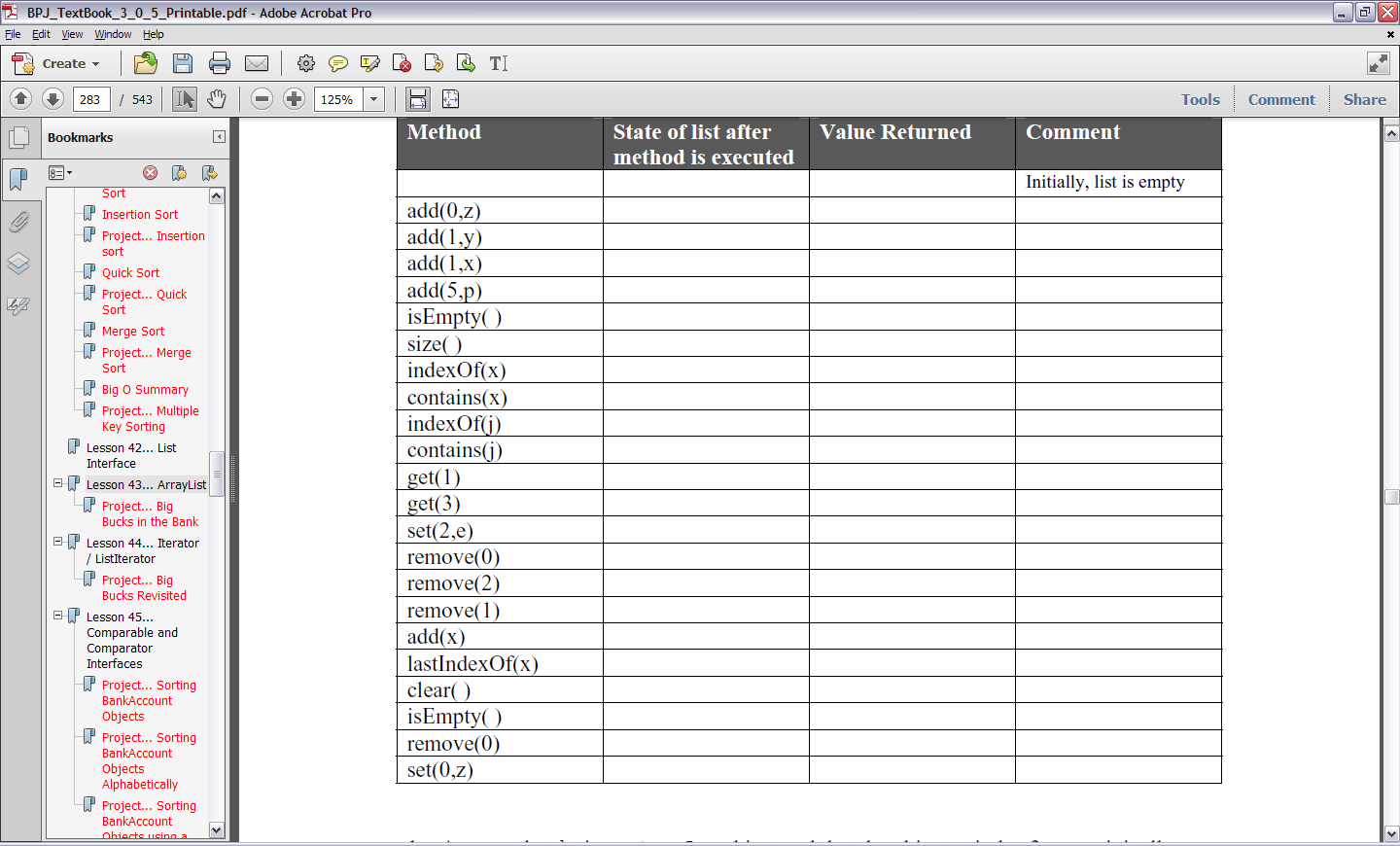
**Name:**

**Advanced Programming in Java**

**ArrayLists**

**Lab Exercise 12/3/2019**

Fill in the blanks below, except the comment column. Only put a note in the comment column if there is an exception (error). The letters *e*, *j*, *p*, *x*, *y,* and *z* all represent objects.



1. Assume that *lst* is an *ArrayList* object and that the object at index 2 was originally a

wrapper class *Integer* before being added to the list. Also assume that *lst* was **not** created

with type parameters. The following code will “almost” work. Put in the modification

that will make it work.

Integer iw = lst.get(2);

2. public static void theMethod(List opal)

{

Iterator iter = ??????;

… more code …

}

What code replaces ?????? so that an *Iterator* object is created from *opal*?

3. Write code that will instantiate an *ArrayList* object called *alst* and have the restriction that

only *String* objects can be stored in it.

4. *ArrayList*s are restricted in that only \_\_\_\_\_\_\_\_\_\_\_\_ can be stored in them.

5. What is the main advantage in using an *ArrayList* object as opposed to an ordinary array?

6. What is a disadvantage of using primitive variable types with an *ArrayList* object.

In problems 7 - 9 an operation is performed with the “ordinary” array *ary*. Write equivalent code

that performs the same operation on the *ArrayList* object called *a*. Assume that Java 5.0 is being

used and give two answers for each problem (parts A and B). For A part assume that *a* was

created with *List a = new ArrayList( );* and for B part assume *List<Integer>a = new*

*ArrayList<Integer>( );* was used:

7. int x = 19;

ary[5] = x;

8. int gh = ary[22];

9. int sz = ary.length;

10. int kd = ary[101];

ary[101] = 17;

Use the set method:

11. //Before inserting a new number, 127, at position 59, it will be necessary to move all

//up one notch. Assume that the logical size of our array is *logicalSize*.

for(int j = logicalSize; j >=59, j--)

{

ary[j+1] = ary[j];

}

ary[59] = 127; //insert the new number, 127, at index 59.

What code using *List* method(s) does the equivalent of the above code?

12. What does the following code accomplish? (*alist* is an *ArrayList* object)

while(!alist.isEmpty( ) )

{

alist.removeLast( );

}

13. What one line of code will accomplish the same thing as does the code in #10 above?

14. Write a single line of code that will retrieve the *String* object stored at index 99 of the

*ArrayList* object *buster* and then store it in a *String* called *myString*.

15. What type variable is always returned when retrieving items from an *ArrayList* object?

**Project… Big Bucks in the Bank**

Create a project called *BigBucks*. It will have two classes in it, a *Tester* class and a *BankAccount*

class. The code for *BankAccount* follows:

public class BankAccount

{

public String name;

public double balance;

public BankAccount(String nm, double amt)

{

name = nm;

balance = amt;

}

public void deposit(double dp)

{

balance = balance + dp;

}

public void withdraw(double wd)

{

balance = balance - wd;

}

}

You will need to create a *Tester* class that has a *main* method that provides a loop that lets you

enter several *BankAccount* objects. As each is entered, it will be added to an *ArrayList* object.

After several accounts have been entered, a loop will step through each *BankAccount* object in

the *ArrayList* and decide which account has the largest balance that will then be printed.

Following is the output screen after a typical run:

Please enter the name to whom the account belongs. ("Exit" to abort) Jim Jones

Please enter the amount of the deposit. 186.22

Please enter the name to whom the account belongs. ("Exit" to abort) Bill Gates

Please enter the amount of the deposit. 102.15

Please enter the name to whom the account belongs. ("Exit" to abort) Helen Hunt

Please enter the amount of the deposit. 1034.02

Please enter the name to whom the account belongs. ("Exit" to abort) Charles Manson

Please enter the amount of the deposit. 870.85

Please enter the name to whom the account belongs. ("Exit" to abort) exit

The account with the largest balance belongs to Helen Hunt.

The amount is $1034.02.

A partially complete *Tester* class is presented below. You are to complete the parts indicated in

order to achieve the screen output above.

import java.io.\*;

import java.util.\*; //includes ArrayList

import java.text.\*; //for NumberFormat

public class Tester

{

public static void main(String args[])

{

NumberFormat formatter = NumberFormat.getNumberInstance( );

formatter.setMinimumFractionDigits(2);

formatter.setMaximumFractionDigits(2);

String name;

//**Instantiate an ArrayList object here called aryList**

do

{

Scanner kbReader = new Scanner(System.in);

System.out.print("Please enter the name to whom" +

" the account belongs.(\"Exit\" to abort)");

name = kbReader.nextLine( );

if( !name.equalsIgnoreCase("EXIT") )

{

System.out.print("Please enter the amount" +

" of the deposit. ");

double amount = kbReader.nextDouble();

System.out.println(" "); //gives an eye-pleasing blank line

// **Create a BankAccount object**

**// Add it to the ArrayList object**

}

}while(!name.equalsIgnoreCase("EXIT"));

**// Search aryList and print out the name and**

**// amount of the largest bank account**

BankAccount ba = //**get first account in the list**

double maxBalance = ba.balance;

String maxName = ba.name;

for(int j = 1; j < aryLst.size( ); j++)

{

**//Step through the remaining objects and decide which one has**

**//largest balance (compare each balance to maxBalance)**

}

**//Print answer**

}//end of main

}//end of Tester class