Today’s topics

Java
  Arrays
Upcoming
  Functions

Reading
  Great Ideas, Chapter 3
Arrays

- Aggregate data type
- Deal with items of same type
  - Lists of words
  - Numbers
- Analogies
  - Mailboxes in post office
  - CD racks with slots
- Simplifies naming
  - What if you needed to come up with unique name for each data item?
- Allows use of loops
- Required for many mathematical and statistical problems
- Multiple elements or cells
• Use *subscript* or *index* to access an element
  \[x[5] = 20;\]
  \[foo.setText("Result is " + x[5]);\]
• First element is element 0, not 1!!!
• Often used in loops
  \[
  \text{int } k = 0, \text{ sum } = 0; \\
  \text{while } ( k < 10 ) \\
  \{
  \text{sum } = \text{sum } + \text{measurements}[k]; \\
  k = k + 1;
  \}
  \]
• Note that subscript is a variable, *k*
Creating Arrays

- **Declaration**
  
  
  ```
  double weights[];
  ```

- **Definition**
  
  ```
  weights = new double[50];
  ```

- **Combine declaration and definition**
  
  ```
  double weights[] = new double[50];
  ```

  ```
  int num[] = new int[6];
  ```


  ```
  ```

Arrays & Loops

```
num[0] = 0;
int k = 2;
while(k < num.length)
{
    num[k] = k * k;
    k = k + 1;
}
```

| 0 | 21 | 4 | 9 | 16 | 25 |

• Subscript range errors!!!
  • Java checks (many languages do not)
  • Costs & tradeoffs
Array Examples

- Sum up elements in a list (4 ways to do same thing)

  ```java
  int k = 0, sum = 0;
  while (k < 10)
  { sum = sum + data[k];
    k = k + 1;
  }
  int k = 1, sum = 0;
  while (k <= 10)
  { sum = sum + data[k - 1];
    k = k + 1;
  }

  int k = 9, sum = 0;
  while (k >= 0)
  { sum = sum + data[k];
    k = k - 1;
  }
  int k = 10, sum = 0;
  while (k > 0)
  { k = k - 1;
    sum = sum + data[k];
  }
  ```

- Count occurrences of something
- Search for something
- Information retrieval
public class Hotel extends java.applet.Applet implements ActionListener {
    TextField mInstruc, mHotelCensus;
    IntField gRoomNo, gNoGuests;
    Button bRegister;
    int k=0, totGuests = 0, noOccupied = 0, roomNo, noGuests;
    int room[];

    public void init() {
        room = new int[500];
        k = 0;
        while (k < 500) {
            room[k] = 0;
            k = k + 1;
        }
    }
}
mInstruct = new TextField(60);
mInstruct.setText("Enter room number, number of guests, then press Register");

gRoomNo = new IntField(6);
gNoGuests = new IntField(6);
bRegister = new Button("Register");
mHotelCensus = new TextField(60);
bRegister.addActionListener(this);
add(mInstruct);
add(gRoomNo);
add(gNoGuests);
add(bRegister);
add(mHotelCensus);
public void actionPerformed(ActionEvent event) {
    Object cause = event.getSource();
    if (cause == bRegister) {
        roomNo = gRoomNo.getInt();
        noGuests = gNoGuests.getInt();
        if (room[roomNo] != 0)
            mHotelCensus.setText("That room is occupied!");
        else {
            room[roomNo] = noGuests;
            totGuests = totGuests + noGuests;
            noOccupied = noOccupied + 1;
            mHotelCensus.setText("There are " + totGuests +
                                  " occupying " + noOccupied + " rooms.");
        }
    }
}
Simple Statistics

What should a simple statistics program include?

• Get data into array
  • One item at a time
  • Practical program would use files
• Allow display of data
  • Display one item at a time
  • (Could have used TextArea to display all at once)
• Actual computations
  • Maximum, Minimum, Mean, N
• Control
  • Appropriate buttons
Simple Statistics

- How do we compute the Mean (Average)?
  - Sum
  - Count
  - Compute
- How do we find extrema?
  - Largest
  - Smallest
- Will package as functions (methods)
public class ArrayStats extends java.applet.Applet implements ActionListener
{
    TextField mInstruct, mAnswer;
    IntField iCount;
    double list[];
    Button bStore, bShow, bExtremes, bMean, bClear;
    int count, nextFree, nextUse;

double mean(double[] list, int size) {
    int k = 0;
    double sum = 0.0;
    while (k < size) {
        sum = sum + list[k];
        k = k + 1;
    }
    return sum/size;
}
Stats Program.2

double max(double[] list, int size)
{
    int k = 1;
    double largest = list[0];
    while (k < size) {
        if (list[k] > largest) {
            largest = list[k];
        }
        k = k + 1;
    }
    return largest;
}
double min(double[] list, int size)
{
    int k = 1;
    double smallest = list[0];
    while (k < size) {
        if (list[k] < smallest) {
            smallest = list[k];
        }
        k = k + 1;
    }
    return smallest;
}
public void init() {
    list = new double[100];
    mInstruct = new TextField(70);
    mAnswer = new TextField(70);
    mInstruct.setText("Enter Value, then press Store button");
    iCount = new IntField(10);
    bStore = new Button("Store");
    bShow = new Button("Show");
    bExtremes = new Button("Extremes");
    bMean = new Button("Mean");
    bClear = new Button("Clear");
    nextFree = 0;
    nextUse = 0;
    bStore.addActionListener(this);
    bShow.addActionListener(this);
    bExtremes.addActionListener(this);
    bMean.addActionListener(this);
    bClear.addActionListener(this);
    add(mInstruct); add(iCount); add(bStore); add(bShow);
    add(bExtremes); add(bMean); add(bClear); add(mAnswer);
}
public void actionPerformed(ActionEvent event)
{
    int value, total;
    Object cause = event.getSource();
    if (cause == bStore) {
        value = iCount.getInt();
        list[nextFree] = value;
        nextFree = nextFree + 1;
        iCount.setInt();  // clear IntField
    }
    if (cause == bShow) {
        mAnswer.setText("The value in element "+nextUse+" is "+
                        list[nextUse]);

        nextUse = (nextUse + 1)% nextFree;
    }
}
if (cause == bExtremes){
    mAnswer.setText("The largest data item is " + max(list, nextFree) + " and the smallest data item is " + min(list, nextFree));
}
if (cause == bMean) {
    mAnswer.setText("The average is " + mean(list, nextFree));
}
if (cause == bClear) {
    nextUse = 0;
    nextFree = 0;
    mAnswer.setText("The old data has been cleared out");
}
}
New Stuff in Stats Program

• Java Programming: Functions (Methods)
  • Parameters/Arguments
  • Return statement
  • Return type on method header (not just void)
• Control
  • Entering and Displaying data
• Algorithms
  • Mean
  • Min
  • Max