**List Interface Exercises**

Interface:

1) What does the following code output?

[4, 5, 6, 1, 2, 3]

public class Main

{

public static void main(String[] args)

{

List<Integer> aList = new ArrayList<Integer>(Arrays.asList(1,2,3)); List<Integer> aList2 = new ArrayList<Integer>(Arrays.asList(4,5,6)); List<Integer> lList = new LinkedList<Integer>();

lList.addAll(aList2);

lList.addAll(aList);

System.out.println(lList);

}

}

2) Does the following compile?

No

public class Main

{

public static void main(String[] args)

{

List<Integer> aList = new ArrayList<Integer>(Arrays.asList(1,2,3)); List<Integer> aList2 = new ArrayList<Integer>(Arrays.asList(4,5,6)); List<Float> lList = new LinkedList<Float>();

lList.addAll(aList2);

lList.addAll(aList);

System.out.println(lList);

}

}

3) What is the output of the following?

[]

public class Main

{

public static void main(String[] args)

{

List<Integer> aList = new ArrayList<Integer>(Arrays.asList(1,2,3));

List<Integer> aList2 = new ArrayList<Integer>(Arrays.asList(4,5,6));

List<Integer> lList = new LinkedList<Integer>();

lList.addAll(aList2);

lList.addAll(aList);

lList.clear();

System.out.println(lList);

}

}

4) What is the output of the following?

lList contains 4

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

**public class** Main

{

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

{

List<Integer> lList = **new** LinkedList<Integer>(Arrays.*asList*(1,2,3,4,5,6));

**if**(lList.contains(4))

{

System.*out*.println("lList contains 4");

}

**else**

{

System.*out*.println("lList does not contain 4");

}

}

}

5) Does the following compile?

Yes

List<Integer> lList = **new** LinkedList<Integer>(Arrays.*asList*(1,2,3,4,5,6));

**if**(lList.contains(4.0))

{

System.*out*.println("lList contains 4");

}

**else**

{

System.*out*.println("lList does not contain 4");

}

6) What is the output of the following?

lList does not contain 4

List<Integer> lList = **new** LinkedList<Integer>(Arrays.*asList*(1,2,3,4,5,6));

**if**(lList.contains(4.0))

{

System.*out*.println("lList contains 4");

}

**else**

{

System.*out*.println("lList does not contain 4");

}

10) Does the following compile?

Yes

List<Integer> lList = **new** LinkedList<Integer>(Arrays.*asList*(1,2,3,4,5,6));

List<Float> oList = **new**

LinkedList<Float>(Arrays.*asList*(1.0f,2.0f,3.0f,4.0f,5.0f,6.0f,7.0f));

**if**(lList.containsAll(oList))

{

System.*out*.println("lList contains 4");

}

**else**

{

System.*out*.println("lList does not contain 4");

}

11) What is the output of the following?

lList is equal to vList

List<Integer> lList = **new** LinkedList<Integer>(Arrays.*asList*(1,2,3,4,5,6));

List<Integer> vList = **new** Vector<Integer>(Arrays.*asList*(1,2,3,4,5,6));

**if**(lList.equals(vList))

{

System.*out*.println("lList is equal to vList");

}

**else**

{

System.*out*.println("lList is equal to vList");

}

12) What is the output of the following?

-12

List<Integer> lList = **new** LinkedList<Integer>(Arrays.*asList*(1,2,3,4,5,6));

System.*out*.println(lList.indexOf(1.0f));

13) What is the output of the following?

1

List<Integer> lList = **new** LinkedList<Integer>(Arrays.*asList*(1,2,3,4,5,6));

System.*out*.println(lList.indexOf(2));

14) What is the output of the following?

fooList is not equal to fooVector

**public class** Foo

{

**public int** bar;

**public** Foo(**int** bar\_)

{

bar = bar\_;

}

}

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

{

List<Foo> fooList = **new** ArrayList<Foo>(Arrays.*asList*(**new** Foo(1), **new** Foo(2)));

List<Foo> fooVector = **new** Vector<Foo>(Arrays.*asList*(**new** Foo(1), **new** Foo(2)));

**if**(fooList.equals(fooVector))

{

System.*out*.println("fooList is equal to fooVector");

}

**else**

{

System.*out*.println("fooList is not equal to fooVector");

}

}

15) What is the output of the following?

-1

List<Foo> fooList = **new** ArrayList<Foo>(Arrays.*asList*(**new** Foo(1), **new** Foo(2))); List<Foo> fooVector = **new** Vector<Foo>(Arrays.*asList*(**new** Foo(1), **new** Foo(2))); System.*out*.println(fooList.indexOf(**new** Foo(2)));

16) What is the output of the following?

5

List<Integer> aList = **new** ArrayList(Arrays.*asList*(1,2,3,4,5,3,6));

System.*out*.println(aList.lastIndexOf(3));

17) What is the output of the following?

[4, 5, 6]

List<Integer> aList = **new** ArrayList(Arrays.*asList*(1,2,3,4,5,3,6)); aList.removeAll(Arrays.*asList*(1,2,3)); System.*out*.println(aList);

18) What is the output of the following?

No elements removed

[1, 2, 3, 4, 5, 3, 6]

List<Integer> aList = **new** ArrayList(Arrays.*asList*(1,2,3,4,5,3,6)); **if**(!aList.removeAll(Arrays.*asList*(7,8,9,10))){

System.*out*.println("No elements removed");

}

System.*out*.println(aList);

19) What is the output of the following?

[]

List<Integer> aList = **new** ArrayList(Arrays.*asList*(1,2,3,4,5,3,6)); **if**(!aList.retainAll(Arrays.*asList*(1.0f,2.0f))){

System.*out*.println("List not changed");

}

System.*out*.println(aList);

20) What is the output of the following?

List not changed

[1, 2, 3, 4, 5, 3, 6]

List<Integer> aList = **new** ArrayList(Arrays.*asList*(1,2,3,4,5,3,6)); **if**(!aList.retainAll(Arrays.*asList*(1,2,3,4,5,6))){

System.*out*.println("List not changed");

}

System.*out*.println(aList);

}

1. Write a function, public static int[] toArray(List<Integer> list), to convert a List<Integer> to an array of int. Use List.toArray.
2. What is the output of the following?

[3, 4, 5]

List<Integer> aList = **new** ArrayList(Arrays.*asList*(1,2,3,4,5,3,6));

System.*out*.println(aList.subList(2, aList.lastIndexOf(3)));

1. Write a function, public static void printList(List<String> list) that replicates the printing of System.out.println(list).

*public static void* printList(List<String> list)  
{  
 System.*out*.print("[");  
 *for* (*int* i = *0*; i < list.size() - *1*; i++)  
 {  
 System.*out*.print(list.get(i) + ", ");  
 }  
   
 System.*out*.print(list.get(list.size() - *1*) + "]");  
   
}

1. Will the following run without issue?

No

List<String> animals = **new** ArrayList(Arrays.*asList*("dog","cat","sheep")); Iterator<String> iter = animals.iterator(); **while**(iter.hasNext())

{

iter.next();

iter.remove();

iter.next();

iter.remove();

}

* 1. Using ListIterator write a function, public static void printNumbersDigits(int number). This function should create a List containing the digits of the number as elements. Then using a ListIterator traverse the List backwards printing out the digits separated by commas. E.x. 1,2,3,4,5,6

1. Write a function, public static void setAll(String name, String set, List<String> list), this function should replace all instances of name with set.
2. Should you prefer raw types to generics?
3. Yes
4. No

28) Which of the following is the correct way to get a value from a raw type List?

1. **int** a = (Integer)list.get(0);
2. **int** b = list<integer>.get(0);
3. **int** c =list.get(0);
4. **int** d d=list.get(0) asInteger;
5. List is a?
   1. **Interface**
   2. Class
   3. Object
   4. Method

Answer: a

30) Which of the following implement List?

1. ArrayList
2. Vector
3. Array
4. I only
5. II only
6. I, II, and II
7. **I and II**

Answer: d

31) What is the output of the following?

List<Integer> list = **new** ArrayList(3);

list.add(**new** Integer(1));

list.add(**new** Integer(2));

list.add(**new** Integer(1));

Integer one = list.get(0);

Integer two = list.get(2);

**if**(one == two)

System.*out*.println(one + " is equql to " + two); **else**

System.*out*.println(one + " is NOT equal to " + two);

1. 1 is equal to 1
2. 2 is NOT equal to 1
3. 1 is NOT equal to 2
4. **1 is NOT equal to 1**

Answer: d

32) What is the output of the following?

List<Integer> list = **new** ArrayList();

ListIterator<Integer> lIter = list.listIterator();

lIter.add(0);

lIter.add(1);

System.*out*.println(list);

1. No output
2. **[0, 1]**
3. [1, 0]
4. [1, 1]

Answer: b

33) What is the output of the following?

List<Integer> list = **new** ArrayList(Arrays.*asList*(1,2,1,3,1,4,1,5)); ListIterator<Integer> lIter = list.listIterator();

**while**(lIter.hasNext())

{

**if**(lIter.next() == 1)

{

lIter.add(lIter.next() + 1);

}

}

1. **[1, 2, 3, 2, 3, 4, 1, 4, 5, 1, 5, 6]**
2. [1, 2, 3, 2, 3, 4, 3, 4, 5, 4, 5, 6]
3. [1, 2, 3, 2, 3, 4, 1, 4, 5, 2, 5, 6]
4. No output, infinite loop

Answer: a