W2D1 Homework

1 What’s character of these Collection, List, Set and Map? ArrayList and LinkedList? HashMap and TreeMap?

**List,Set,Map**

**List** is a collection of ordered elements that can have duplicates and is dynamic which means that its size can be resizable, list also allow any number of null values. **Set** on the other hand, is a collection of unordered and unique elements and only allows a single null value at most. **Map** provides data structure depending on the value of key pair and hashing, lastly, map can only have one null key and any number of null values.

**ArrayList and Linked List**

**ArrayList** and **LinkedList** are both part of list interface with different implementations. Their methods and results are almost the same. But both have disadvantages and advantages depending on what the requirement is.  
  
When it comes to **searching** an element **ArrayList** is the best choice because it uses array data structure for searching an element in a list while **LinkedList** goes through all the contents of the list until it finds the right one. When **removing** elements **LinkedList** does better performance because it only moves the neighbor of the element but not all the elements unlike in **ArrayList** all elements needs to shift in order to fill out the blank spaces inside the array. Memory consumption of LinkedList is higher than ArrayList.

**HashMap and TreeMap  
  
TreeMap** is basically a kind of a sorted map that’s why it’s better to use when iterating on the other hand **HashMap** is unsorted which makes no guarantee that the results will be the same during the iteration.

2. （List）Read the codes

**import java.util.\*;**

**public class Test {**

**public static void main(String args[]) {**

**List list = new ArrayList();**

**list.add("Hello");**

**list.add("World");**

**list.add(1, "Learn");**

**list.add(1, "Java");**

**printList(list);**

**}**

**public static void printList(List list) {**

**// 1**

**for (int i = 0; i < list.size(); i++) {**

**System.out.println(list.get(i));**

**}**

**for (Object o : list) {**

**System.out.println(o);**

**}**

**Iterator itor = list.iterator();**

**while (itor.hasNext()) {**

**System.out.println(itor.next());**

**}**

**}**

**}**

Requirement:

Complete the codes at //1, and need to print out all the elements of the list.  
**import** java.util.\*;

**public** **class** Test {

**public** **static** **void** main(String args[]) {

List list = **new** ArrayList();

list.add("Hello");

list.add("World");

list.add(1, "Learn");

list.add(1, "Java");

*printList*(list);

}

**public** **static** **void** printList(List list) {

ArrayList<String> basic = **new** ArrayList<String>();

**for** (**int** i = 0; i < list.size(); i++) {

System.***out***.println(list.get(i));

}

**for** (Object o : list) {

System.***out***.println(o);

}

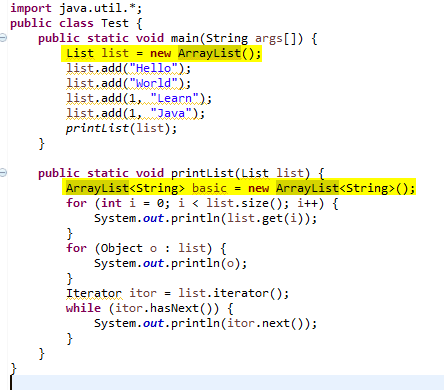
Iterator itor = list.iterator();

**while** (itor.hasNext()) {

System.***out***.println(itor.next());

}

}

}  


1. Write the output of the code.



1. Where and how to modify if change Arraylist with LinkedList? What’s the difference between ArrayList and LinkedList?

ArrayList basically uses dynamic array to store elements while LinkedList uses doubly linked list to store elements in a collection  
**import** java.util.\*;

**public** **class** Test {

**public** **static** **void** main(String args[]) {

List list = **new** LinkedList();

list.add("Hello");

list.add("World");

list.add(1, "Learn");

list.add(1, "Java");

*printList*(list);

}

**public** **static** **void** printList(List list) {

LinkedList<String> basic = **new** LinkedList<String>();

**for** (**int** i = 0; i < list.size(); i++) {

System.***out***.println(list.get(i));

}

**for** (Object o : list) {

System.***out***.println(o);

}

Iterator itor = list.iterator();

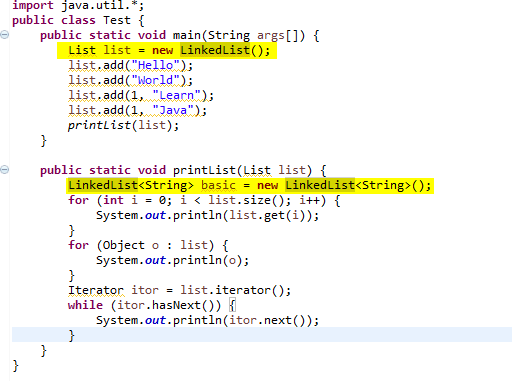
**while** (itor.hasNext()) {

System.***out***.println(itor.next());

}

}

}



Where and how to modify if change Arraylist with Vector? What’s the difference between ArrayList and Vector?  
ArrayList is unsynchronized wherein there can be multiple threads running at the same time in the List while vector is synchronized which means that only one thread can perform an operation at the same time.  
**import** java.util.\*;

**public** **class** Test {

**public** **static** **void** main(String args[]) {

List list = **new** Vector();

list.add("Hello");

list.add("World");

list.add(1, "Learn");

list.add(1, "Java");

*printList*(list);

}

**public** **static** **void** printList(List list) {

Vector<String> basic = **new** Vector<String>();

**for** (**int** i = 0; i < list.size(); i++) {

System.***out***.println(list.get(i));

}

**for** (Object o : list) {

System.***out***.println(o);

}

Iterator itor = list.iterator();

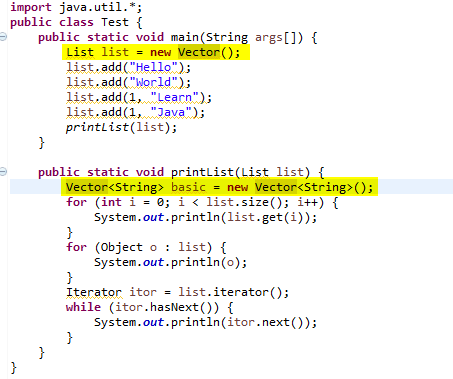
**while** (itor.hasNext()) {

System.***out***.println(itor.next());

}

}

}



3. （List）Write the output of the program.

**import** java.util.\*;

**public** **class** TestList {

**public** **static** **void** main(String args[]) {

List list = **new** ArrayList();

list.add("Hello");

list.add("World");

list.add("Hello");

list.add("Learn");

list.remove("Hello");

list.remove(0);

**for**(**int** i = 0; i < list.size(); i++) {

System.***out***.println(list.get(i));

}

}

}

Output:   


4. Select the right one?

**import** java.util.\*;

**public** **class** TestListSet {

**public** **static** **void** main(String args[]) {

List list = **new** ArrayList();

list.add("Hello");

list.add("Learn");

list.add("Hello");

list.add("Welcome");

Set set = **new** HashSet();

set.addAll(list);

System.***out***.println(set.size());

}

}

1. Compile with error
2. Compile correctly, but throw exception when running.
3. Compile and run well, and output 3
4. Compile and run well, and output 4

5 (List, Map)

**public** **class** Worker {

**private** **int** age;

**private** String name;

**private** **double** salary;

**public** Worker() {

}

**public** Worker (String name, **int** age, **double** salary) {

**this**.name = name;

**this**.age = age;

**this**.salary = salary;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **double** getSalary() {

**return** salary;

}

**public** **void** setSalary(**double** salary) {

**this**.salary = salary;

}

**public** **void** work() {

System.***out***.println(name + "is working");

}

}

Please finish the requirement:

1. To create a List and add three workers, and their information shown like this:

|  |  |  |
| --- | --- | --- |
| Name | Age | Salary |
| Simon | 20 | 10000 |
| Jame | 25 | 13000 |
| Alex | 22 | 12000 |

1. Add one worker before Jame ( Steven, 24, 15000)
2. Remove the worker Alex’s information
3. Go through the list using for statement and print out all the worker’s information.
4. Go through the list using Iterator statement to call all the worker’s method work.
5. Over write the equals method for the class Worker. New equals method return true only if the workers’ name, age and salary are the same at the same time.
6. **Sort the all the workers from high to low by salary** and print out the all the workers information with the format “Name: “ + name + “ Salary: “ + salary.
7. Add a id to Worker class, and save the above data to workMap. Map<String, Worker > ( Worker ID, Worker) . **At least three ways t**o go through the workMap, to print out all the workder’s information with Worker id and all other information like “Worker Id: “ + “Name: “ + name + “Age: “ + age + “ Salary: “ + salary.