Arrays and Strings

- Arrays - homogeneous indexed collections
- Selection sort as array example
  - Introduction to complexity analysis
- String class, revisited

Arrays

- Homogeneous collections of elements of same type
- Indexed by integers
  - Element 0, Element 1, Element 2, etc.
- Used to store values related to one another
- Number of elements specified at creation time
- Can also initialize entire array with values
- Aggregate data type

BNF

<array-decl> →<type> <id> [ ] = new

<type> [<limit>];

<array-decl> →<type> <id> [ ] = {<values>};

Array has limit elements, 0 through limit-1

Note: [ ] are terminals in the language here, not *[ ]* which mean optional construct.

```
char grades [ ]= {'A','B','C','D','E'};
int months [ ] = new int [12];
if (grades[2]=='C') ....
```

Array Properties

- **Array size is fixed at creation time**
- **Equivalent declarations:**

```
1. int A [] = new int [4];
2. int A [];  A = new int[4];
```

- **Array assignment does not copy values**

  `int B [] = A;` **will associate the same array with references A and B**

- **Can use an array element whereever the same type variable could be used**

Array Properties

- **Access to elements is checked for staying within the declared array bounds, 0 - (limit-1)**
  - **ArrayIndexOutOf BoundsException can be raised**
- **`length` property stores declared length of an array**
  - **`A.length` is 4**
- **Arrays can be used as arguments to methods, like variables**
- **Arrays can be instance variables of a class**

Example

```
//finds largest element in an
// integer array
public static int arraymax(int[]x){
  int largest = x[0];
  for (int i=1; i<x.length; i++){
    if(x[i]>largest)largest=x[i];
  }
 return largest;
}
```

Histogram Program Fragment

```
static final int numofStudents = 10;
static final int points = 150;
public static void main (String[] args)
  throws IOException{
  int score [] = new int [numofStudents];
  int bins [] = new int [(points/10+1)];
  int i, s;
  //read in test scores for class
  for (i=0; i<numofStudents; i++){
      s = score[i]/10;//calculate bin#
      bins[s]++;}
  ...
```

see examples hist.java, hist.output

Histogram Program Fragment

```
//print histogram
for (i=1; i<points/10+1; i++){
   System.out.print(i*10 + ":");
   for (s=0; s<bins[i]; s++){
      System.out.print("*");
   }
   System.out.println();
}
}//end of main
```

Sorting Algorithms

- Sorts algorithms used to produce ordered data, either numerically or lexacographically ordered
- Often data is stored in arrays whose elements are moved around until they are in (ascending or descending) order
- Many different methods of varying complexity

First Approach

- Have array of integers: 8 9 10 3 5 7 4 to sort.

new array	original array
	8 9 10 3 5 7 4
3	8 9 10 _ 5 7 4
3 4	8 9 10 _ 5 7 _
3 4 5	8 9 10 _ _ 7 _
3 4 5 7	8 9 10 _ _ _ _
3 4 5 7 8	_ 9 10 _ _ _ _
3 4 5 7 8 9	_ _ 10 _ _ _ _
3 4 5 7 8 9 10	_ _ _ _ _ _ _

Second Approach

- **Can we do this with only 1 array?**

original array

8 9 10 3 5 7 4

3 9 10 8 5 7 4

3 4 10 8 5 7 9

3 4 5 8 10 7 9

3 4 5 7 10 8 9

3 4 5 7 8 10 9

3 4 5 7 8 9 10

3 4 5 7 8 9 10

yes, by building the partially sorted entries at the front of the array and exchanging the smallest entry with the one at the boundary of this area each time the unsorted values are searched

Selection sort

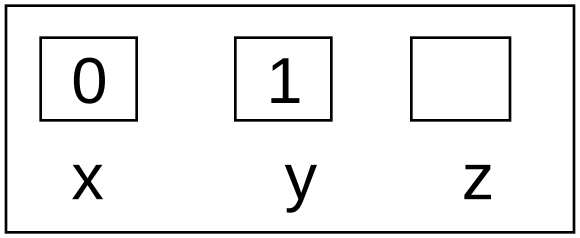
Not very efficient as re-examine all remaining numbers each time.

Selection Sort - Exerpt

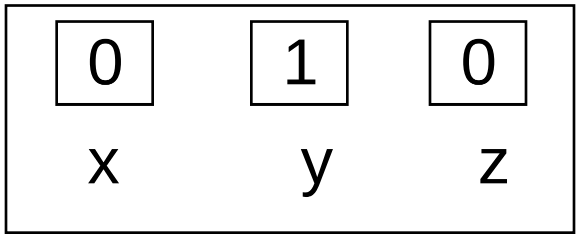
```
//to sort descending exchange > for <
void selection Sort(int [] a){
  int tmp, chosen;
  for(int left=0; left<a.length-1; left++){
     chosen = left;//first unsorted number
     for (int j=left+1; j<a.length; j++){
     //find smallest unsorted element
          if (a[j]<a[chosen]) chosen=j;}
     //exchange a[chosen] with a[left]
     tmp = a[chosen];
     a[chosen] = a[left];
     a[left] = tmp;
  }}
```

How to interchange values?

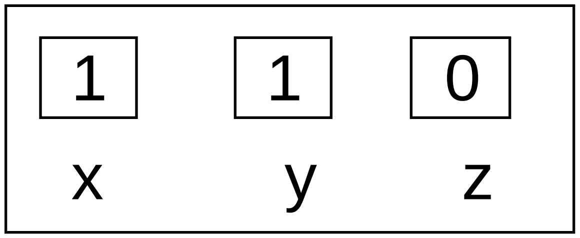
```
int x=0, y=1;
int z;
```



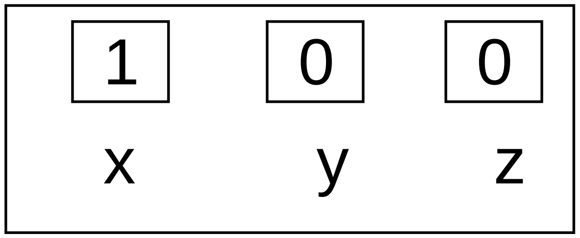
```
z = x;
```



```
x = y;
```



```
y = z;
```



Analysis

iteration	number of comparisons	
1	6	n-1
2	5	n-2
3	4	n-3
4	3	n-4
...		
6	1	n- (n-1)= 1
7	none	

How many comparisons in all?

$$(n-1)+(n-2)+(n-3)+\ldots+3+2+1=(n)(n-1)/2 \sim n^2$$

How slow is n^2 ?

- **Population of US is ~250 million**
  - **2.5e8 * 2.5e8 = 6.25 e16 comparison operations!**
- **Suppose we can execute 100 million operations per second (optimistic)**
  - **6.25 e16 / 1.0 e8 = 6.25 e8 seconds**
  - **8.54 e4 seconds in a day, 3.16 e7 seconds in a year**
  - **Takes 10 years to perform sort!**
  - **1 year per line of code!**

Multidimensional Arrays

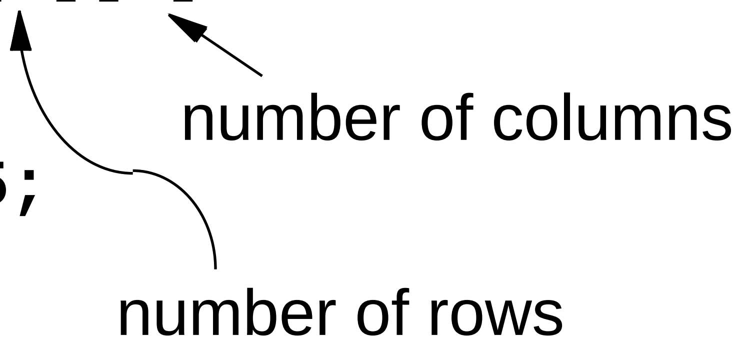
- Use restricted case
  - Only 2D arrays
  - Only reference elements
- Declaration

```
int tmp[ ][ ] = new int [4][2]
```

- Refer to an array element

```
tmp[2][1] = tmp[0][1] + 5;
```

0,0	0,1
1,0	1,1
2,0	2,1
3,0	3,1



String Class

- A Java type that is not primitive
- Special constructor - no new

```
String bigOne = “hippopotamus”;
```

- Immutable values
- Standard interface:
  - length, charAt, equals, compareTo, indexOf, substring, toCharArray
  - allows lexacographic comparison of String objects, substring extraction

String Class Interface

```
int length();//length in chars
char charAt(int index);//returns
//char at position index in string,
//first position is 0.
```

```
String s = “Barbara Ryder”;
char c = s.charAt(0);//c is ‘B’
char c = s.charAt(3);//c is ‘b’
int i = s.length();//i is 13(not 12!)
```

String Class Interface

```
//use instead of == for String
boolean equals (Object obj);
//lexacographic comparison; (-1,0,1)
int compareTo(String s)
```

```
String s = “abc”,t = “abdc”,w = “ab”;
int i;
i = s.compareTo(t);//i is -1
i = s.compareTo(w);//i is 1
i = s.compareTo(“abc”);//i is 0
if (!(s.equals(“abc”)))i = 5;//i is 0
```

String Class Interface

```
int indexOf(String s);
String substring(int begin,int end);
char [] toCharArray();//a class
  //method
```

```
String s = “abc”,t = “abdc”,v;int i;
i = s.indexOf(“ab”);//i is 0
i = s.indexOf(“def”);//i is -1 as
  //substring isn’t in string s
v=s.substring(0,1) + ”e”;//v is “abe”
char [] c; c = toCharArray(t);//makes
  c contain ‘a’,’b’,’d’,’c’
```

Selection Sort for Strings

- **What changes necessary to previous code to handle strings instead of integers?**
  - **String a[ ] parameter**
  - **use of compareTo( ) in if statement**
- **Would be nice to define one sort routine to use with Object a[ ] as long as proper comparison operation is available - idea behind *generics* or *templates* in OOPLs**