You should turn in the solutions to the written part of this assignment (questions 1 and 2) as a PDF file through Canvas before Midnight (by 11:59pm) on February 11th. The solutions should be produced using editing software programs, such as LaTeX or Word, otherwise they will not be graded. You should turn in the source code to each programming question (questions 3 and 4) separately through Canvas before Midnight (by 11:59pm) on Febuary 11th. Thus, each group will have three distinct submissions in Canvas for this assignment. The assignment should be done in groups of two students. Each submission must contain the full name, OSU email, and ONID of every member of the group.

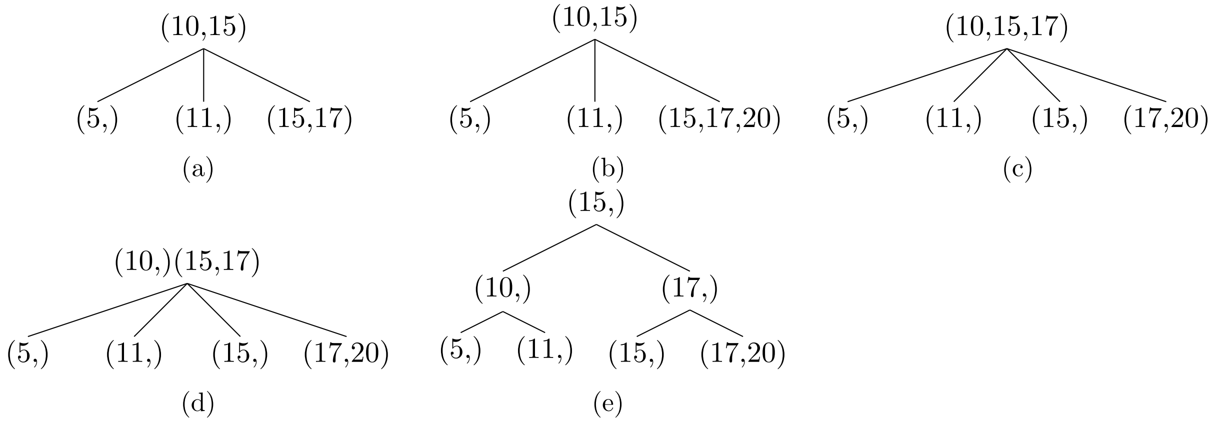
1: Tree Indexing (1 point)

Assume that the block size is 30 bytes. In this question, each node in a B+ tree must fit in a block and its size must be as close as possible to the size of a block. Answer the following parts using B+ trees.

(a) Assume that all data values are integers with the fixed size of 8 bytes and each record pointer takes at most 4 bytes. Find a B+ tree whose height changes from 2 to 3 when the value 20 is inserted. Note that the height of a tree is depth of the deepest node plus one. For example, the height of a tree with a single node is 1. Show your structure before and after the insertion and describe the steps that produced the resulting structure. (0.5 point)

(solution) The order of the B+ tree must be two as its nodes will have the greatest possible size that fit in a block.

Assume a B+ tree of order 1 on the left. The steps of insertion is shown below:



(a) (b) (c)

(d) (e)

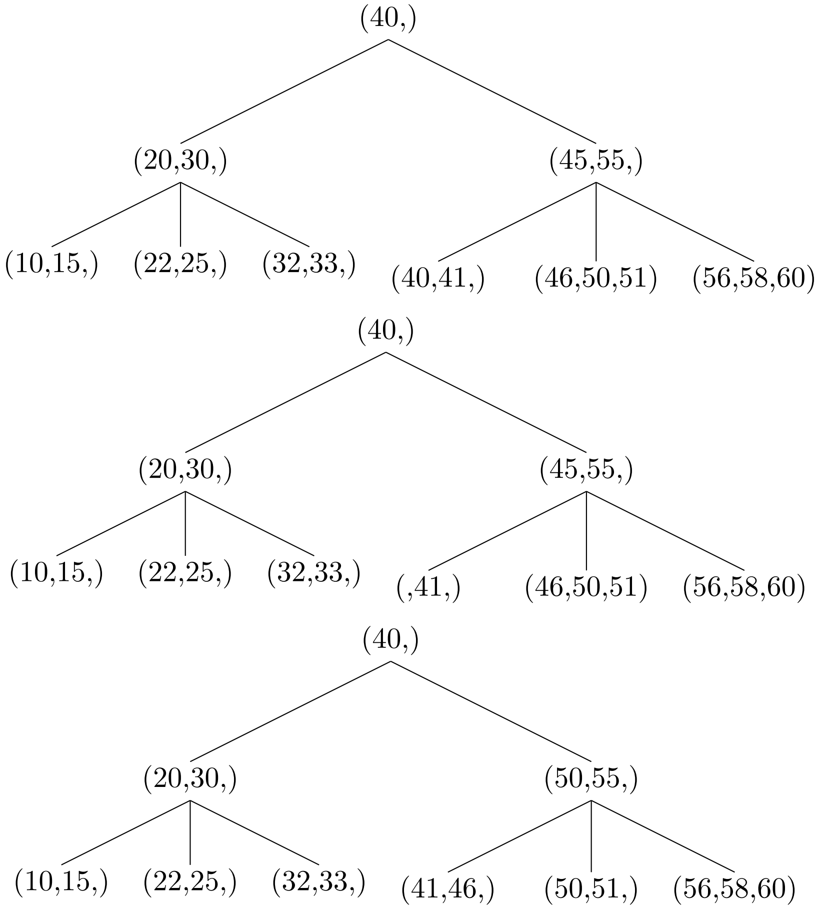
Diagram 1: Steps of the insertion

(b) Assume that all data values are integers with the fixed size of 4 bytes and each record pointer takes at most 4 bytes. Find a B+ tree in which the deletion of the value 40 leads to a redistribution. Show your structure before and after the deletion and describe the steps that produced the resulting structure. Your example could be different from the answer given for part (a). (0.5 point)

(solution)

The order of the B+ tree must be three as its nodes will have the greatest possible size that fit in a block.

The tree will change as shown below:



2: Indexing (1 point)

(a) Consider the following relational schema:
Emp(*eid*:integer, *ename*:string, *age*:integer, *salary*:real)

The underlined attributes are keys for their relations. Consider the following SQL query:

```
Select *
From Emp
Where age = 20 and salary > 20000
```

Describe the index on Emp that improves the running time of this query more than every other index over most Emp relation instances. You may explain the attributes, their order, type (B+ tree or hash) of the index and whether it should be clustered. (1 point)

(solution)

Clustered B+ tree index on (age,salary). Since the equality condition applies to age and the range condition to salary, the age should be the first attribute in the compound (composite) index.

3: Hash Indexing (5 points)

(a) Assume that we have a relation Employee(id, name, bio, manager-id). The values of id and manager-id are character strings with fixed size of 8 bytes. The values of name and bio are character strings and take at most 200 and 500 bytes. Note that as opposed to the values of id and manager-id, the sizes of the values of name (bio) are not fixed and are between 1 to 200 (500) bytes. The size of each block is 4096 bytes (4KB). The size of each record is less than the size of a block. Write a C++ program that reads an input Employee relation and builds a linear hash index for the relation using attribute id. You may use a hash function of your choice. Your program must increment the value of n if the average number of records per each block exceeds 80% of the block capacity. The input relation is stored in a CSV file, i.e., each tuple is in a separate line and fields of each record are separated by commas. The program must also support looking up a tuple using its id.

- The program must accept switch C for index creation mode and L for lookup in its command line. The switch L is succeeded by the value of input id.
- Your program must assume that the input CSV file is in the current working directory, i.e., the one from which your program is running, and its name is Employee.csv
- The program must store the indexed relation in a file with the name EmployeeIndex on the current working directory.
- Your program must run on hadoop-master.engr.oregonstate.edu. Submissions should also include the g++ command (including arguments) that was used to compile the program. Each student has an account on the *hadoop-master.engr.oregonstate.edu* server, which is a Linux machine. You can use the following *bash* command to connect to it:

  ```
  > ssh your_onid_username@hadoop-master.engr.oregonstate.edu
  ```

  It will prompt you for your ONID password. You will need to be connected to the VPN in order to access the server.
- You can use the following commands to compile and run C++ code:

  ```
  > g++ main.cpp -o main.out
  > main.out
  ```

4: R-Tree (5 points)

(a) Using the relations and data described in question (3), write a C++ program that reads an input Employee relation and builds an R-Tree index for the relation using attributes id and name. The input relation is stored in a CSV file, i.e., each tuple is in a separate line and fields of each record are separated by commas.

- The program must accept switch C for index creation in its command line.

- Your program must assume that the input CSV file is in the current working directory, i.e., the one from which your program is running, and its name is Employee.csv
- The program must store the index relation in a file with the name EmployeeRTree on the current working directory.
- Your program must run on hadoop-master.engr.oregonstate.edu. Submissions should also include the g++ command (including arguments) that was used to compile the program. Each student has an account on the *hadoop-master.engr.oregonstate.edu* server, which is a Linux machine. You can use the following *bash* command to connect to it:

  ```
  > ssh your_onid_username@hadoop-master.engr.oregonstate.edu
  ```

  It will prompt you for your ONID password. You will need to be connected to the VPN in order to access the server.
- You can use the following commands to compile and run C++ code:

  ```
  > g++ main.cpp -o main.out
  > main.out
  ```