CmpSci 187 Discussion #3: Dog Teams
Individual Handout

Marc Liberatore and John Ridgway

9 February 2015

Once again we have to do the discussion as homework; so here it is. It is due February 17, 2015 at the beginning of your discussion section.

Our goal here is to get some practice using linked lists. The code you are given is:

- A Dog class, where a Dog object contains a name and a weight,
- An LLDogNode class, whose objects are linked list nodes each containing a Dog, and
- A stub DogTeam class, in which you will complete three methods.

A DogTeam contains one or more Dog objects in some order. Your task is to implement the following three methods:

1. insertHead, which puts a new Dog at the head of the list,
2. insertTail, which puts a new Dog at the tail of the list, and
3. weightDiff, which returns the difference between the weights of the heaviest and the lightest Dog in the list. (A sled dog team is more effective if all the dogs in it have close to the same weight.)

When you have completed the methods, write their code on your response sheet — you need not repeat the code in the given source file. You've been given a main method in DogTeam, which you may supplement with other test code if your like, though this need not be handed in. You've also been given a method that prints out the names and weights of the dogs in a team.

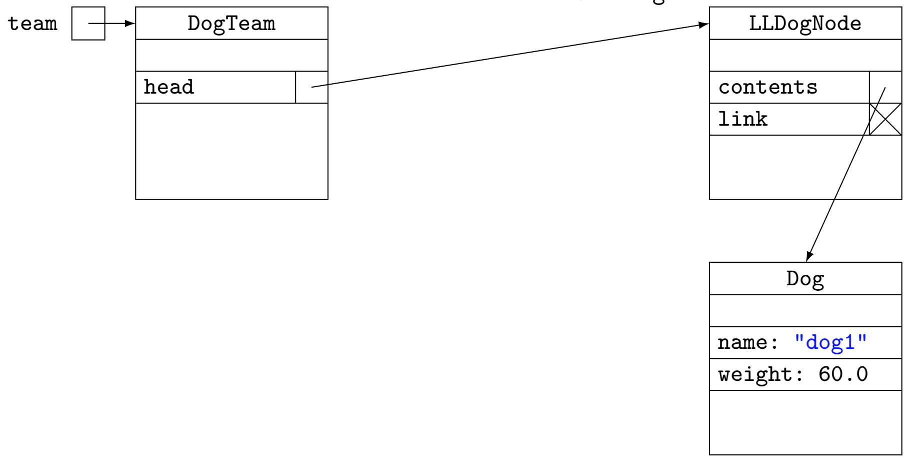
```java
public class Dog {

  private String name;
  private double weight;

  public Dog (String name, double weight) {
    this.name = name;
    this.weight = weight;
  }

  public String getName() {
    return this.name;
  }

  public double getWeight() {
    return this.weight;
  }

  public void setName (String name) {
    this.name = name;
  }

  public void setWeight (double weight) {
    this.weight = weight;
  }
}
```

```java
public class LLDogNode {

  private Dog contents;
  private LLDogNode link;

  public LLDogNode (Dog dog, LLDogNode link) {
    this.contents = dog;
    this.link = link;
  }

  public Dog getContents() {
    return contents;
  }

  public LLDogNode getLink() {
    return link;
  }

  public void setContents(Dog dog) {
    contents = dog;
  }

  public void setLink (LLDogNode link) {
    this.link = link;
  }
}
```

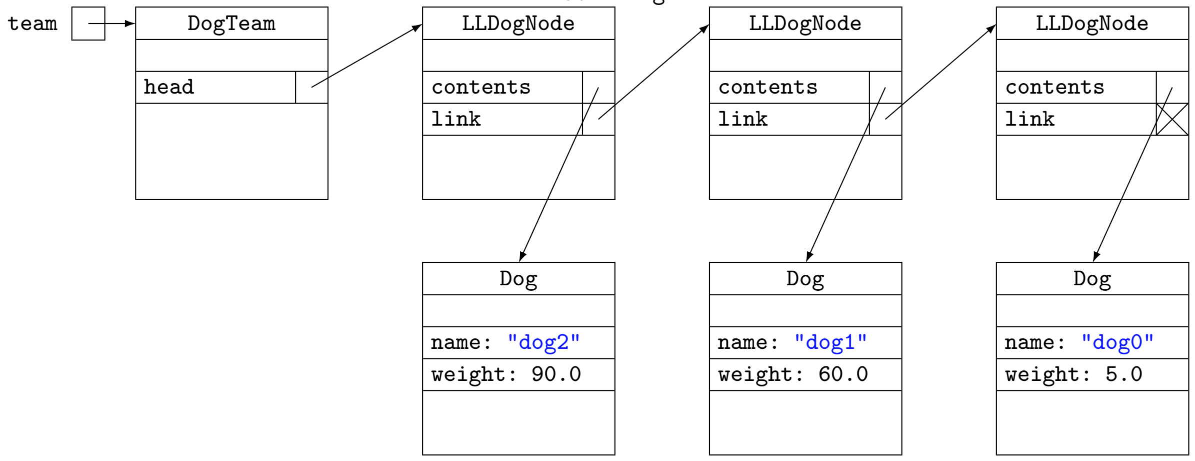
```java
public class DogTeam {

  private LLDogNode head;

  public DogTeam(Dog dog) {
    head = new LLDogNode(dog, null);
  }

  public void printTeam() {
    LLDogNode cur = head;
    int dogNumber = 1;

    System.out.println("-----------------");
    while (cur != null) {
      System.out.println(dogNumber + ". " + cur.getContents().getName() +
                         ", " + cur.getContents().getWeight());
      cur = cur.getLink();
      dogNumber += 1;
    }
  }


  public static void main(String[] args) {

    DogTeam team = new DogTeam(new Dog("dog1", 60));
    team.printTeam();
    System.out.println("weightDiff: " + team.weightDiff());

    team.insertTail(new Dog("dog0",  5));
    team.insertHead(new Dog("dog2",  90));
    team.printTeam();
    System.out.println("weightDiff: " + team.weightDiff());

    team.insertHead(new Dog("dog3",  7));
    team.insertTail(new Dog("dog4",  100));
    team.insertTail(new Dog("dog10", 205));
    team.printTeam();
    System.out.println("weightDiff: " + team.weightDiff());
  }
```

Since, once again, you are doing this as homework rather than in a discussion session, we are providing a couple of starting pictures to help you out. We strongly suggest that you add to these pictures in order to help you understand just what you need to do.

After line 25 of `DogTeam`.



After line 30 of `DogTeam`.



Names: ____

```
  public void insertHead(Dog dog) {
    // TODO(0)
    // puts new node containing dog at the head of the list



















  }

  public void insertTail(Dog dog) {
    // TODO(1)
    // puts new node containing dog at the tail of the list


















  }

  public double weightDiff() {
    // TODO(2)
    // returns difference between max and min weights of dogs in list
    // pre: this list contains at least one node
    return 0.0;


















  }
}
```