



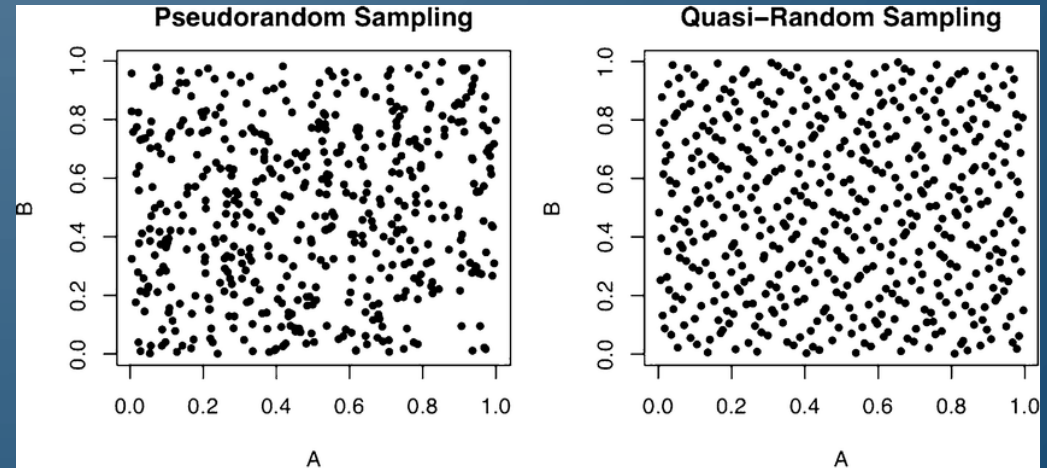
# PROCEDURAL GENERATION

The background is a solid dark blue color. In the four corners, there are decorative white line-art patterns that resemble circuit traces or a stylized tree structure. These patterns consist of thin lines that branch out and terminate in small circles, creating a sense of connectivity and technology.

# BASICS OF RANDOMNESS

# GENERATING RANDOM BITS

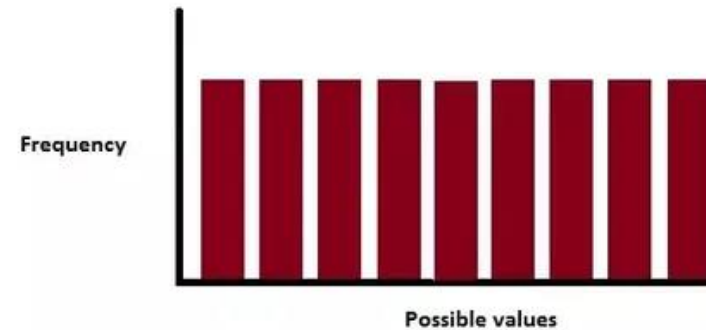
- Random bits are generated from **pseudorandom number generators**, or algorithms that generate a sequence of bits that appears random
  - Sequence based on a "seed" that starts the algorithm off
  - Set of bits are composed and manipulated to interpret higher level types, e.g., integers or floating-point numbers
- Numbers can also be generated through quasi-random or truly random generation (quantum computers)



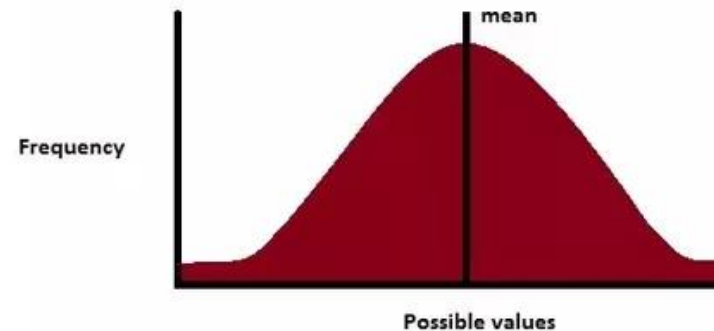
# VARIOUS DISTRIBUTIONS

- Random numbers are manipulated so that they then reflect a probability distribution
- Uniform distribution – all values equally likely in a range (or discrete set)
- Gaussian distribution – values near a mean are more likely than far away from a mean

UNIFORM DISTRIBUTION

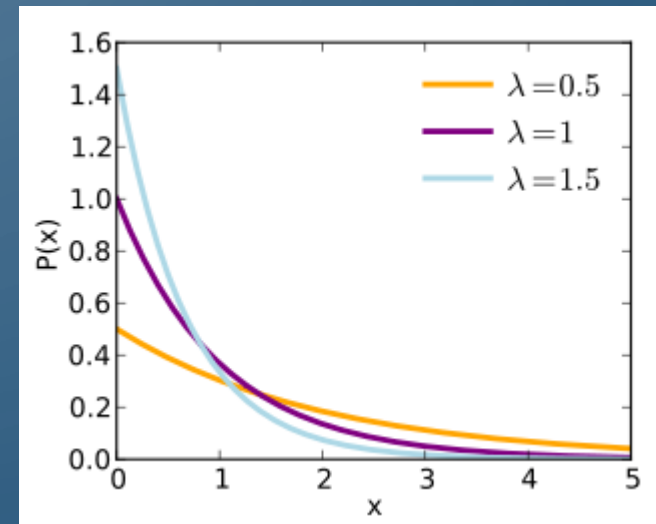


NORMAL DISTRIBUTION



# VARIOUS DISTRIBUTIONS

- Binomial distribution – series of coin flips
- Poisson distribution – expresses probability of a given number of events occurring within a time interval
- Exponential distribution – represents time between different Poisson events
- Etc.



# EXERCISE

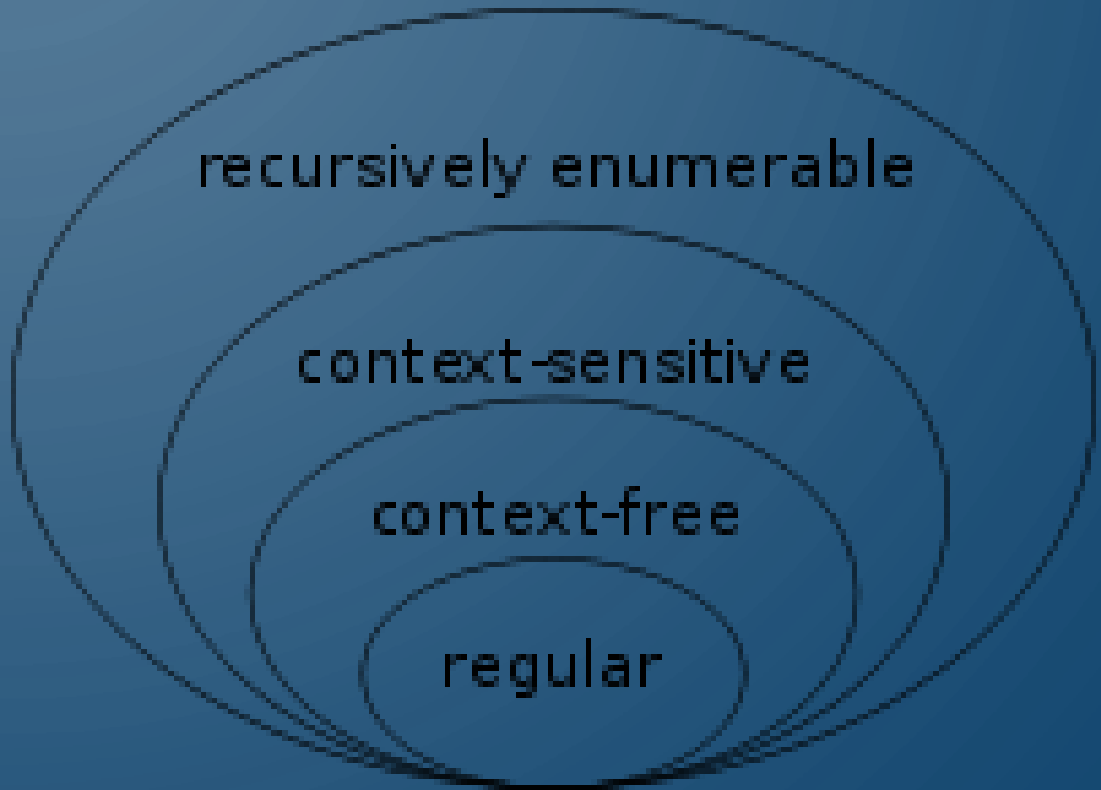
- With a partner
  - Define an algorithm to generate points uniformly within a disc of radius  $r$
  - Define an algorithm to generate velocities biased towards a target velocity for a series of objects
  - Where in games you have played have you seen this most basic form of generation?

The image features a dark blue background with white, stylized circuit board traces in the corners. These traces consist of straight lines that branch out and terminate in small circles, resembling electronic components or data paths. The traces are located in the top-left, top-right, bottom-left, and bottom-right corners, framing the central text.

# GENERATIVE GRAMMARS

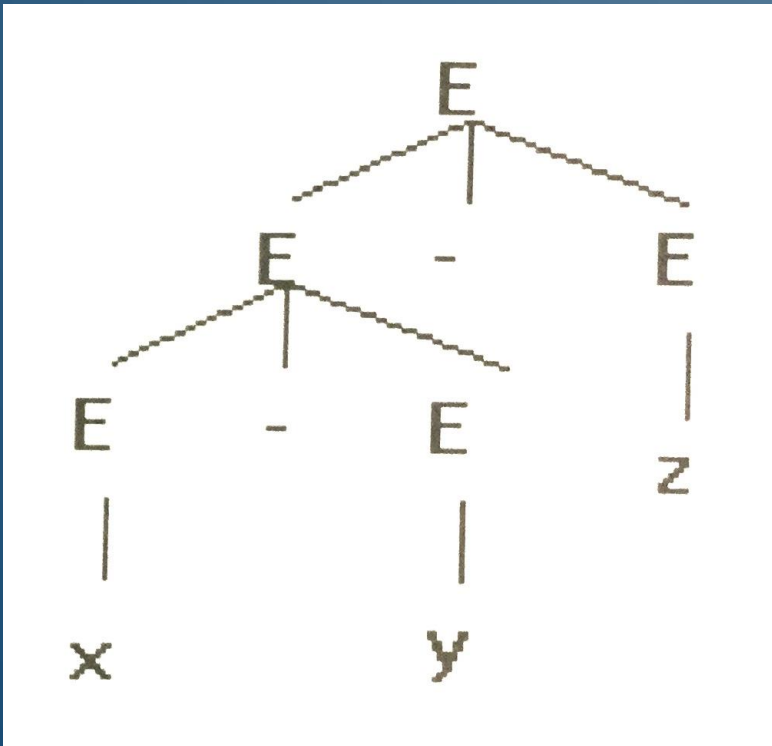
# WHAT'S A GRAMMAR?

- A set of production rules for strings in a language
- Example: all strings containing at least one "a" followed by the same number of "b"s
  - "ab", "aabb", etc
  - Production rules:  
 $S \leftarrow aSb$   
 $S \leftarrow ab$





# HOW CAN THESE BE USEFUL FOR GENERATING CONTENT?



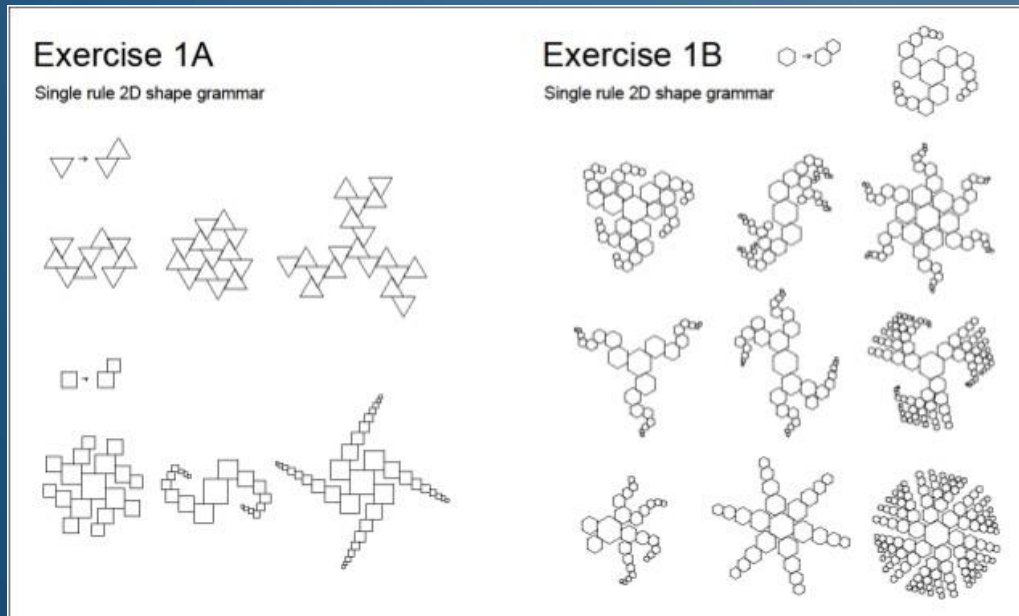
- Expressed as a **generative grammar**, they define rules to generate strings in a language, vs "accept" strings in the language
- Simple procedure could be to uniformly pick a production rule repeatedly to generate a random string from the language
  - Essentially forms a random grammar tree
  - Can apply constraints and non-random decision making as well

# LINDENMAYER SYSTEM

- Combines generative grammar of strings with a translation system to translate the string into geometric structures



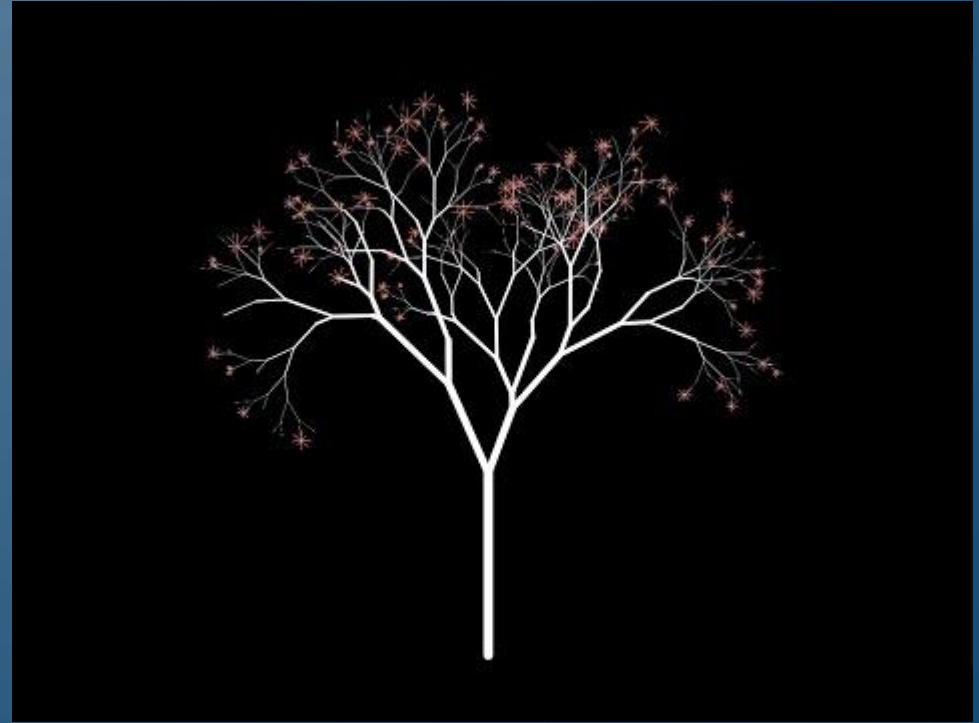
# SHAPE GRAMMAR



- Production system to generate a geometric shape
- Rules describe how an existing part of a shape

# ADVANCED METHODS FOR PROCEDURAL GENERATION

- Image filtering
- Spatial algorithms
  - Fractals
- Simulation and modeling
  - Cellular automata
- AI
  - Genetic algorithms
  - Neural networks



The background is a solid dark blue color. In the four corners, there are decorative white line-art elements that resemble circuit traces or a network diagram. These lines connect to small white circles, creating a sense of connectivity and technology.

HOW DOES PROCEDURAL GENERATION COME  
INTO PLAY IN YOUR GAME CONCEPTS?

# SUMMARY

- Procedural generation can greatly enhance the user experience and replayability of a game
- Many methods can be employed at all levels of game content

# EXAM

- Closed note/book exam
- No questions on Unity
- Format – five sections and a bonus
  - Q1 – T/F – Definitions/Concepts from FGD
  - Q2 – Fill-in-the-blank – Definitions/Concepts from GPAT
  - Q3 – Free response – Game programming (Game loop, input, sound)
  - Q4 – Free response – Game programming (Graphics, camera models, and procedural generation)
  - Q5 – Free response – Analyze design decisions
  - Bonus – ?