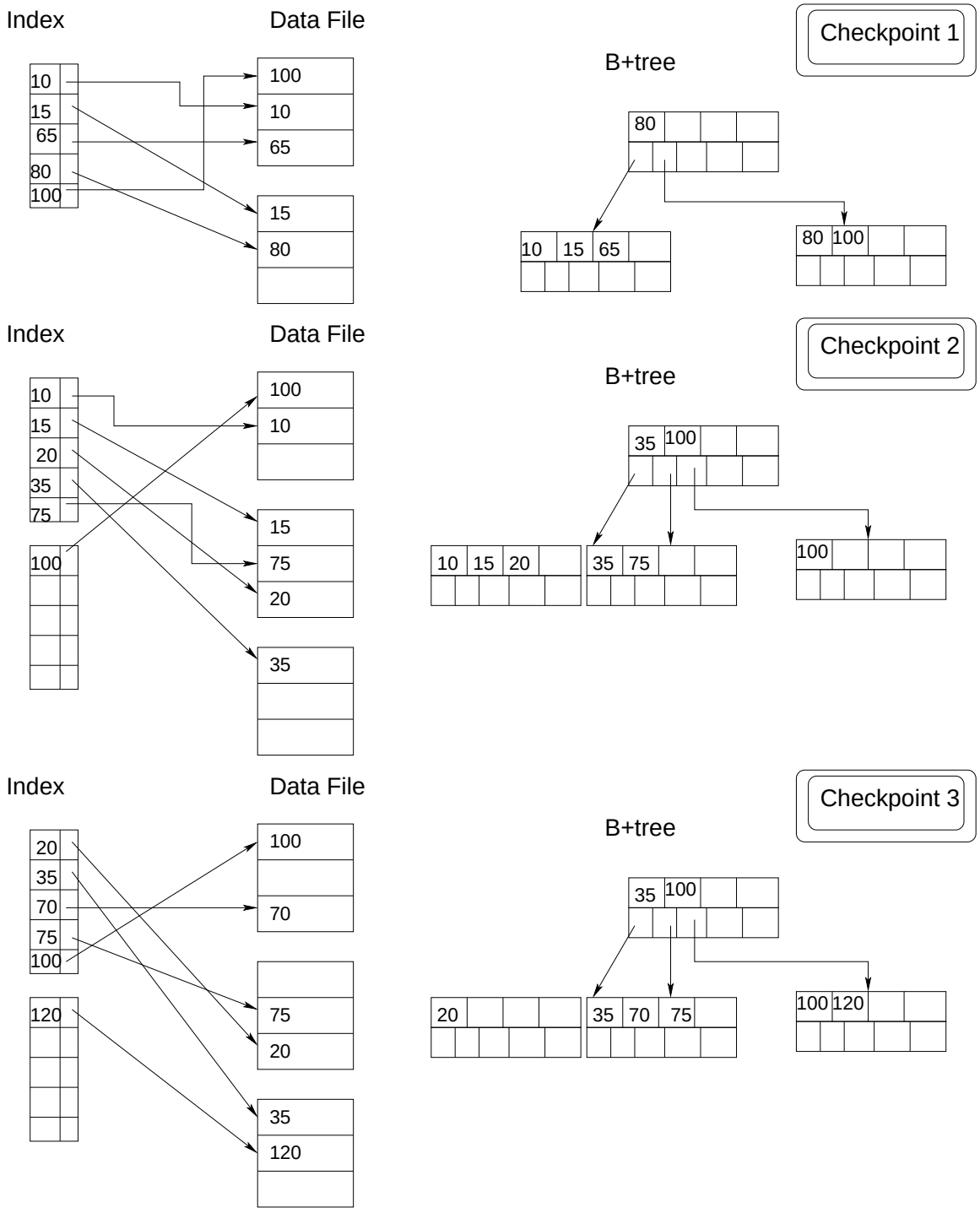


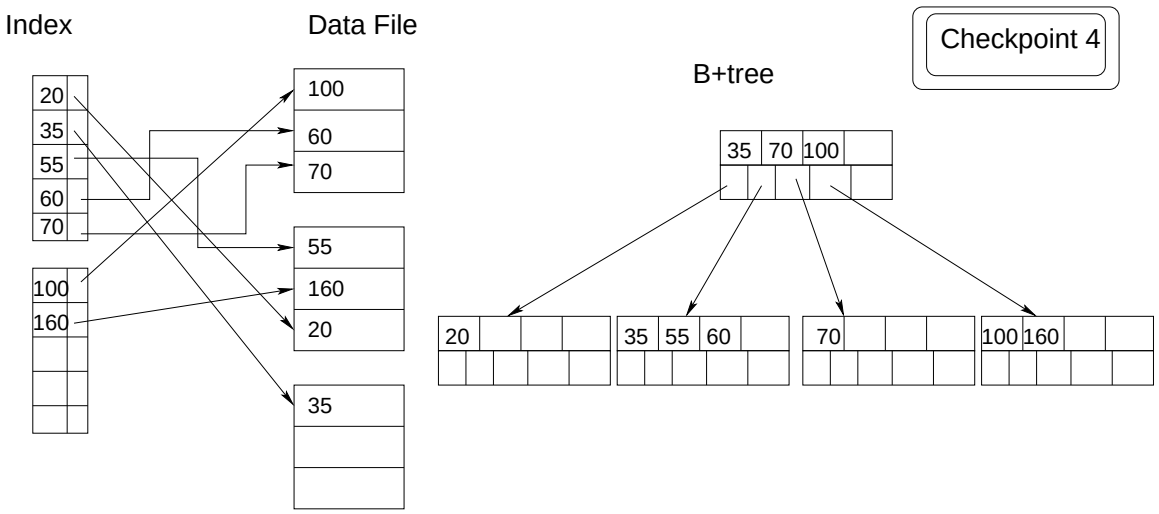
Problem 3

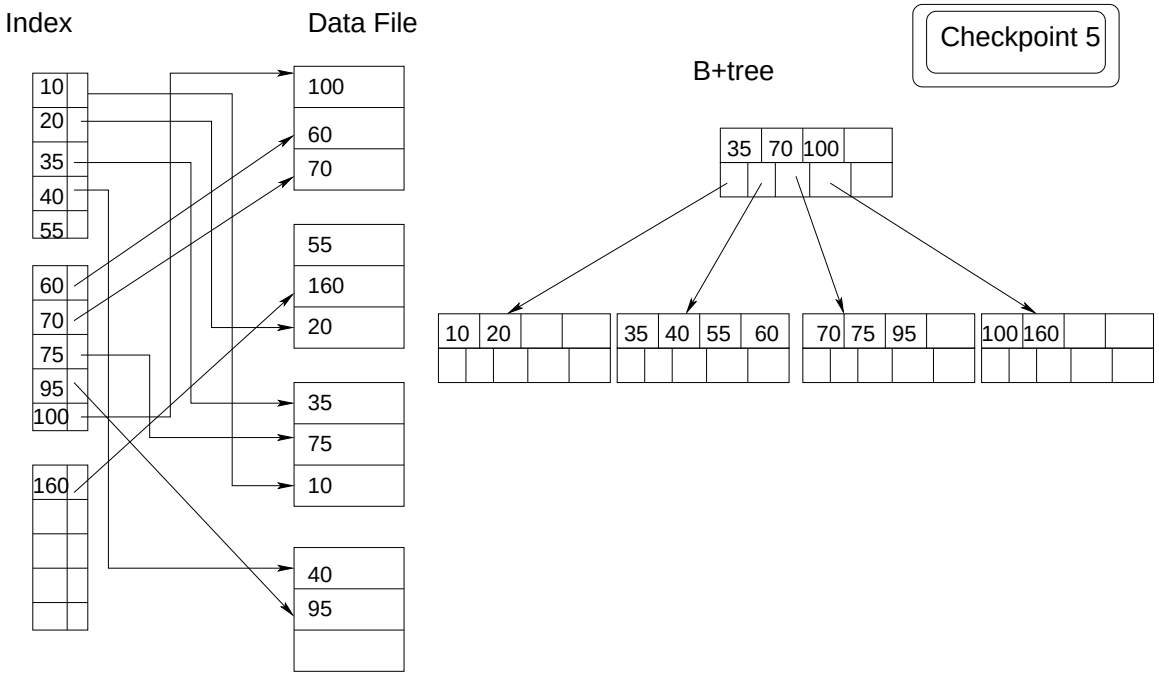
For Data File B, show the state of the following index files after every five

`InsertDelete` operations:

1. Simple secondary index (5 index records per page) on X;
2. B+-tree secondary index (4 key values per page) on X;




Index
Data File
20
35
55
60
70
100
160
100
60
70
55
160
20
35
Checkpoint 4
B+tree
35 70 100
20
35 55 60
70
100 160


Index
Data File
10
20
35
40
55
60
70
75
95
100
160
100
60
70
55
160
20
35
75
10
40
95
Checkpoint 5
B+tree
35 70 100
10 20
35 40 55 60
70 75 95
100 160