Analysis:

loop 1  takes $\Theta(n)$

if input elements are random numbers in the interval $[0,1)$,
then each bucket expects to get $O(1)$ values,
then each run of insertion sort takes $\Theta(1)$ time

$T(n) = \Theta(n)$ time

What if input element are numbers that aren’t random?
all the inputs could end up in the same bucket,
then running Insertion-Sort on that bucket takes $\Theta(n^2)$ time
Bucket sort: assumes that input is random and in the interval \([0,1)\)

basic idea: divide interval \([0,1)\) into \(n\) equal-sized subintervals (buckets)
  distribute \(n\) input numbers into the buckets
  sort the numbers within each bucket

**Bucket-Sort(A)**

  for \(i = 1\) to \(n\)
    insert \(A[i]\) into \(B[\lfloor nA[i] \rfloor]\)
  for \(i = 0\) to \(n-1\)
    sort list \(B[i]\) using Insertion-Sort
  concatenate the lists \(B[0], B[1], \ldots, B[n-1]\) together in order

\[
\begin{array}{cccccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
\hline
A & 0.78 & 0.17 & 0.39 & 0.26 & 0.72 & 0.94 & 0.21 & 0.12 & 0.23 & 0.68 \\
\hline
B & / & 0.17 & 0.26 & 0.39 / / 0.68 0.78 / 0.94 \\
 & 0.12 & 0.21 & 0.23 & / & / 0.72 & & &
\end{array}
\]
Running time of Counting-Sort:

- loop 1 $\Theta(k)$
- loop 2 $\Theta(n)$
- loop 3 $\Theta(k)$
- loop 4 $\Theta(n)$

$T(n) = \Theta(n+k)$

What if we are given an input that is integers in the range 289 to 1500?

- run Counting-Sort($A$, $B$, 1500-288)
- postprocess the output array: for $i = 1$ to $n$ do $B[i] = B[i] + 288$

What if we do not know anything about input?

we cannot use Counting-Sort
\( k = 6 \)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3</td>
<td>6</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Lecture 6: Counting Sort, Bucket Sort

Counting sort: assumes that each input element is integer in the range 1 to k

Basic idea:
   for each input element x, determine number of elements less than x
   use this to place element x directly into its position in the output array

Counting-Sort(A,B,k) /* input array A, output array B */
   for i = 1 to k
       C[i]=0
   for j = 1 to n
       C[A[j]]=C[A[j]]+1
   /*C[i] now contains the number of elements equal to i*/
   for i = 2 to k
       C[i]=C[i]+C[i-1]
   /*C[i] now contains the number of elements less than or equal to i*/
   for j = n downto 1
       B[C[A[j]]]=A[j]
       C[A[j]]=C[A[j]]-1