

# Forum: [Hack Forums](#)

Thread: [Java Contact Recorder](#)

**Consulting Detective** - July 31, 2018, 1:40 am

It has been 2 weeks since I last posted in the Java section.

I have learned heaps since then. Constructors, encapsulation, polymorphism, composition, inheritance, pass by value vs pass by reference and so on. Right now I'm practicing ArrayLists. Just want to share something I made. It is a simple phonebook, it can search, add, delete and modify contacts. I know there is no input sanitation in my code, but I'm getting my hands dirty on that later.

PHP Code

```
<?php
import java.util.Scanner;

public class MobilePhone{

    public static void main(String[] args){

        Contacts c = new Contacts();
        Scanner scan = new Scanner(System.in);

        int choice = -1;

        System.out.println("-----Sherlock Phone-----");
        System.out.println("[1]Search Contact");
        System.out.println("[2]Add Contact");
        System.out.println("[3]Delete Contact");
        System.out.println("[4]Modify Contact");
        System.out.println("[0]Exit");

        while(choice != 0){
            System.out.println("-----");
            System.out.print("What would you like to do? ");
            choice = scan.nextInt();

            switch(choice){
                case 1:
                    System.out.print("Name to search: ");
                    c.getContact(scan.next());
                    break;
                case 2:
                    System.out.print("Name to add: ");
                    if(c.addContact(scan.next())){
                        System.out.print("Enter phone number: ");
                        c.addPhoneNumber(scan.nextInt());
                    }
                    break;
                case 3:
```

```

        System.out.print("Name to delete: ");
        c.delContact(scan.next());
        break;
    case 4:
        int tempIndex;
        int tempPhoneNumber;
        String tempName;
        System.out.print("Enter contact to modify: ");
        tempIndex = c.modContact(scan.next());
        if(tempIndex >= 0){
            System.out.print("Enter new name: ");
            tempName = scan.next();
            c.updateName(tempIndex, tempName);
            System.out.print("Enter new phone number: ");
            tempPhoneNumber = scan.nextInt();
            c.updatePhoneNumber(tempIndex, tempPhoneNumber);
        }else{
            System.out.println("Contact does not exist.");
        }
        break;
    case 0:
        break;
    default:
    }
}
}
}
}

```

#### PHP Code

```

<?php
import java.util.ArrayList;

class Contacts{

    private ArrayList<String> names = new ArrayList<String>();
    private ArrayList<Integer> phoneNumbers = new ArrayList<Integer>();

    //~~~~~

    public void getContact(String name){
        if(names.contains(name)){
            System.out.println("Name:" + name);
            System.out.println("Phone Number: " + getPhoneNumber(names.indexOf(name)));
        }else{
            System.out.println("Contact not found.");
        }
    }

    public int getPhoneNumber(int index){
        return phoneNumbers.get(index);
    }

    //~~~~~

    public boolean addContact(String name){

```

```

        if(names.contains(name)){
            System.out.println("Name already exists.");
            return false;
        }else{
            names.add(name);
        }
        return true;
    }
    public void addPhoneNumber(int phoneNumber){
        phoneNumbers.add(phoneNumber);
        System.out.println("Contact added.");
    }
}
//~~~~~

public void delContact(String name){
    if(names.contains(name)){
        names.remove(name);
        System.out.println("Contact deleted.");
    }else{
        System.out.println("Contact does not exist.");
    }
}
//~~~~~

public int modContact(String name){
    if(names.contains(name)){
        return names.indexOf(name);
    }
    return -1;
}
public void updateName(int index, String name){
    names.set(index, name);
}
public void updatePhoneNumber(int index, int phoneNumber){
    phoneNumbers.set(index, phoneNumber);
}
}

```

**hash** - July 31, 2018, 3:11 am

Congratulations on your improvement! Some things that you can eventually improve on is that you don't want your logic to be spread out. In your main especially case 4 there is a lot of logic involved. You probably want to make it either a function or let another class or function handle it. Also what happens when I insert 2x the same name? ;)

What happens when you spread logic is that when you change your scan class or want to modify something in terms of contacts the code fails in multiple classes. Also if you're busy with ArrayLists check how to make your own! I cannot post any links yet but search for Java Enumerator class. It's fun to play around with that as well!

**Tenno** - July 31, 2018, 8:39 am

If you want to write a bit more generic, you can declare an arraylist like this:  
 PHP Code

```
<?php
private List<Integer> myList = new ArrayList<>()
```

So if you want to change it to a LinkedList for example, you just type new LinkedList instead of ArrayList.

Anyways, you've definitely improved!

**Consulting Detective** - August 1, 2018, 12:33 am

(Jul 31, 2018 03:11 AM)Allumination Wrote: [\[CITE\]](#) Congratulations on your improvement! Some things that you can eventually improve on is that you don't want your logic to be spread out. In your main especially case 4 there is a lot of logic involved. You probably want to make it either a function or let another class or function handle it. Also what happens when I insert 2x the same name? ;)

What happens when you spread logic is that when you change your scan class or want to modify something in terms of contacts the code fails in multiple classes. Also if you're busy with ArrayLists check how to make your own! I cannot post any links yet but search for Java Enumerator class. It's fun to play around with that as well!

Case 4 does look a bit cluttered. Guess I need to learn which code should be a separated and be implemented as a method instead. But doesn't that ruin the pattern I have? If I'm going to put the logic of a case in a method? Maybe I should do the same for all the other cases?

(Jul 31, 2018 08:39 AM)Tenno Wrote: [\[CITE\]](#) If you want to write a bit more generic, you can declare an arraylist like this:

PHP Code

```
<?php
private List<Integer> myList = new ArrayList<>()
```

So if you want to change it to a LinkedList for example, you just type new LinkedList instead of ArrayList.

Anyways, you've definitely improved!

Thanks :)

I haven't looked at linked lists yet. I will do that now.

Holy shit. I just made my own int ArrayList. This is awesome.

PHP Code

```
<?php
import java.util.ArrayList;

public class SherlockArrayLists{

    public static void main(String[] args){

        ArrayList<CustomInts> ci = new ArrayList<CustomInts>();

        for(int i=0; i<10; i++){
            ci.add(new CustomInts(i + 10));
        }

        for(CustomInts temp : ci){
```

```

        System.out.println(temp.getInt());
    }
}

//~~~~~

```

```

class CustomInts{

```

```

    private int i;

```

```

    public CustomInts(int i){
        this.i = i;
    }

```

```

    public int getInt(){
        return i;
    }
}

```

hash - August 1, 2018, 3:15 am

(Aug 1, 2018 12:33 AM)Consulting Detective Wrote: [\[CITE\]](#) Case 4 does look a bit cluttered. Guess I need to learn which code should be a separated and be implemented as a method instead. But doesn't that ruin the pattern I have? If I'm going to put the logic of a case in a method? Maybe I should do the same for all the other cases?

Thanks :)

I haven't looked at linked lists yet. I will do that now.

Holy shit. I just made my own int ArrayList. This is awesome.

PHP Code

```

<?php

```

```

import java.util.ArrayList;

```

```

public class SherlockArrayLists{

```

```

    public static void main(String[] args){

```

```

        ArrayList<CustomInts> ci = new ArrayList<CustomInts>();

```

```

        for(int i=0; i<10; i++){
            ci.add(new CustomInts(i + 10));
        }

```

```

        for(CustomInts temp : ci){
            System.out.println(temp.getInt());
        }
    }
}

```

```
}
```

```
//~~~~~
```

```
class CustomInts{
```

```
    private int i;
```

```
    public CustomInts(int i){
```

```
        this.i = i;
```

```
    }
```

```
    public int getInt(){
```

```
        return i;
```

```
    }
```

```
}
```

For now it's okay since it's not a very complicated method. But if you are going to add input validation you can make a method for that and re-use it for all inputs. Another tip is to always comment a method or case. What goes in and what will be returned. It's a good thing to start with early because commenting is not as easy as it looks.

If you want to study more in-depth about programming in Java study Gang of Four patterns and Applying UML and patterns. Helps to program in a more professional way.