Lab Exercise 6
Implementing the Generic Bag Class

1 Introduction

In a previous lab, you implemented an IntArrayBag class and a StringArrayBag. Each of these used a
dynamically sized array as its underlying data structure. In this lab exercise you will implement a generic
ArrayBag class which uses an array as its underlying data structure. Because this ArrayBag class is a generic
class, its implementation will not require it to store a specific type of data.

2 Convert ArrayBag from String to Generic

Using Eclipse, import “Lab6”:

/afs/cs.uwm.edu/users/classes/cs351/401/pantherid/git/lab6.git

Currently, ArrayBag implements a bag of Strings with an array. Change this class to use an array of generic
types, so that you can use this class to implement a bag of any type you choose.

In particular, there are two drivers, StringDriver.java and IntegerDriver.java. You should be able to
run StringDriver.java and see similar results to a previous lab’s. Try to run IntegerDriver.java and
see what happens. If you convert ArrayBag.java to use a generic type, you should be able to run both
drivers, each making use of the same class.

To designate a generic type to be used by a class, you must put a generic type parameter in the class
definition. Here we use the symbol T to represent the generic type:

public class ClassName<T> {

The generic type can now be used throughout the class definition, including its fields and methods:

public T getter() {
    return _data;
}

When an object of this class is created, it will be parameterized by a specific type, like so:

ClassName<Integer> object = new ClassName<Integer>();

For this object, any methods that use the type T, such as our getter above, will now appear to use Integer
in place of the generic symbol T. This object’s getter will return an Integer.

However, at compile time, Java does not know what type these generics will actually be. This leads to some
interesting consequences. For example, you cannot create an array of generic type. Instead, you need to
create an array of Objects and then cast it:

T[] tArray = (T[]) new Object[SIZE];

You also have to use the equals method when comparing two values of generic type. For example:

generic1.equals( generic2 )
3 Update StringDriver

Now that you have changed ArrayBag to use a generic type, update StringDriver to use the generic type. You should see that it runs and gives the same output as before.

4 Fixing IntegerDriver

IntegerDriver.java is already written to use the generic version of ArrayBag. You should expect that IntegerDriver should now run with no problem. However, you will see an error message when you try to run it. This is because the array returned by getData is, in fact, an array of Objects that has simply been cast to something else. Thus the assignment of this array to a reference variable holding an array of a specific type (in this case Integer) will fail.

Look at the code where the error occurs and understand what it is meant to accomplish. Find another way, using ArrayBag methods, to accomplish this goal.

Once these steps have been completed, please see your TA to receive credit for this lab.