Dealing with lots of values

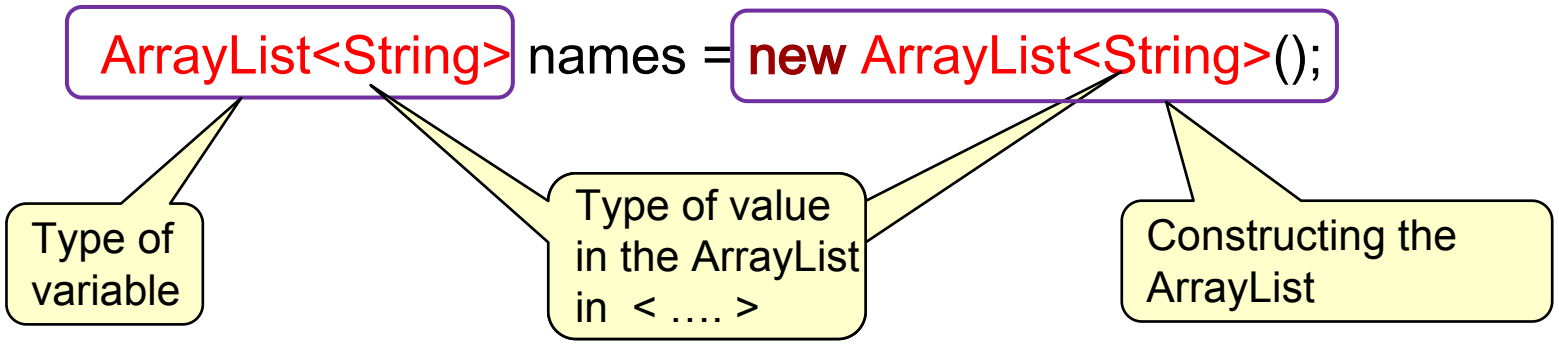
- We've used ArrayLists (and Lists)
  - Road Profiler,
  - WordSearcher, SalesVisualiser, FileEditor,
- ArrayLists of numbers, Strings, other objects.
- Created by methods
  - UI.askNumbers(…) and UI.askStrings(…)
  - Files.readAllLines(Path.of(filename)) (actually, gave us a List, not ArrayList)
- Used **for** each loops to step through items in an ArrayList
- What more can you do with an ArrayList?

Using ArrayLists

- How can we create a new ArrayList?
- How can we add items to an ArrayList?
- How can we access items in an ArrayList without having to step through the whole list.
- How can we modify an ArrayList?

Using ArrayList: create new

- Creating empty ArrayList:



  - Must specify the type of value
  - Can't be int or double or boolean!!!
    Have to use Integer or Double or Boolean ("wrapper" objects)

    ```
    ArrayList<Integer> counts = new ArrayList<Integer>();
    ```

  - Must have the ( ) - standard constructor, with no arguments.
  - Must Import java.util.ArrayList

Using ArrayList: adding to the end

- Adding an item at the end of an ArrayList: .add(….) method

```
ArrayList<Double> values = new ArrayList<Double>();
for (int i = 1;  i <= 10; i++){
     values.add (UI.askDouble("Enter "+ i + “th number"));
}
```

- value to be added must be of the right type for the list.

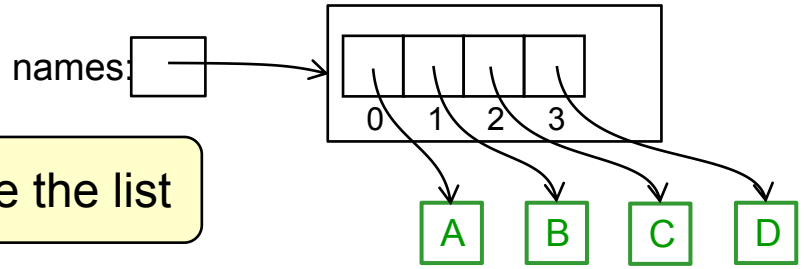
```
ArrayList<String> lines = new ArrayList<String>();
UI.println("Enter lines, end with empty line");
String line = UI.askString(">");
while (! line.equals("") ) {
     lines.add (line);
     line = UI.askString(“>"));
}
```

Acting on each item in an ArrayList:

- Acting on every item from 0 … *end* ⇒ use a simple for each loop:
  - Rule: MUST NOT CHANGE THE LIST INSIDE THE FOR EACH LOOP!!
  - Print out all the names:

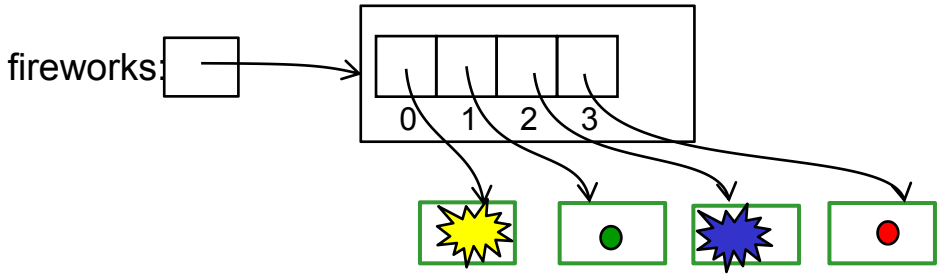
```
for (String name :  names) {
    UI.println( "Hello " +  name);
}
```

Doesn't change the list




```
for (double num :  numbers) {
    if (num > 100) {
        numbers.add(num);
    }
}
```

```
for (Firework firework : fireworks) {
    firework.step();
}
```



Changes the values in the firework objects, but doesn't change the list – the firework objects are still the same objects.

Garden Program: Flower class

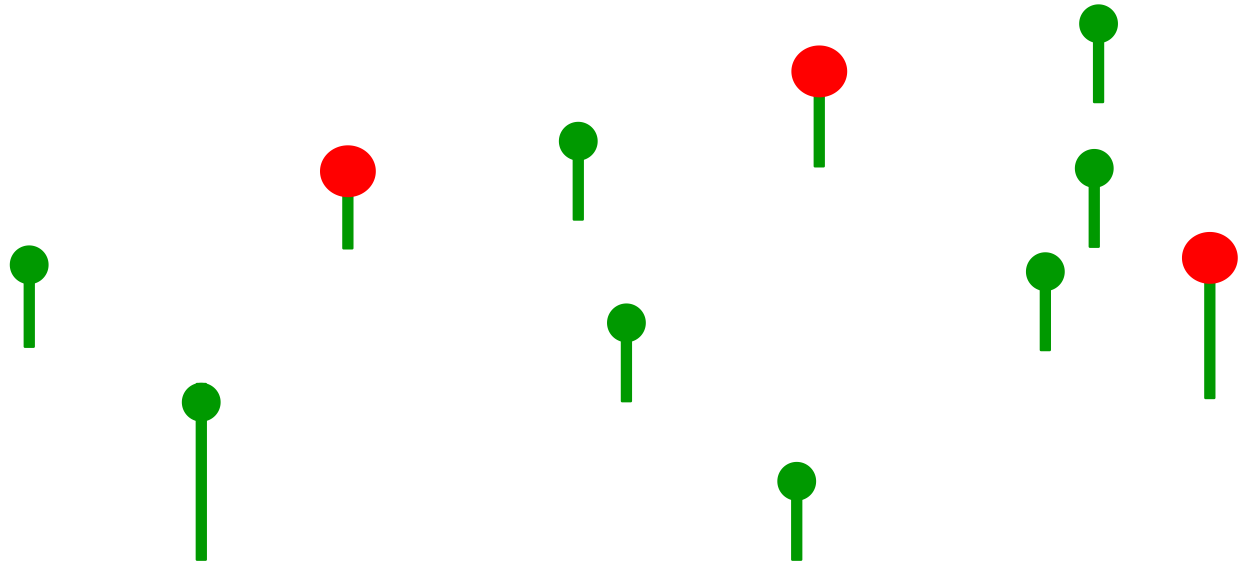
Flower(double x, double y)

void bloom() Make the flower bloom, if it is in the "bud" stage

void grow(int v) make the flower taller, if it is in the "bud" or "bloom" stage.

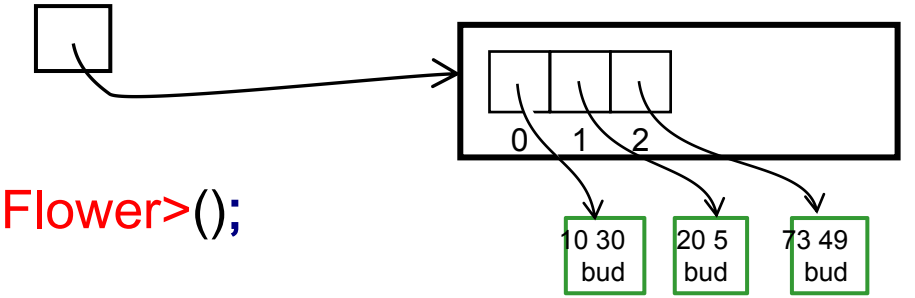
void pick() Make the flower half its height, if in the "bud" or the "bloom" stage,

boolean touching(double x, double y) Is the point x,y on top of the flower?

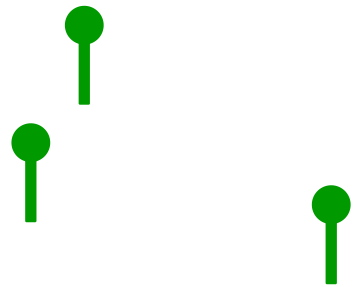


Garden Program

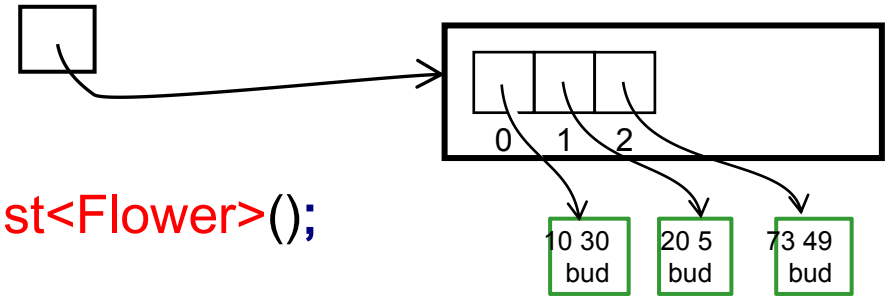
- Needs to keep track of all the flowers that have been planted
  - needs and ArrayList of Flowers:




```
public class Garden {
    private  ArrayList<Flower> flowers = new  ArrayList<Flower>();
```



Garden Program

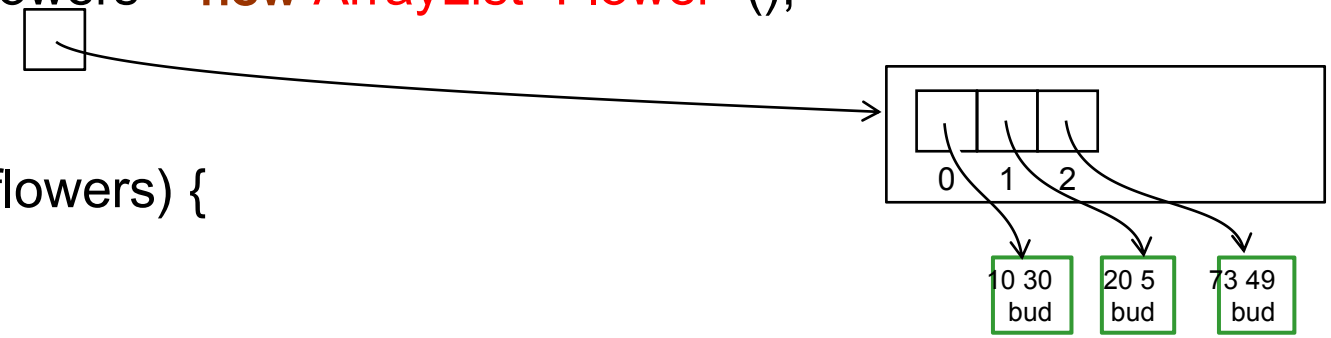


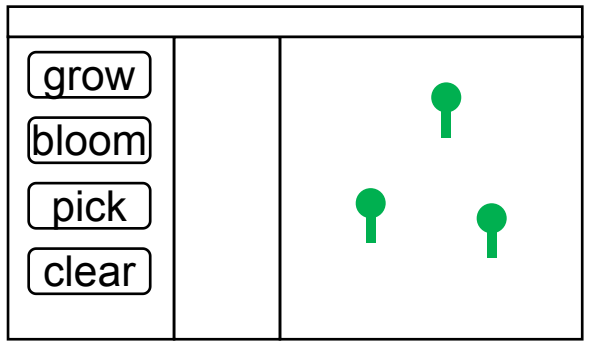

```
public class Garden {
    private  ArrayList<Flower> flowers = new  ArrayList<Flower>();

    public void setupGUI(){
        UI.setMouseListener( this ::doMouse);
        UI.addButton("Grow",   this::doGrow);
        UI.addButton("Bloom", this::doBloom);
        UI.addButton("Pick",     this::doPick);
        UI.addButton("Clear",   this::doClear);
    }

    public  void doMouse(String action, double x, double y){
        if (action.equals("released")){
            Flower fl = new Flower(x, y);          // or    this.flowers.add(new Flower(x,y));
            this.flowers.add(fl);
        }
    }
```

Garden program

- Operate on the ArrayList of Flowers:

```
private ArrayList<Flower> flowers = new ArrayList<Flower>();

public void doGrow(){
    for (Flower flower : this.flowers) {
        flower.grow();
    }
}
public void doBloom(){
    for (Flower flower : this.flowers) {
        flower.bloom();
    }
}
public void doClear(){
    UI.clearGraphics();
    this.flowers.clear();
    //or   this.flowers = new ArrayList<Flower>();
}
```

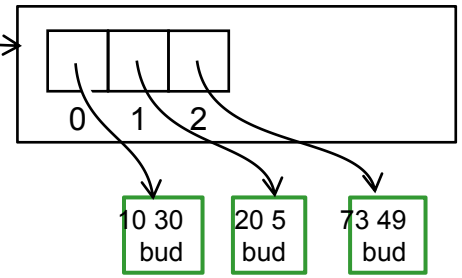


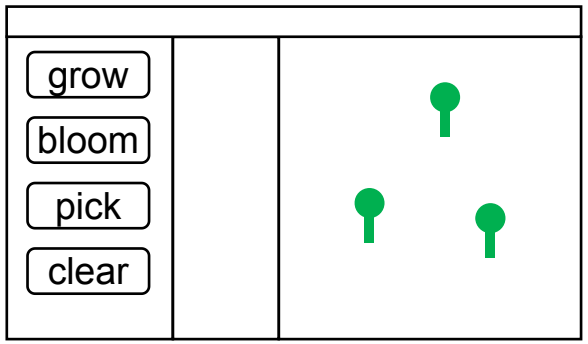


Garden program: Selecting Flowers

- Operate on a *selected* Flower:

```
private ArrayList<Flower> flowers = new ArrayList<Flower>();

private Flower selectedFlower;


public void doGrowOne(){
    this.selectedFlower.grow(10);
}
public void doBloomOne(){
    this.selectedFlower.bloom();
}
public void doPickOne(){
    this.selectedFlower.pick();
}
```

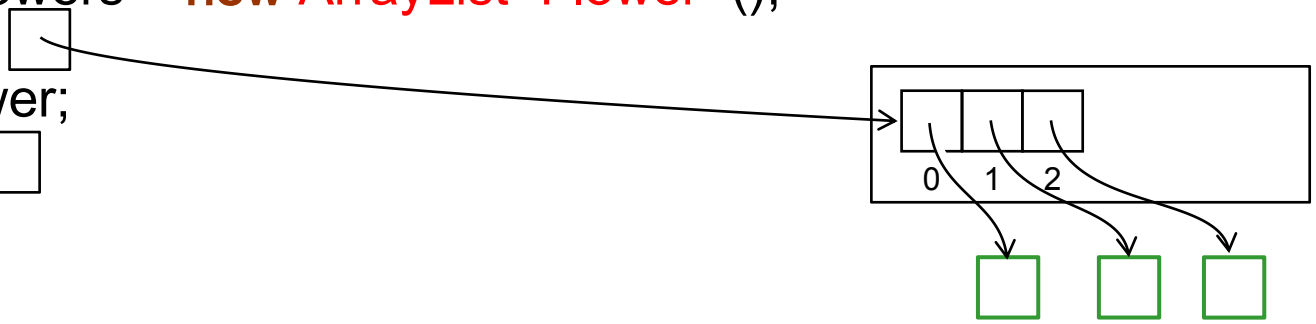




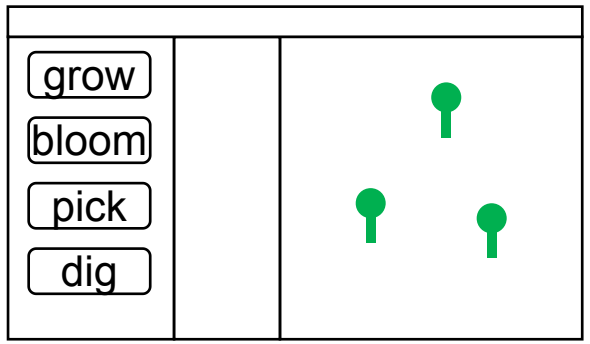
Garden program : Selecting Flowers

- Operate on a *selected* Flower:

```
private ArrayList<Flower> flowers = new ArrayList<Flower>();

private Flower  selectedFlower;

    :
```



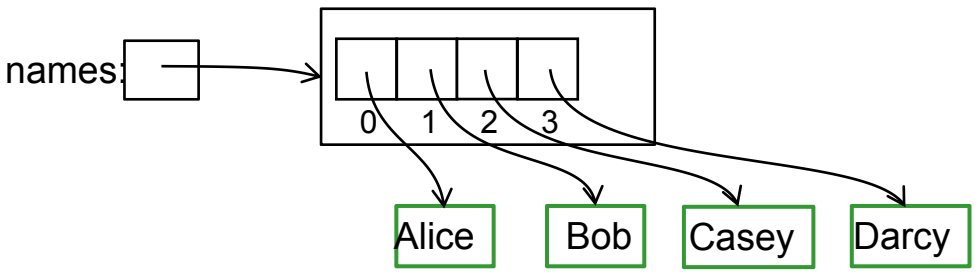

```
public void doMouse(String action, double x, double y){
    if (action.equals("released")){
        for (Flower flower : this.flowers) {
            if  (flower.touching(x, y) ) {
                this.selectedFlower = flower ;
                return ;
            }
        }   // if not touching a flower, plant one
        this.flowers.add(new Flower(x, y));
    }
}
```



Doing more with ArrayLists

- How do we
  - add a new value in the middle of an ArrayList?
  - access the values in the ArrayList, except by stepping through the whole list?
  - change the value at an index in the ArrayList?
  - remove a value from the ArrayList?
  - find out how big the ArrayList is?
  - clear the ArrayList?

- Items in an ArrayList are numbered: with an index starting at 0:



Using ArrayList

- For all actions, call methods on the ArrayList:
  - size(), add(…), get(…), set(…), remove(…), clear(), contains(…), indexOf(…), isEmpty(), …

```
ArrayList<String> names = new ArrayList<String>();

names.add("Jim");

names.add("Jan");

if (names.get(0).equals("Jim") {
    names.set(0, "Jane");
    …

 names.add(1, "Bob");

 names.remove("Bob");

names.remove(0);
```

Adds an item at the end of the list

Gets the item at a position

Replaces the item at a position with a new item

Adds an item at a position

Removes item (1st occurrence)

Removes item at a position