Probability in Computing

CS 237

LECTURE 25

Last time

- Exponential distribution
- Poisson process
- Poisson random variables

Today

- Finish Poisson process
- Use of probability in algorithms

Reminder

- HW 12 is due Thursday

Review: Poisson Process

Definition: A Poisson process is a sequence of arrivals over time, where:

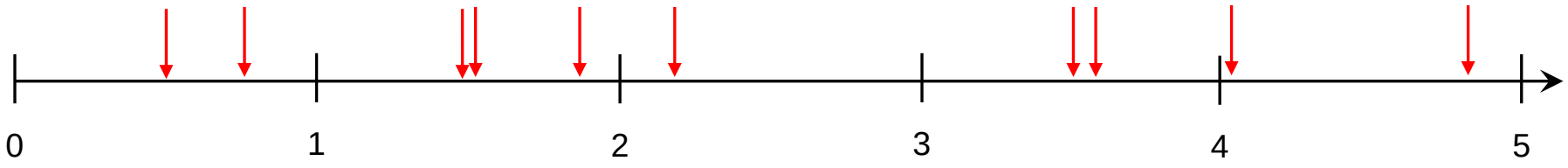
1) The mean arrival rate

$$\lambda = \frac{\text{Mean Number of Arrivals}}{\text{Unit Time}}$$

is constant over all time, for any unit interval [t .. t+1)

2) The number of arrivals in two non-overlapping intervals is independent; and

3) The probability of two arrivals occurring at the same time is 0.



Review: Poisson Random Variables

- Consider a Poisson Process
- Fix the unit time interval (e.g., 1 second or 1 year)
- Supposed the mean number of arrivals in a unit interval is λ
- Let X be the number of arrivals in a unit interval. (It has the same distribution for each unit interval.)
- Then we call X a Poisson Random Variable with rate parameter λ
  Notation: $X \sim \text{Poisson}(\lambda)$

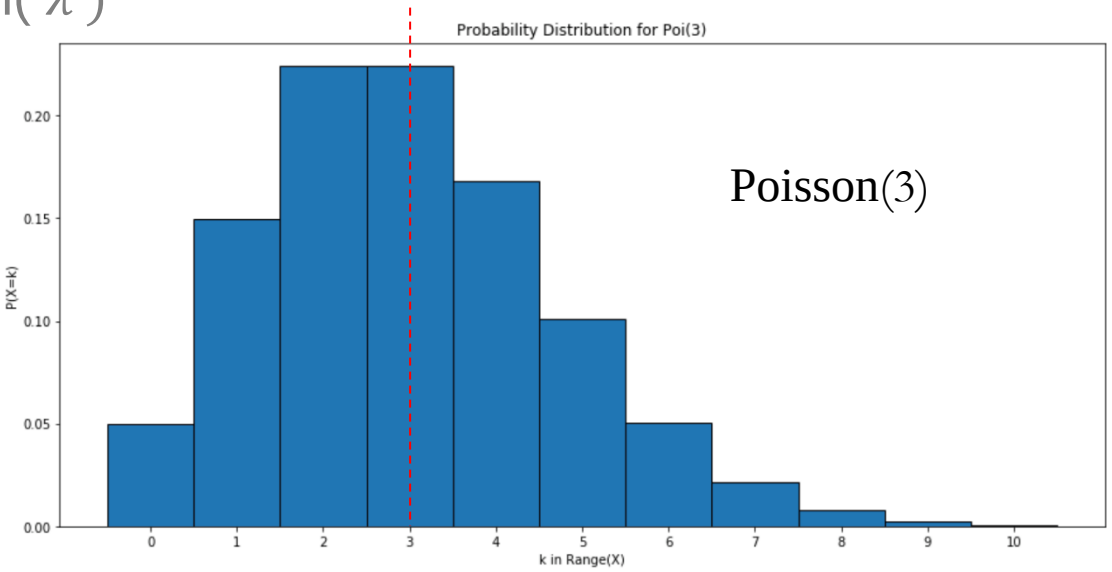
$$\text{Range}(X) = \{0,1,2,\dots\}$$

$$f_X(k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

$$\mathbb{E}(X) = \lambda$$

$$\text{Var}(X) = \lambda$$

Convention: $0! = 1$



Top Hat question (Join Code: 033357)

- Assume that large meteorites hit the earth following the Poisson process, on average, twice per century.

Is it more likely that we will have 0 or 1 large meteorite hit the earth in the next 50 years?

A. 0 is more likely

B. 1 is more likely

C. Equally likely

D. None of the above

Reminder:
If $X \sim Poisson(\lambda)$,

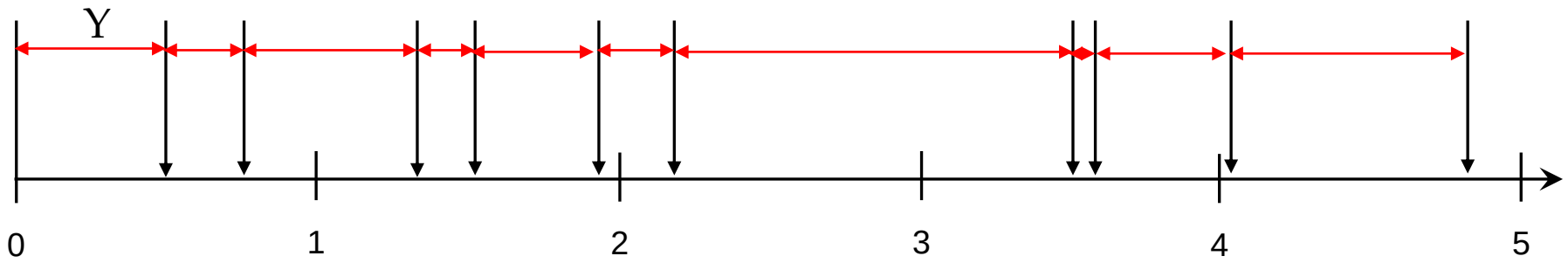
$$f_X(k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

Solution:

- Rate of arrivals per interval of length 50 years is ; so $\lambda =$
- $X \sim Poisson(\ )$
- $\Pr(X = 0) =$
- $\Pr(X = 1) =$

Interarrival Times of Poisson Process

- Let Y be the time until the first arrival.



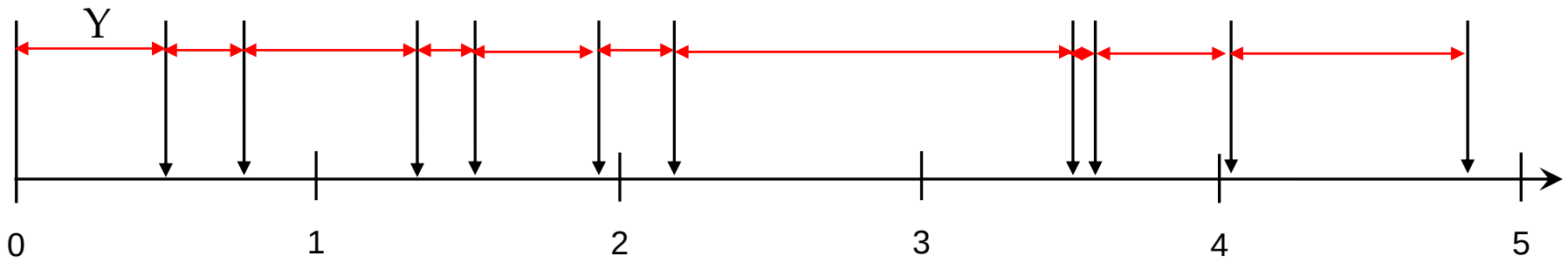
- The arrivals are independent, so at any time t, probabilistically the Poisson process starts all over again (the events don't remember the past!) :

- Y = "the time between any two consecutive arrivals"

What is the distribution of Y?

Interarrival Times of Poisson Process

- Let Y be the time until the first arrival.



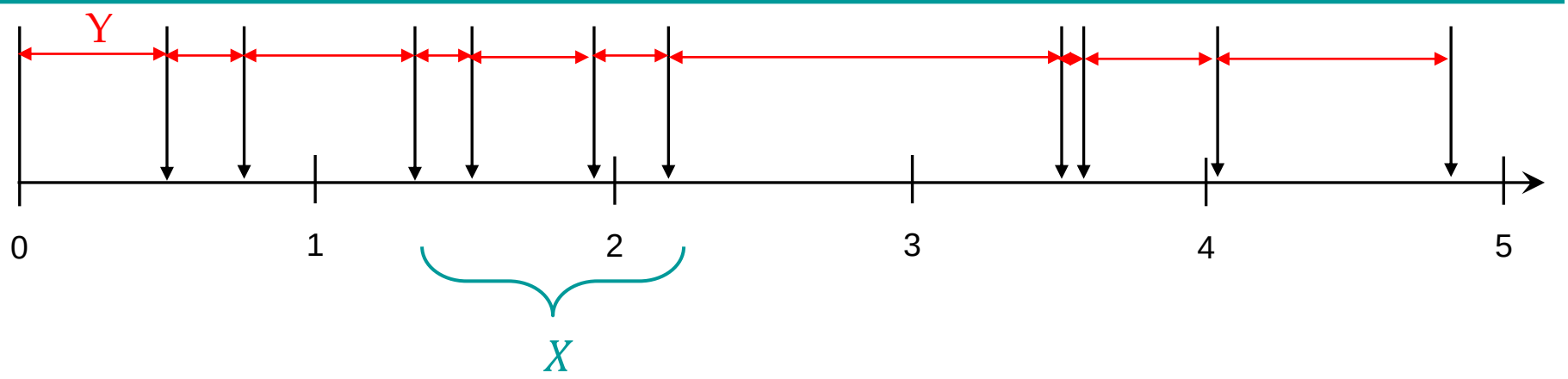
$$\Pr(Y > t) = \Pr(\text{``}no\ arrivals\ in\ the\ interval\ [0, t]''\text{)}$$

Reminder:
If $X \sim Poisson(\mu)$,

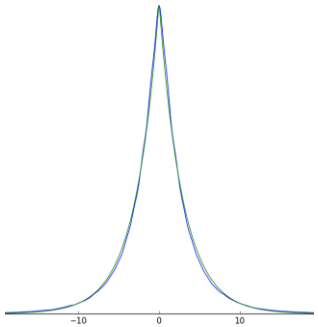
$$f_X(k) = \frac{\mu^k e^{-\mu}}{k!}$$

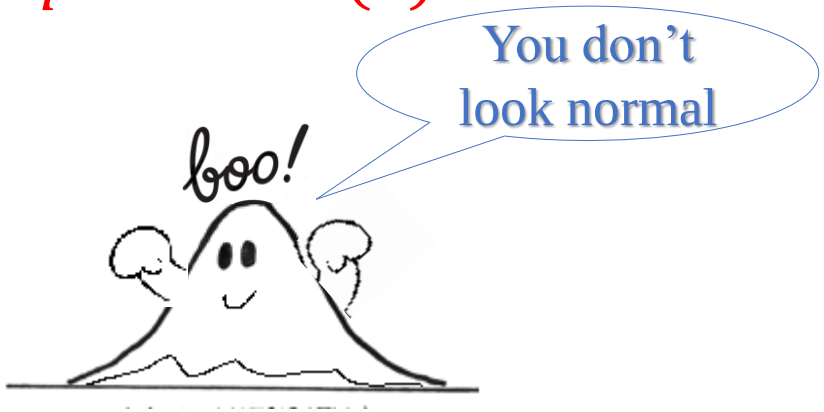
$k =$
Rate of arrivals per interval of length t is

Summary: Poisson Process RVs



- The number of arrivals in a unit interval: $X \sim Poisson(\lambda)$
- The time between arrivals: $Y \sim Exponential(\lambda)$





CS 237

Probability in Algorithms

Probability is used in the design and analysis of algorithms

1. to analyze deterministic algorithms when inputs come from a specified distribution
2. to design and analyze algorithms that use randomness

An Application: Sorting

How to organize your bookshelf?

- We will show how to do it really quickly when your input comes from a uniform distribution

The sorting problem

Input: n numbers $a_1, a_2, \ldots, a_n$

Output: the same numbers in the sorted order:

$$b_1 \leq b_2 \leq \cdots \leq b_n$$

Example:

Input: 8 2 4 9 3 6

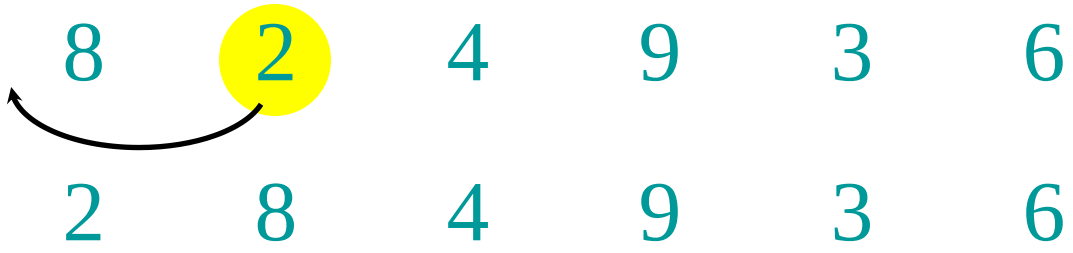
Output: 2 3 4 6 8 9

Insertion sort: example

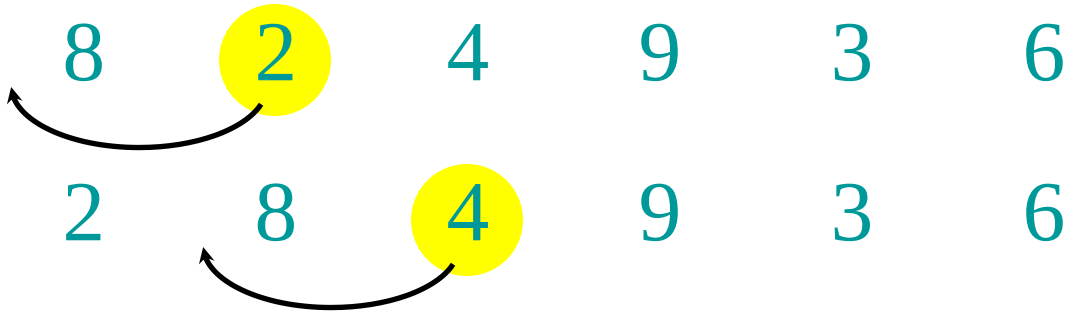
8 2 4 9 3 6

Insertion sort: example

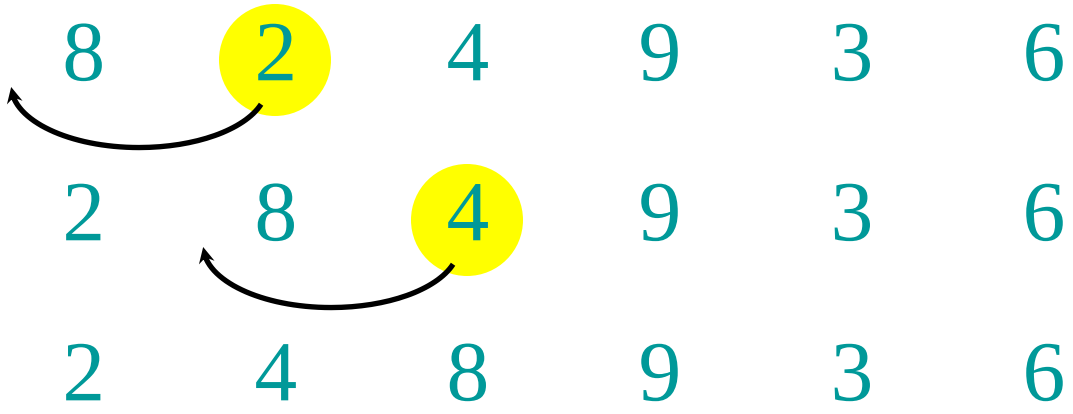
Insertion sort: example



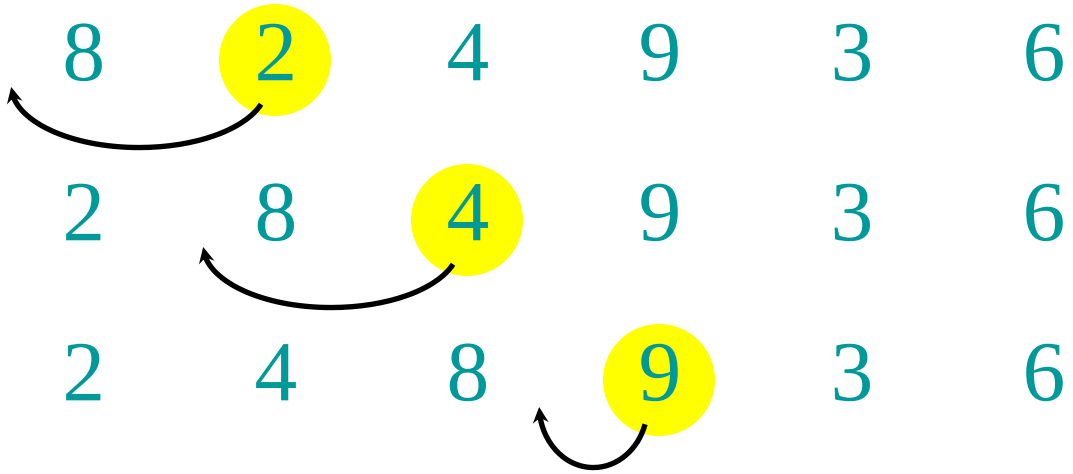
Insertion sort: example



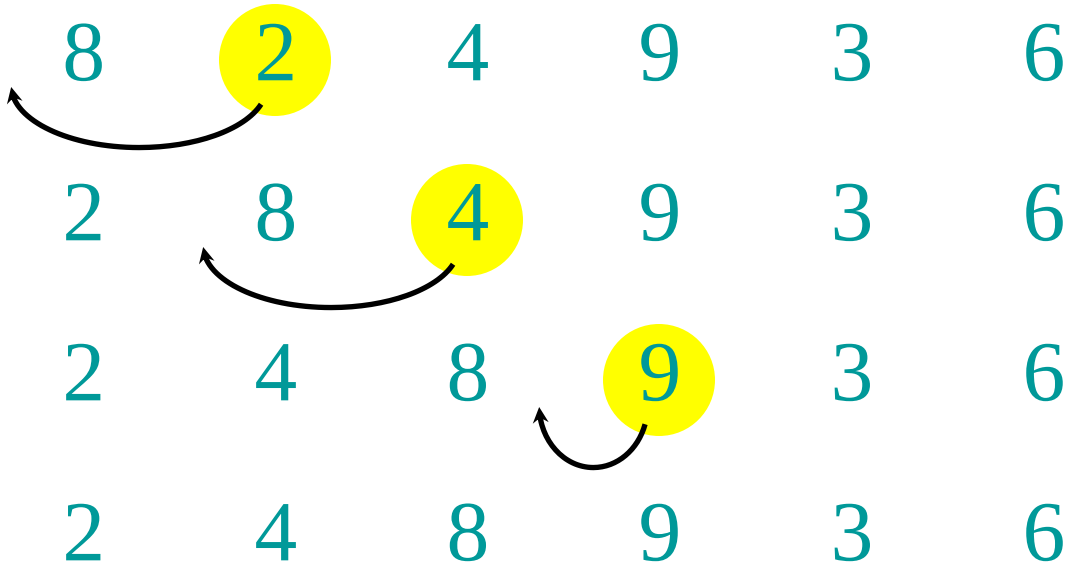
Insertion sort: example



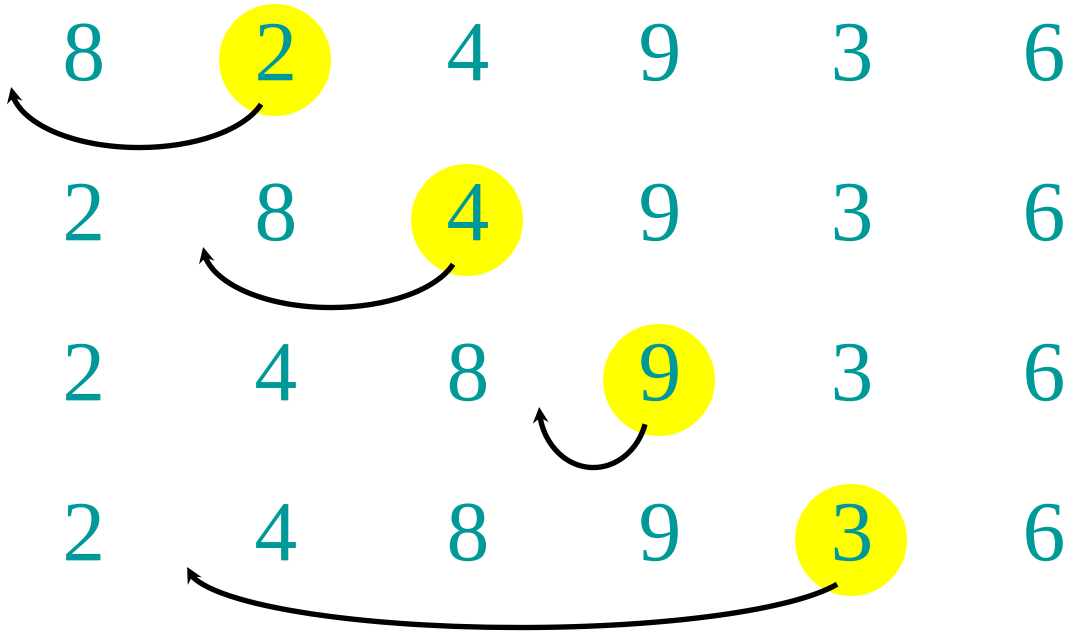
Insertion sort: example



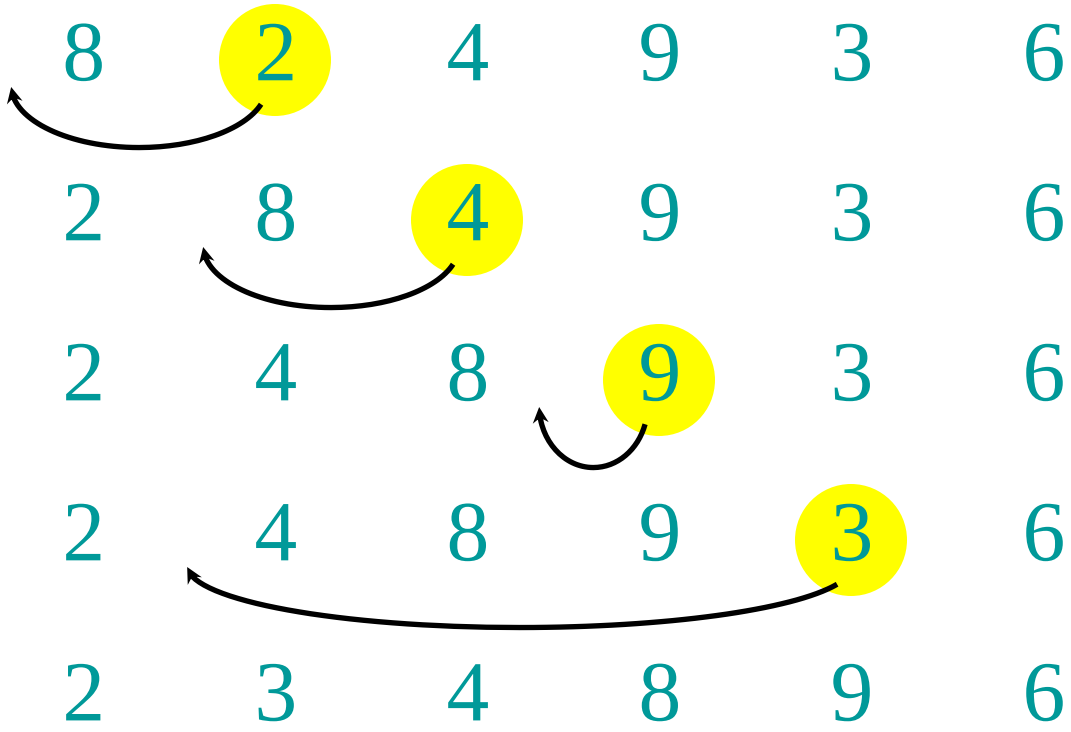
Insertion sort: example



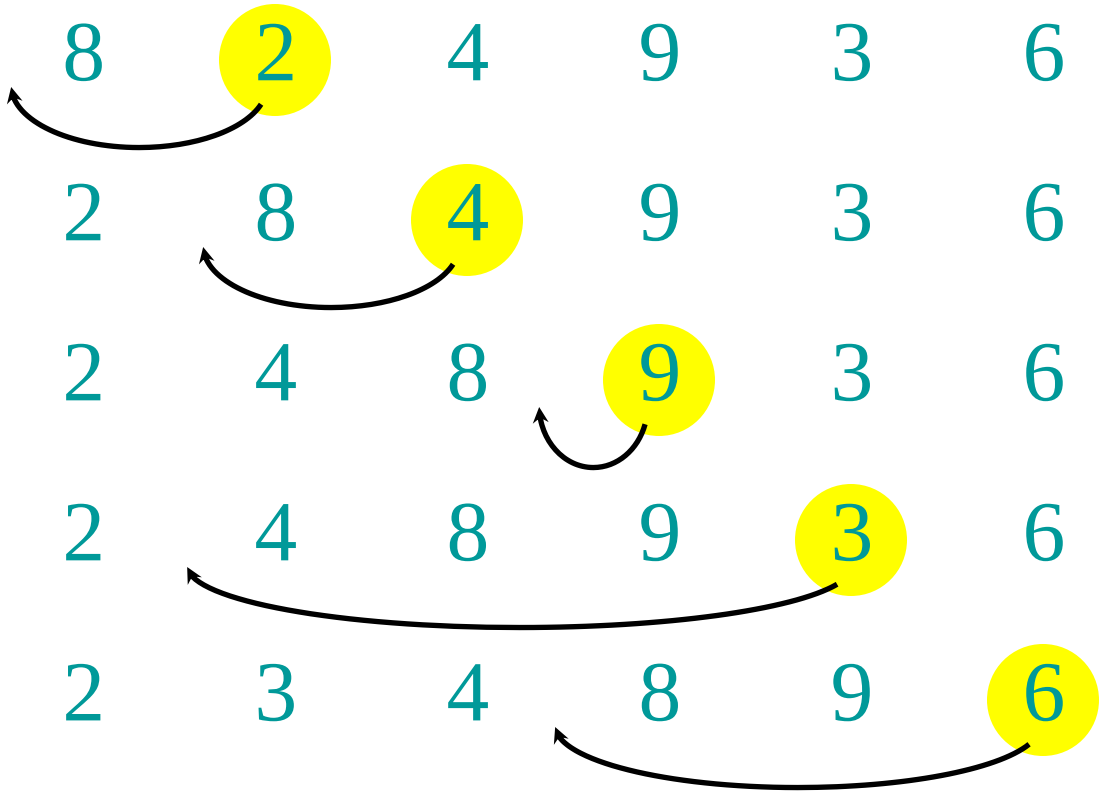
Insertion sort: example



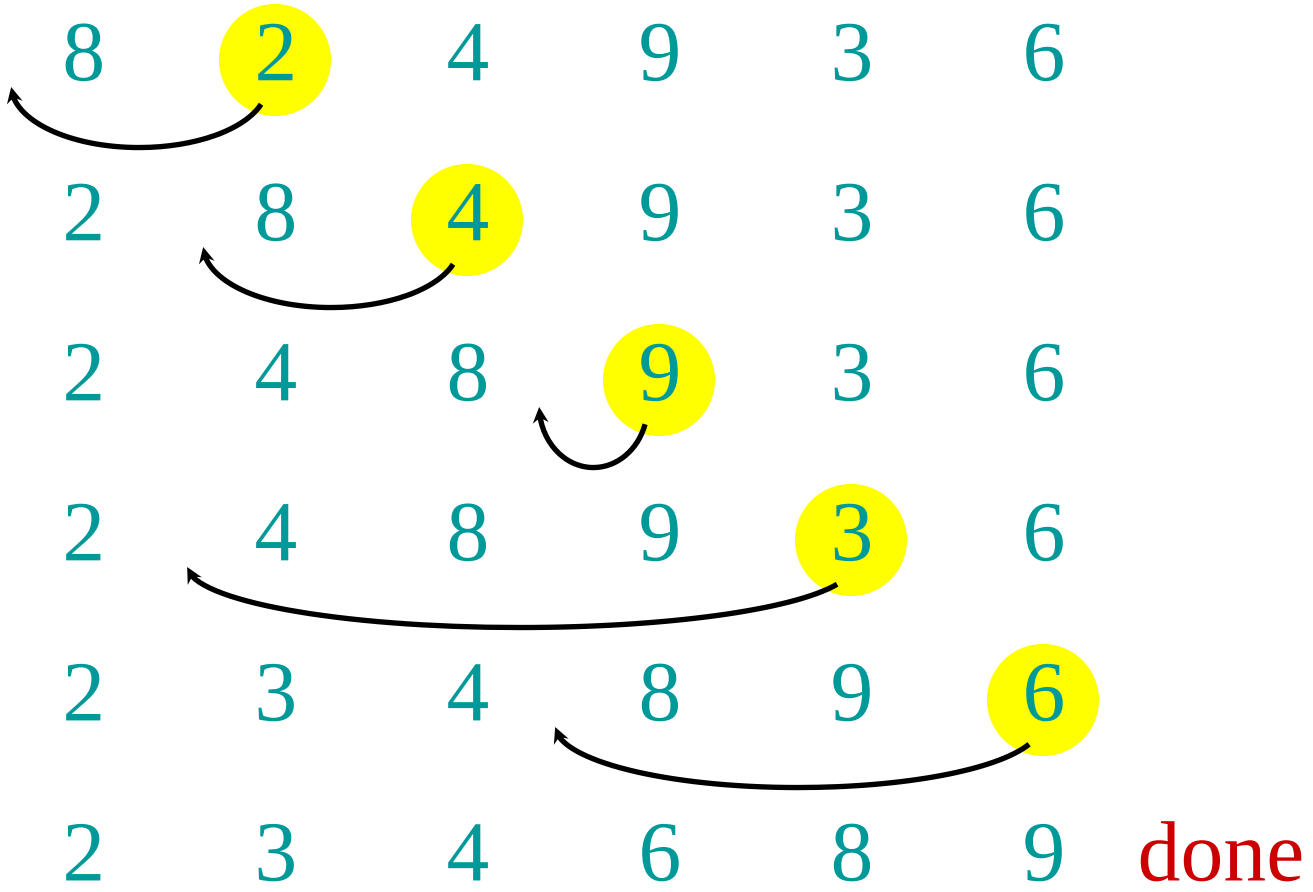
Insertion sort: example



Insertion sort: example



Insertion sort: example



Insertion Sort: Running Time

We use:

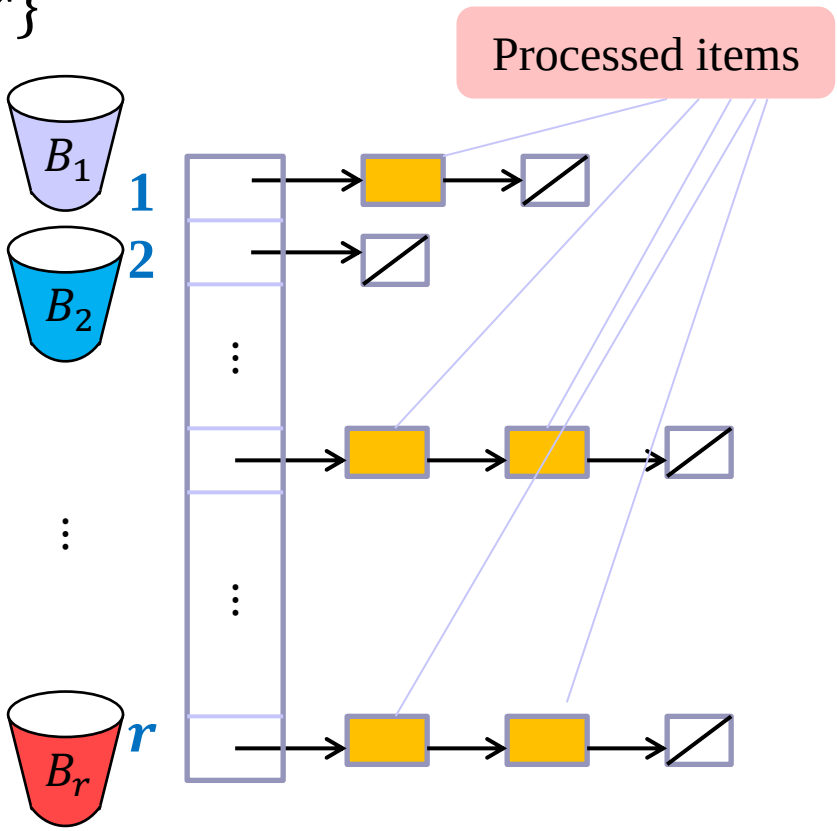
- 1 step to insert the second element
- 2 steps to insert the third element
- …
- $n - 1$ steps to insert the n^{th} element

Total: steps

Bucket Sort

- We can sort much faster when the range of numbers is small
- Example: Sorting solutions to one HW problem by score (1 to 10)
- Input: n integers from range $\{1, \ldots, r\}$

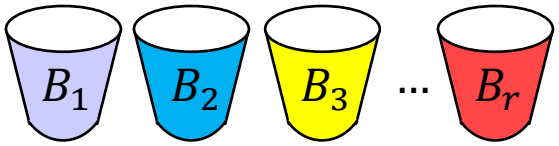
1. Put items with the same value into the same bucket.
2. Keep a linked list for each bucket.
3. Make a pass over the input and put each element in the correct bucket
4. Concatenate the lists.



- If $r \leq n$, we can sort in $O(n)$ time

Bucket Sort

- Given: n integers from range $\{1, \dots, r\}$.



- What if $r > n$? (Suppose for simplicity that n divides r.)

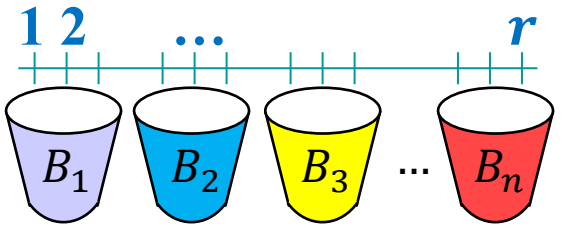
Theorem

If n integers are chosen uniformly and independently from range $\{1, \dots, r\}$, they can be sorted in expected time $O(n)$.

- Expectation is over randomness in the choice of integers: Bucket Sort is deterministic (that is, it does not use randomness).

Bucket Sort

- Idea: Break the range into n buckets.



The expected # of elements in each bucket is 1.

We can easily sort all buckets (using Insertion Sort)

Algorithm. Input: integers $a_1, \dots, a_n$

1. Make linked lists for buckets $B_1, \dots, B_n$.
2. For each i from 1 to n
   let $j = \left\lceil \frac{a_i \cdot n}{r} \right\rceil$ and add a_i to B_j.
3. Sort all buckets using Insertion Sort.
4. Output the concatenation of $B_1, \dots, B_n$

- Steps 1,2, and 4 can be implemented to run in $O(n)$ time.

Lemma

Step 3 runs in expected time $O(n)$.

Bucket Sort: Analysis

Lemma

Step 3 (sorting the buckets) runs in expected time $O(n)$.

Proof: Buckets are bins, elements are balls in Balls-in-the-Bins.

- Let RV X_j = # of elements that land in bucket B_j, for $j = 1, \dots, n$
- Time to sort B_j is: $\leq c \cdot X_j^2$ for some constant c
- Expected run time of Step 3:

$$\leq \mathbb{E}\left(\sum_{j\in\{1,\dots,n\}} cX_j^2\right) = c \cdot \sum_{j\in\{1,\dots,n\}} \mathbb{E}\left(X_j^2\right) = cn \cdot \mathbb{E}(X_1^2)$$

by linearity of expectation

by symmetry

Bucket Sort: Analysis

Lemma

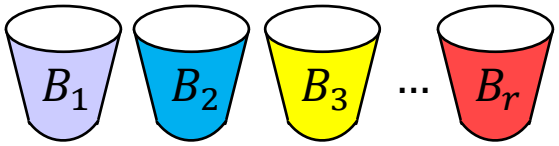
Step 3 (sorting the buckets) runs in expected time $O(n)$.

Proof: Buckets are bins, elements are balls in Balls-in-the-Bins.

- Let RV X_j = # of elements that land in bucket B_j, for $j = 1, \dots, n$
- Expected run time of Step 3: $\leq cn \cdot \mathbb{E}(X_1^2)$
- $X_1 \sim$

Bucket Sort: Conclusion

- Given: n integers from range $\{1, \dots, r\}$.



Theorem

If n integers are chosen uniformly and independently from range $\{1, \dots, r\}$, they can be sorted in expected time $O(n)$.

CS 237

Probability in Algorithms

Probability is used in the design and analysis of algorithms

1. to analyze deterministic algorithms when inputs come from a specified distribution
2. to design and analyze algorithms that use randomness