Simulating Monopoly

Understanding the game

FRAZER-NASH CONSULTANCY

W. H. Bell

http://www.whbell.net/

http://www.fnc.co.uk

Raspberry Pi Day
University of Strathclyde
22/04/2017

Background

- Encourage pupil to learn some statistics.
  - Picked board game that he was playing.
  - Started from basic mathematics.
- Built discrete event simulation.
  - Constructed in steps to introduce concepts and different effects.

Is there a way to win?

Rolling two dice

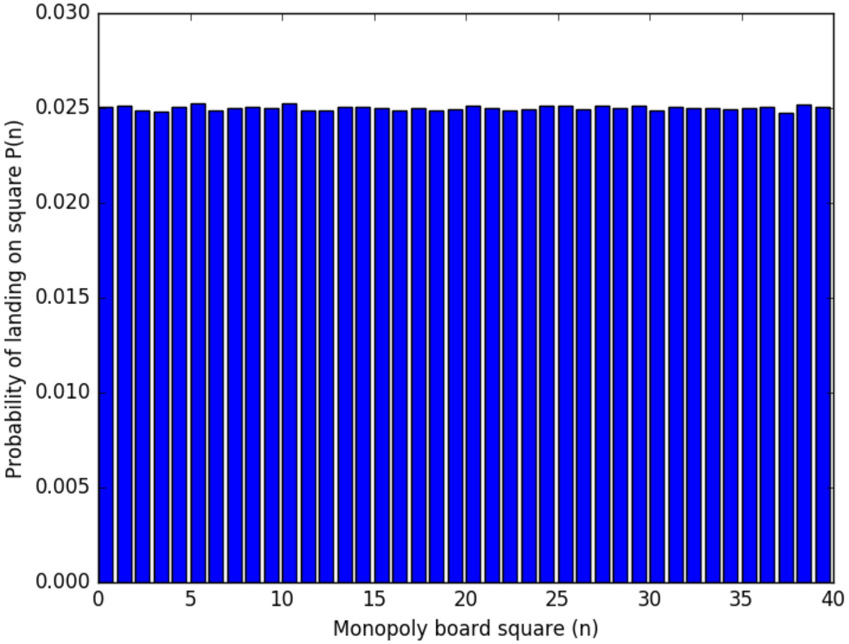
Rolling two dice

```
import random

def rollTwoDice():
  return random.randint(1,6) + random.randint(1,6)

counters=[0.]*12
nRolls = 100
for i in range(nRolls):
  totalValue = rollTwoDice()  # roll the dice

  counters[totalValue-1] = counters[totalValue-1] + 1. # count this total value

for i in range(len(counters)):
  counters[i] = counters[i] / float(nRolls)

print("The probabilities of rolling a total value using two dice:")
for i in range(len(counters)):
  print(" P("+str(i+1)+")="+str(counters[i]))
print("where P(n) is the probability of rolling a total of n on two dice.")
```

```
P(1)=0.0
P(2)=0.02
P(3)=0.06
P(4)=0.05
P(5)=0.08
P(6)=0.15
P(7)=0.13
P(8)=0.21
P(9)=0.11
P(10)=0.09
P(11)=0.07
P(12)=0.03
```

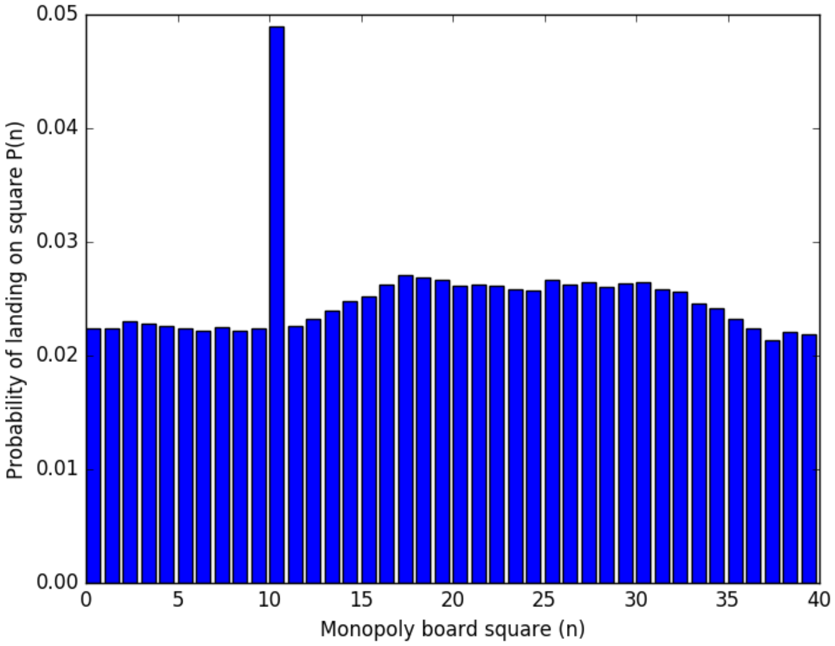
A blank board

```
nsquares = 40
counters=[0.]*nsquares
currentPosition = 0
nRolls = 1000000

for i in range(nRolls):
  totalValue = rollTwoDice()
  currentPosition = currentPosition + totalValue

  if currentPosition >= nsquares:
    currentPosition = currentPosition - nsquares

  counters[currentPosition] = counters[currentPosition] + 1.

for i in range(len(counters)):
  counters[i] = counters[i] / float(nRolls)
```

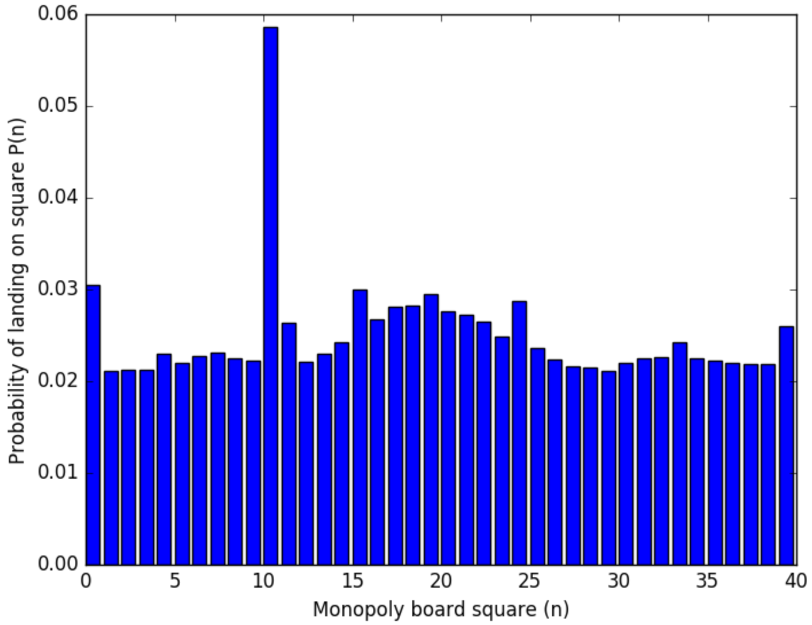


GO TO JAIL

```
nsquares = 40
counters=[0.]*nsquares
currentPosition = 0
nRolls = 1000000

for i in range(nRolls):
  totalValue = rollTwoDice()
  currentPosition = currentPosition + totalValue

  if currentPosition >= nsquares:
    currentPosition = currentPosition – nsquares

  counters[currentPosition] = counters[currentPosition] + 1.

  if currentPosition == 30:
    currentPosition = 10
    counters[currentPosition] = counters[currentPosition] + 1.
```



Adding the card decks



```
# Build the card decks
def createCardDecks(self):

  chanceResults = []

  chanceResults += [ (0, 0) ] # Advance to GO
  chanceResults += [ (0, 11) ] # Advance to Pall Mall
  chanceResults += [ (0, 10) ] # Go to Jail
  chanceResults += [ (0, 15) ] # Take a trip to Marylebone Station
  chanceResults += [ (0, 39) ] # Advance to Mayfair
  chanceResults += [ (0, 24) ] # Advance to Trafalgar square

  chanceResults += [ (1, -3) ] # Go back three spaces
```

Summary

- Combining computer programming with statistics produces a powerful tool.
  - Great for encouraging pupils/students.
  - Applicable to many real world problems.
- Easy to build simulations using Python.

https://github.com/williamhbell/MonopolySimulation