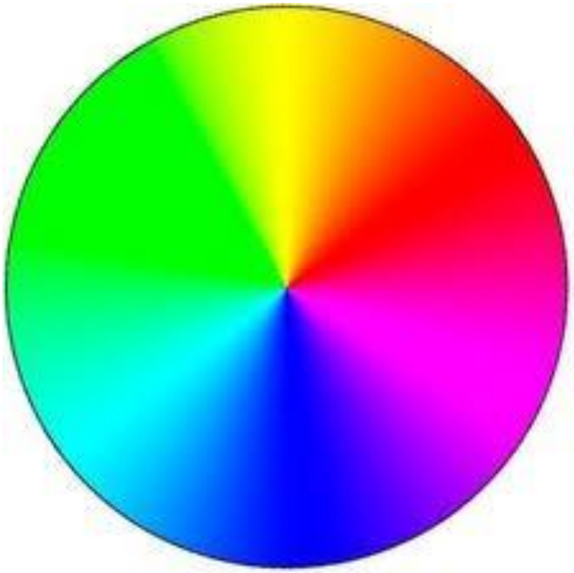
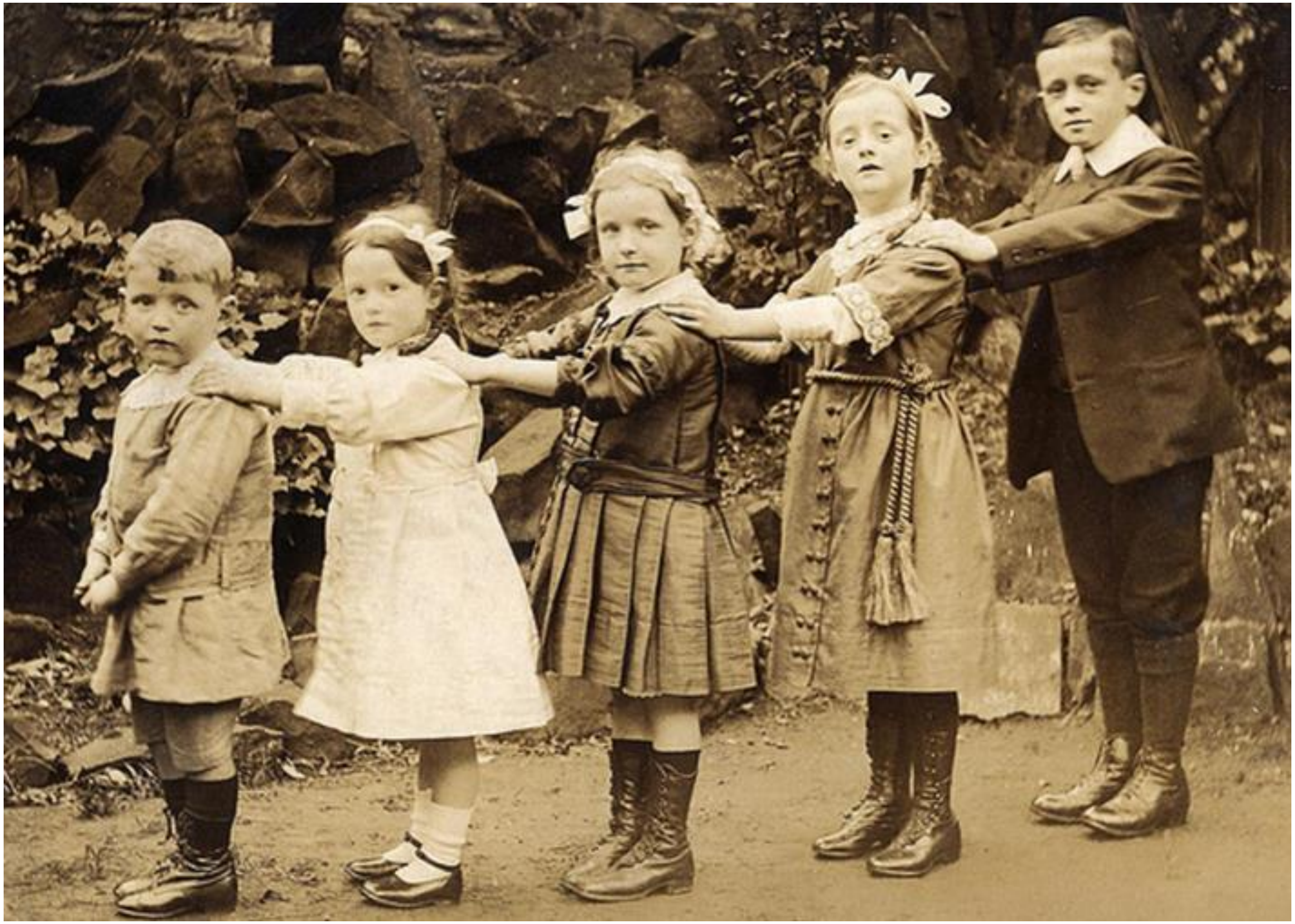
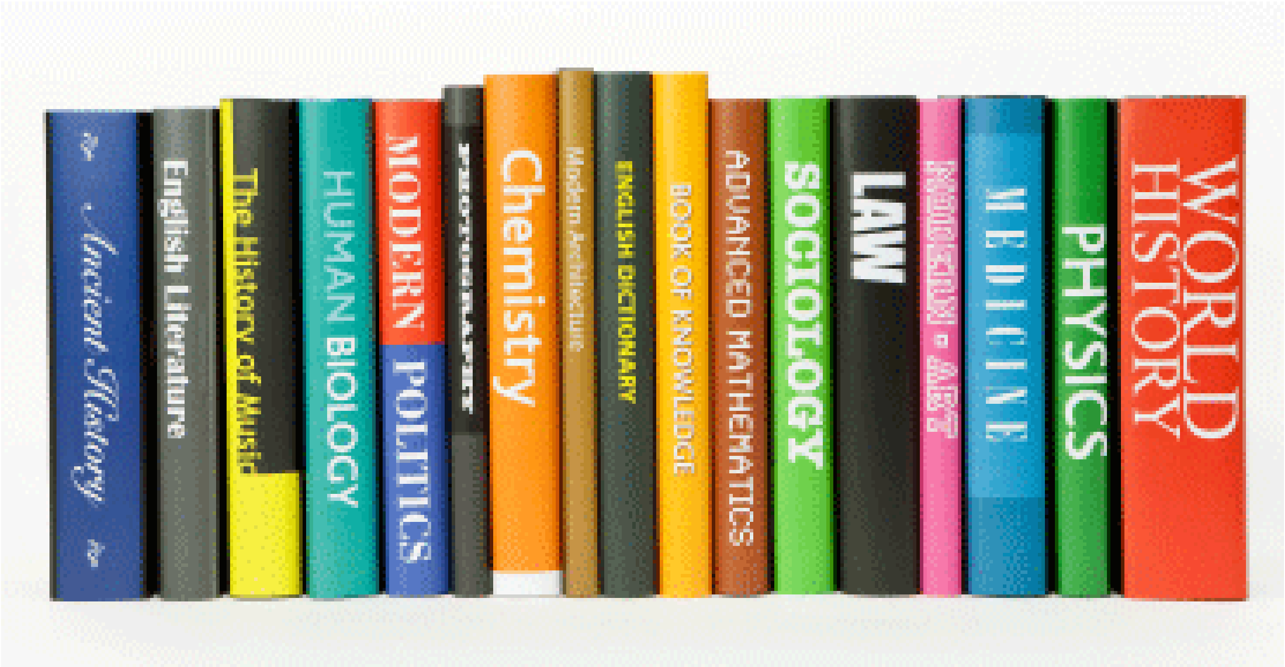
Sorted or Not?

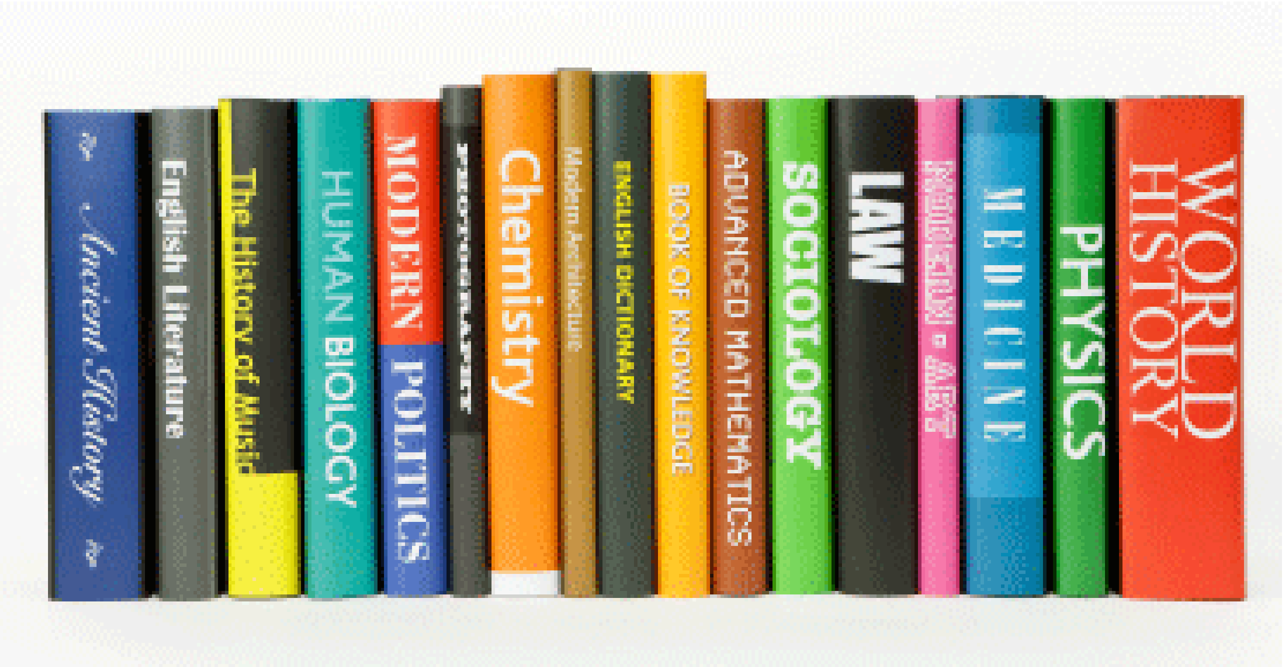
(For those in the group who had trouble with colours)





Yes, in a line. Ordered from shortest to tallest.


Ancient History
English Literature
The History of Music
HUMAN BIOLOGY
MODERN
POLITICS
Chemistry
Modern Architecture
ENGLISH DICTIONARY
BOOK OF KNOWLEDGE
ADVANCED MATHEMATICS
SOCIOLOGY
LAW
MEDICINE
PHYSICS
WORLD
HISTORY



No, while they are in a line, they aren’t in any order.

a
B
C
D
E
F
G
H
I
J

a
B
C
D
E
F
G
H
I
J

Yes, in a line. Ordered alphabetically.

No, not in a line or in any order.

1
2
3
4
5

Yes, in a line and ordered by number.

No, not in a line – they are in groups, but aren’t in order.

Putting Things in Sorted Order

Comparing Things – a reminder...

	String	char, int, double
Less than	X.compareTo(Y)<0	X<Y
Greater than	X.compareTo(Y)>0	X>Y
Equals	X.equals(Y)	X==Y

Why does anyone care about sorting?

Why does anyone care?

- Information is much easier to find in a sorted list.

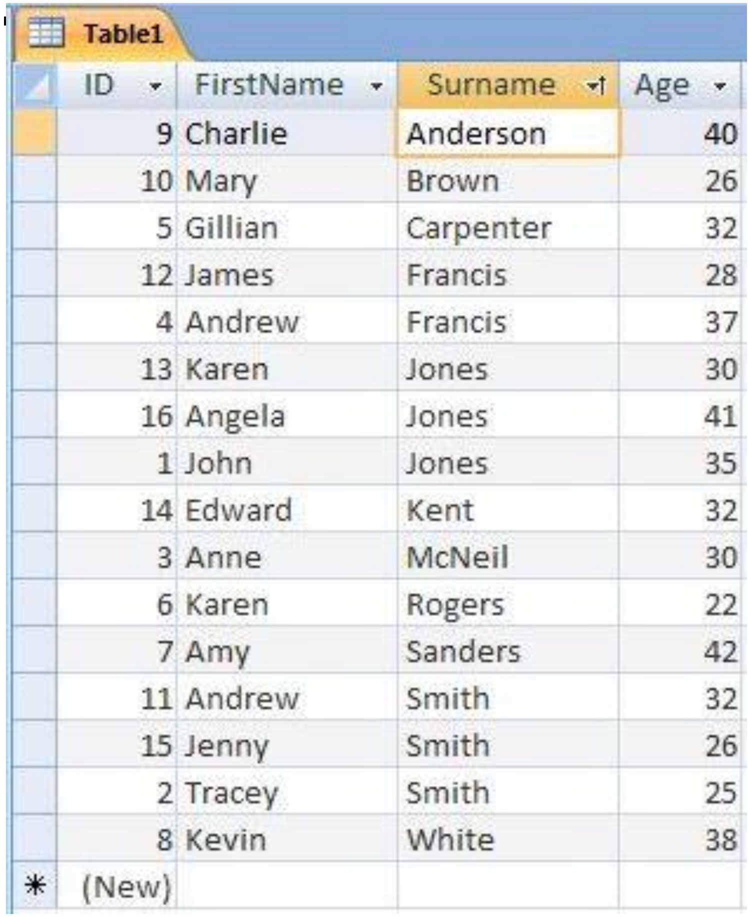
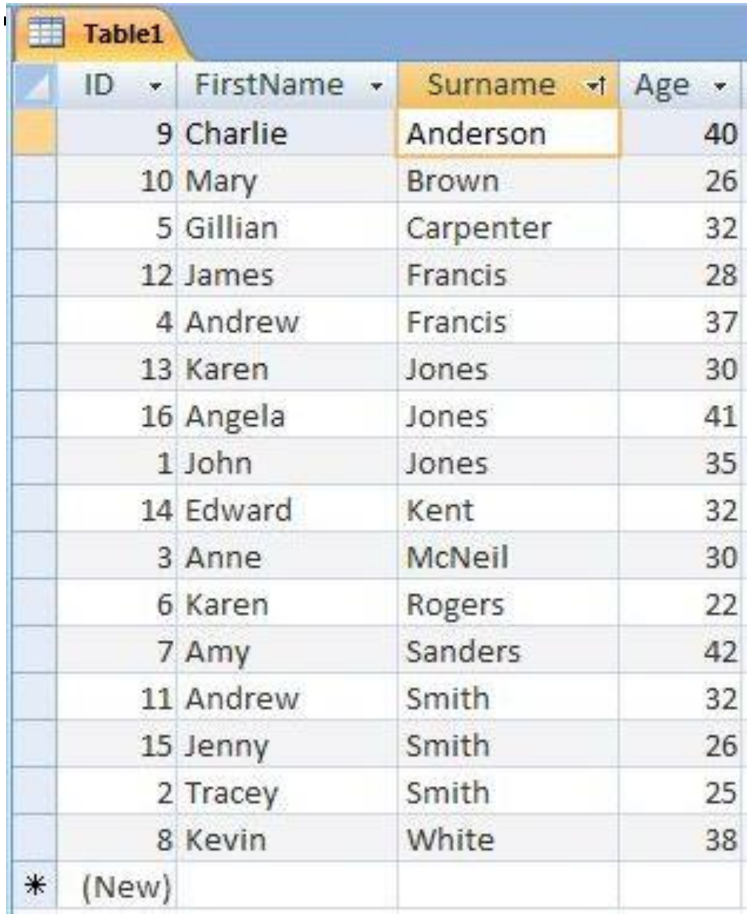


Table1

	ID	FirstName	Surname	Age
	9	Charlie	Anderson	40
	10	Mary	Brown	26
	5	Gillian	Carpenter	32
	12	James	Francis	28
	4	Andrew	Francis	37
	13	Karen	Jones	30
	16	Angela	Jones	41
	1	John	Jones	35
	14	Edward	Kent	32
	3	Anne	McNeil	30
	6	Karen	Rogers	22
	7	Amy	Sanders	42
	11	Andrew	Smith	32
	15	Jenny	Smith	26
	2	Tracey	Smith	25
	8	Kevin	White	38
*	(New)			

Why does anyone care?

- Information is much easier to find in a sorted list.

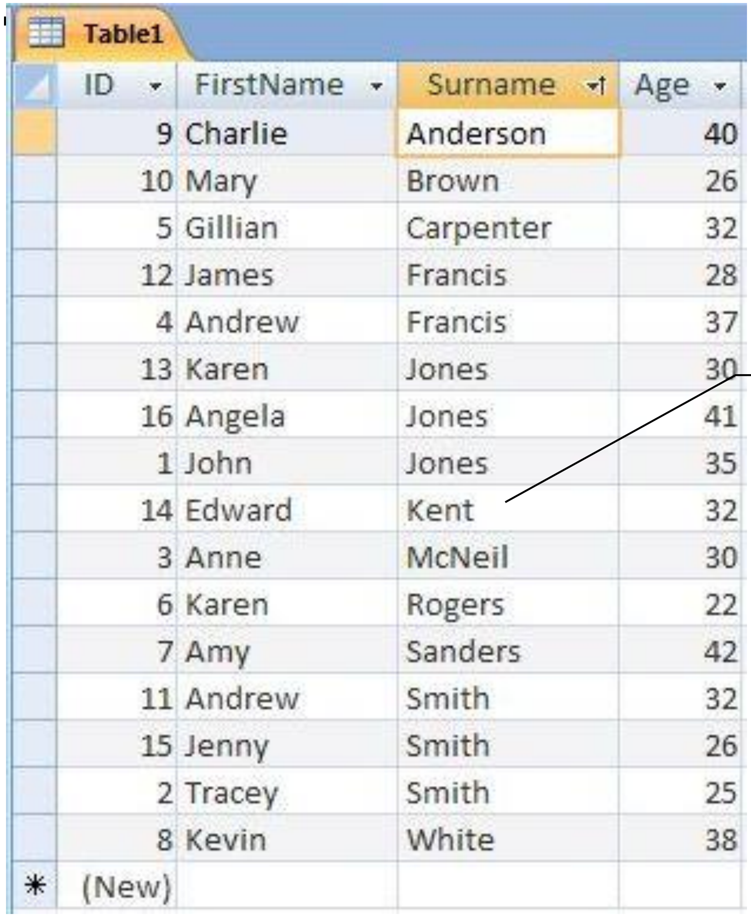


ID	FirstName	Surname	Age
9	Charlie	Anderson	40
10	Mary	Brown	26
5	Gillian	Carpenter	32
12	James	Francis	28
4	Andrew	Francis	37
13	Karen	Jones	30
16	Angela	Jones	41
1	John	Jones	35
14	Edward	Kent	32
3	Anne	McNeil	30
6	Karen	Rogers	22
7	Amy	Sanders	42
11	Andrew	Smith	32
15	Jenny	Smith	26
2	Tracey	Smith	25
8	Kevin	White	38
(New)			

Where’s Kent?

Why does anyone care?

- Information is much easier to find in a sorted list.



ID	FirstName	Surname	Age
9	Charlie	Anderson	40
10	Mary	Brown	26
5	Gillian	Carpenter	32
12	James	Francis	28
4	Andrew	Francis	37
13	Karen	Jones	30
16	Angela	Jones	41
1	John	Jones	35
14	Edward	Kent	32
3	Anne	McNeil	30
6	Karen	Rogers	22
7	Amy	Sanders	42
11	Andrew	Smith	32
15	Jenny	Smith	26
2	Tracey	Smith	25
8	Kevin	White	38
(New)			

Where's Kent?

Here he is. That was easy.

Why does anyone care?

- Information is much easier to find in a sorted list.

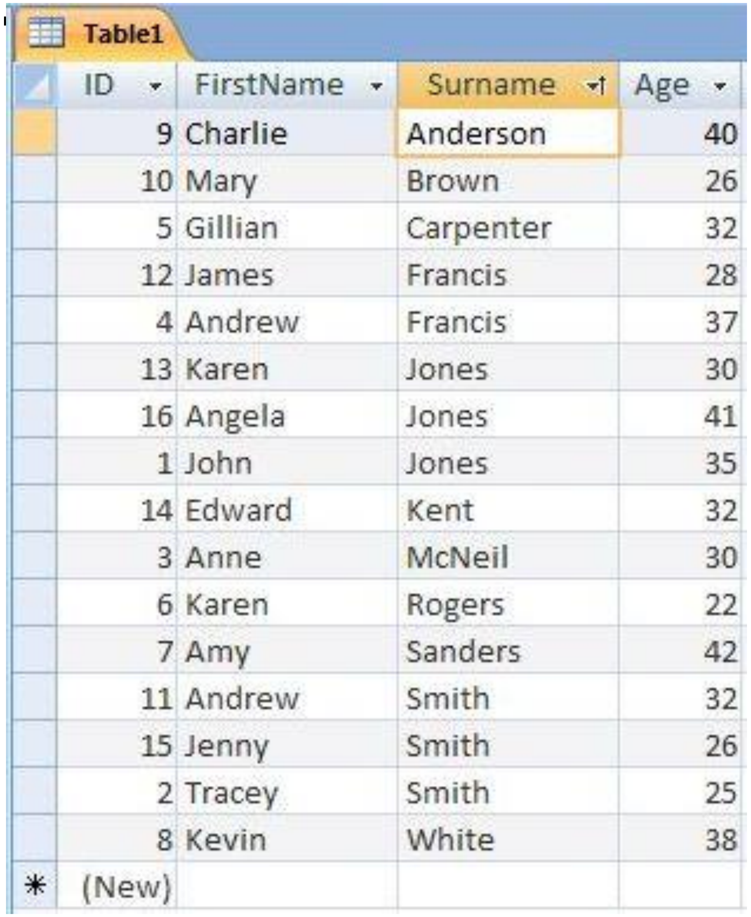


Table1

	ID	FirstName	Surname	Age
	9	Charlie	Anderson	40
	10	Mary	Brown	26
	5	Gillian	Carpenter	32
	12	James	Francis	28
	4	Andrew	Francis	37
	13	Karen	Jones	30
	16	Angela	Jones	41
	1	John	Jones	35
	14	Edward	Kent	32
	3	Anne	McNeil	30
	6	Karen	Rogers	22
	7	Amy	Sanders	42
	11	Andrew	Smith	32
	15	Jenny	Smith	26
	2	Tracey	Smith	25
	8	Kevin	White	38
*	(New)			

Why does anyone care?

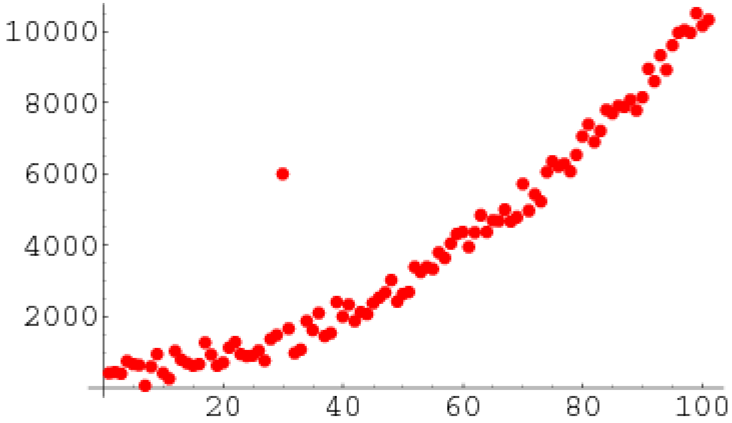
- Telephone directories, dictionaries and book indexes all use alphabetical order, and life would be far more difficult if they didn't.

Why does anyone care?

- If a list of numbers (such as a list of expenses) is sorted into order, the extreme cases are easy to see because they are at the beginning and end of the list.

Why does anyone care?

- If a list of numbers (such as a list of expenses) is sorted into order, the extreme cases are easy to see because they are at the beginning and end of the list.



Why does anyone care?

- Duplicates are also easy to find, because they end up together.

Why do computer scientist care?

- Computers spend a lot of their time sorting things into order, so computer scientists have to find fast and efficient ways of doing this.
- Some of the slower methods such as insertion sort, selection sort and bubble sort can be useful in special situations, but the fast ones such as quicksort are usually used.

Remember Bentley?

N	Cray – great hw, 0(n^3)	TRS-80 – bad hw, O(n)
10	0.000003 sec	0.2 sec
100	0.003 sec	2.0 sec
1000	3 sec	20 sec
2500	47 sec	49 sec
10,000	50 min	3.25 min
100,000	34.7 days	32.5 min
1,000,000	95 years	5.4 hours

Selection Sort

But first… swapping.

Suppose we have:

[0]	[1]
67	83

And we want to swap them:

[0]	[1]
83	67

You can't do this:

[0]	[1]
67	83

```
a [0] = a [1];

a [1] = a [0];
```

[0]	[1]
67	83

```
a [0] = a [1];
```

[0]	[1]
83	83

```
a [1] = a [0];
```

[0]	[1]
67	83

```
a [0] = a [1];
```

[0]	[1]
83	83

```
a [1] = a [0];
```

[0]	[1]
83	83

[0]	[1]
67	83

```
a [0] = a [1];
```

[0]	[1]
83	83

```
a [1] = a [0];
```

[0]	[1]
83	83

We lost 67!

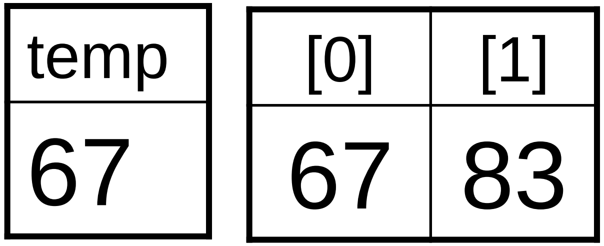
We need a temporary variable
to hold a[0], or 67,
so we don't lose it.

```
int temp = a [0];

a [0] = a [1];

a [1] = temp;
```

temp
67

[0]	[1]
67	83

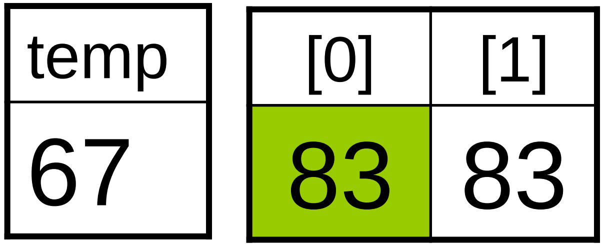
```
int temp = a [0];
```



temp
67

[0]	[1]
67	83

```
a [0] = a [1];
```



temp
67

[0]	[1]
83	83

```
a [1] = temp;
```

```
int temp = a [0];
```

temp
67

[0]	[1]
67	83

```
a [0] = a [1];
```

temp
67

[0]	[1]
83	83

```
a [1] = temp;
```

temp
67

[0]	[1]
83	67

Now the swap works.

Tracing

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8

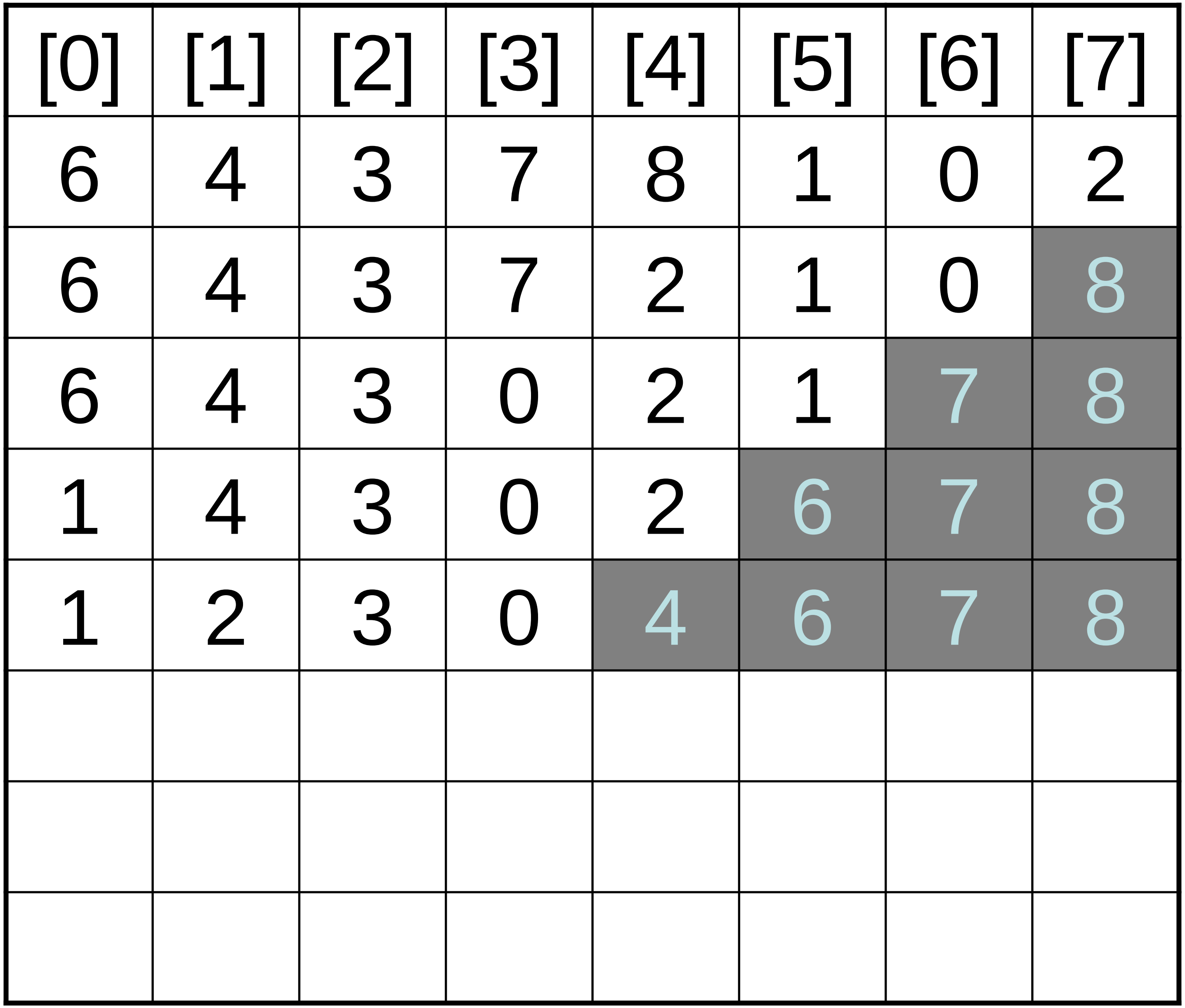
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8



[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8
1	2	0	3	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8
1	2	0	3	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8
1	2	0	3	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8
1	2	0	3	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8
1	2	0	3	4	6	7	8
1	0	2	3	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8
1	2	0	3	4	6	7	8
1	0	2	3	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8
1	2	0	3	4	6	7	8
1	0	2	3	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8
1	2	0	3	4	6	7	8
1	0	2	3	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8
1	2	0	3	4	6	7	8
1	0	2	3	4	6	7	8
0	1	2	3	4	6	7	8

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
6	4	3	7	8	1	0	2
6	4	3	7	2	1	0	8
6	4	3	0	2	1	7	8
1	4	3	0	2	6	7	8
1	2	3	0	4	6	7	8
1	2	0	3	4	6	7	8
1	0	2	3	4	6	7	8
0	1	2	3	4	6	7	8

A very important application of Selection Sort:

“Children are perhaps the greatest advocates of selection sort. Every October, Halloween candies are consumed from best to worst. Whether daily sampling is limited or not, it is clear that choices of the next treat consumed are based on ‘the next biggest piece’ or ‘the next most favorite’ and so on. Children consume treats in decreasing order of acceptability.

Similarly, when we select plants from a greenhouse, check produce in the store or pick strawberries from the farm, we seek the best items first.”

Bailey, Duane. Java Structures. 1999. Pg 80.

A card for you to write:

Selection Sort

- Repeatedly finds the largest number and swaps it into place.
- O(n^2) in all cases. There is no best case.
- It works for all kinds of data, Strings, chars, doubles…

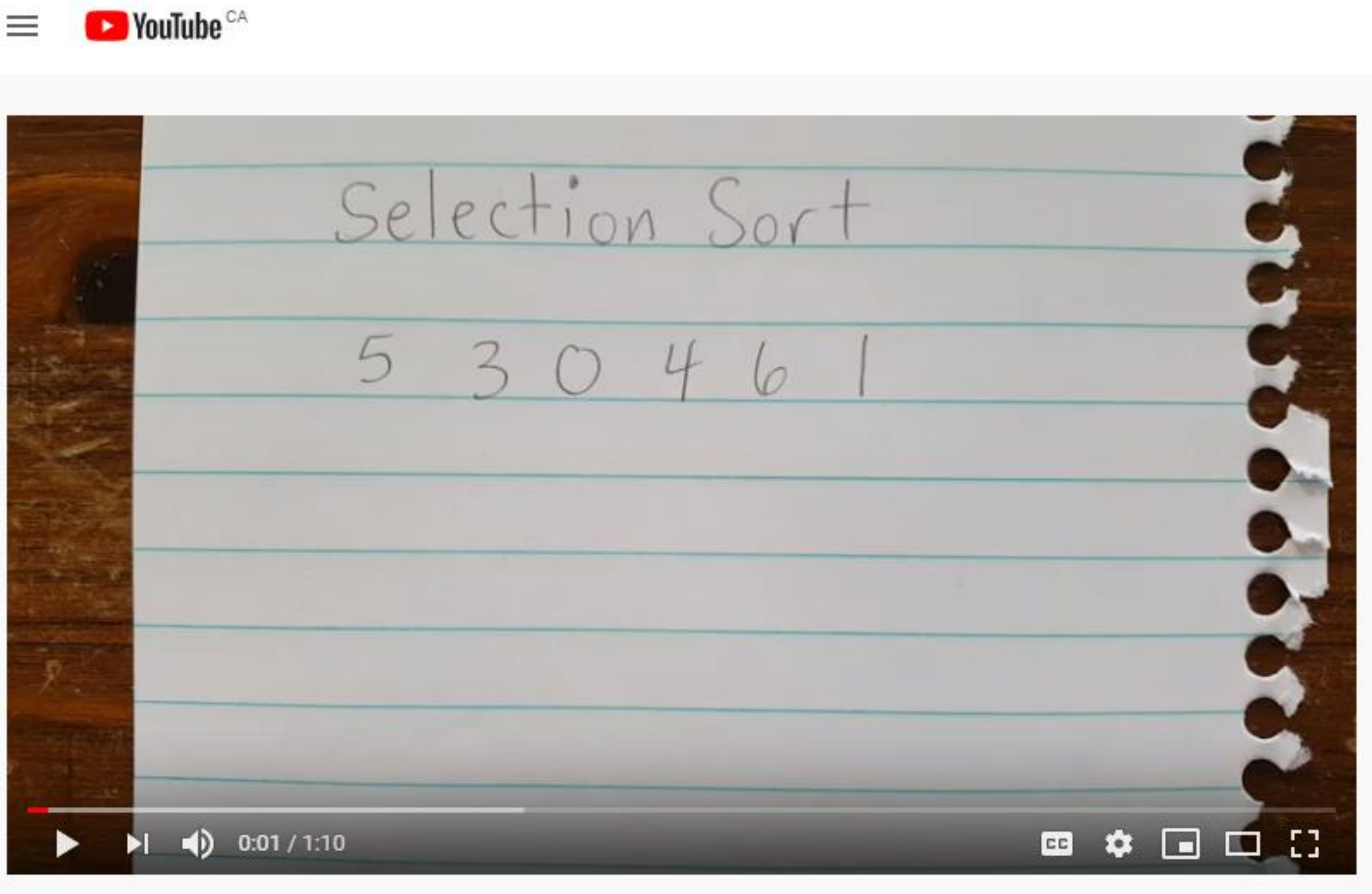
A card for you to write:

Selection Sort Tradeoff

(+) Selection Sort is simple to understand and is simple to code.

(-) It’s simplicity means it isn’t very clever or efficient. That means it is SLOW.

I posted a YouTube video Tracing Selection Sort:



https://www.youtube.com/watch?v=NiKCsHd2K70

To translate it to English…

```
int a[] = {6, 4, 3, 7, 8, 1, 0, 2};
for (int left = a.length - 1 ; left > 0 ; left--)
{
    int max = 0;
    for (int i = 1 ; i < left ; i++)
    {
        if (a [max] < a [i])
            max = i;
    }
    int temp = a [max];
    a [max] = a [left - 1];
    a [left - 1] = temp;
}
```

To translate it to English…

```
Declare the array
for (int left = a.length - 1 ; left > 0 ; left--)
{
    int max = 0;
    for (int i = 1 ; i < left ; i++)
    {
        if (a [max] < a [i])
            max = i;
    }
    int temp = a [max];
    a [max] = a [left - 1];
    a [left - 1] = temp;
}
```

To translate it to English…

```
Declare the array
for (int left = a.length - 1 ; left > 0 ; left--)
{
    Find the max
    int temp = a [max];
    a [max] = a [left - 1];
    a [left - 1] = temp;
}
```

To translate it to English…

```
Declare the array
for (int left = a.length - 1 ; left > 0 ; left--)
{
    Find the max
    Swap the max into place
}
```

To translate it to English…

```
Declare the array
For the non-sorted part of the array
{
    Find the max
    Swap the max into place
}
```