Why do we need Random numbers?

A Big Deal

- Some reasons why computers have changed all of science, engineering, sociology, politics, economics, …
  - They can *process* tons of data quickly
  - They can also *generate* tons of data quickly
    - Example: Roll a pair of dice 10 million times
- Data generation often requires simulating a process with randomness
  - Because some things (e.g., dice rolls) are random
  - Because some things (e.g., disease causes) may not be random, but it's the best guess we have
    - X% probability of cancer if you smoke

Known vs. unknown solutions

- Sometimes mathematicians have discovered a formula that gives an exact answer to a probability problem
  - Example: Probability two dice sum to 7

- But for more complicated problems sometimes no human knows!
  - “Next best thing”: Try it a lot of times and measure the result
    - Use a computer because it’s faster
  - Can be easier and more convincing than the math even when a formula is known

Monte Carlo

- **Monte Carlo** methods are an important strategy for solving some hard problems in computer science

- They rely on the repeated generation of random data to compute their results

- Often used to simulate mathematical or physical systems. Used in many different fields of science
  - See Wikipedia for a long list of applications

Java Basics:
The Random Class

The `Random` class

- A `Random` object generates pseudo-random numbers

Method name	**Description**
`nextInt()`	returns a random integer
`nextInt(`**max**`)`	returns a random integer in the range [0, *max*)
`nextDouble()`	returns a random real number in the range [0.0, 1.0)

- Class `Random` is found in the `java.util` package

```
    import java.util.Random;
```

```
Random rand = new Random(); // create an object
int randomNum = rand.nextInt(); // call methods on it
```

Common Applications

- Generate a random number from 1 to *N*

```
int n = rand.nextInt(25) + 1;  // 1-25 inclusive
```

- Generate a number in a given range [lo, hi] inclusive:

```
int n = rand.nextInt(hi-lo+1) + lo
```

- Example: A random integer between 41 and 45 inclusive:

```
int n = rand.nextInt(5) + 41;
```

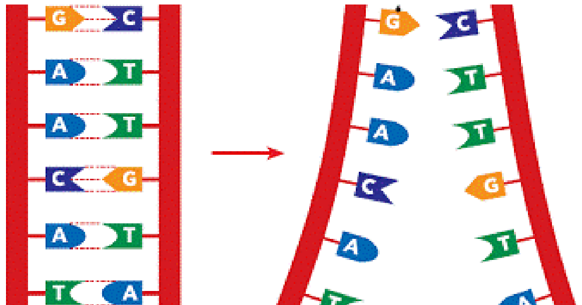
Random text and others

- `Random` can be used in text processing
  - Code to pick a random lowercase letter:

    ```
    char letter = (char)('a' + rand.nextInt(26));
    ```

  - Code to pick a random letter representing a base in a DNA strand (A, C, G, or T):

    ```
    String bases = "ACGT";
    char base = bases.charAt(rand.nextInt(bases.length()));
    ```



`Random` and other types

- `nextDouble` method returns a `double` between 0.0 - 1.0
  - Example: Get a random value between 2.0 and 4.25:

```
double r = rand.nextDouble() * 2.25 + 2.0;
```

- Any finite set of possible values can be mapped to integers
  - E.g., flipping a coin or playing Rock-Paper-Scissors:

```
int r = rand.nextInt(2);
if (r == 0) {
    out.println("Heads");
} else {
    out.println("Tails");
}
```

Random question

- Write code that simulates rolling of two 6-sided dice until a double is rolled

```
2, 3
5, 1
6, 5
1, 2
4, 6
3, 3
It took you 6 tries.
```

Random answer

```
// Rolls two dice until a double is rolled
public static void process() {
    Random rand = new Random(); // create Random object once
    int tries = 0;

    do {
        int die1 = rand.nextInt(6) + 1;
        int die2 = rand.nextInt(6) + 1;
        out.println(die1 + ", " + die2);
        tries++;
    } while (die1 != die2);

    out.println("It took you " + tries + " tries.");
}
```