W2D1 Homework

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

* Collection, Collections, List, Set and Map
  + Collection – Collects or group multiple elements/values and manipulate them(Store,remove,view,replace and etc). it is an interface of List and Set.
  + Collections - It contains polymorphic algorithms that operate on collections. This algorithms is used for Collection and its implementation. You may able to access some of this algorithm for example Collections.shuffle , Collections.sort and etc.
  + List – implements the “Collection” interface it can store an ordered sequence of elements. Elements can be added in front, end or back in the List. Also each elements is accessible by index. It is also an interface for ArrayList , LinkedList and etc.
  + Set – this also implements the “Collection” interface but this type of collection is different to list because this only gets the Unique elements if there is duplicates it removes the duplicate and set is unordered. It is also a Interface for LinkedHashSet, HashSet and etc.
  + Map – is also a Collection but it is not part of the Interface of “Collection” this stores values and per value it has a key to access that certain value. The key can be any Object and also the value can be any Object (Map<Object,Object>). For example Map<Integer, String>.
* ArrayList and LinkedList
  + ArrayList – Implements “List” interface. Arraylist can easily access elements but adding and removing elements always require to move the indexes or change the size so it won’t catch IndexOutOfBounce or NullPointer Exceptions.
  + LinkedList – Also implements “List” interface. LinkedList is a little bit slow in finding a position of a certain element. But it can easily add or remove elements because it once removed it will just point to the next element also same concept in adding a new element.
* HashMap and TreeMap
  + Hashmap – Implements “Map” interface. It can do the basic operation of Map inserting Key and assigning its value then retrieving a value using a key.
  + Treemap – Also implements “Map” interface. It can also do the basic operations of map but once you have inserted the values it will automatically sort it in natural order: using the key and sorting it in Ascending order.

***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());**

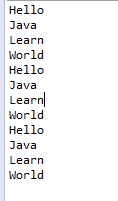
**}**

**}**

**}**

Requirement:

1. **Write the output of the code.**



1. **Where and how to modify if change Arraylist with LinkedList?** 
   * **List list = new ArrayList()** change to **List list = new LinkedList();**
2. **Where and how to modify if change Arraylist with Vector? What’s the difference between ArrayList and Vector?**
   * **List list = new ArrayList()** change to **List list = new Vector();**
   * Vector and ArrayList is almost the same. But the main difference is Vectors are synchronized and ArrayList are not.

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++) {

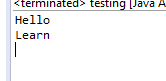
System.***out***.println(list.get(i));

}

}

}

Output of the Program is :



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 (**ANSWER**)
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.