

Programming: Practical 5 Solutions

```
import random
import numpy as np

def RollTwoDiceWithDoubles(current):
    ls = [1, 2, 3, 4, 5, 6]
    dice1 = random.choices(ls, k=3)
    dice2 = random.choices(ls, k=3)

    total = [dice1[i] + dice2[i] for i in range(0, 3)]
    IsDouble = [dice1[i] == dice2[i] for i in range(0, 3)]

    if (IsDouble[0]) & (IsDouble[1]) & (IsDouble[2]):
        current = 10 # Go to Jail
    elif (IsDouble[0]) & (IsDouble[1]):
        current = current + np.sum(total)
    elif IsDouble[0]:
        current = current + total[0] + total[1]
    else:
        current = current + total[0]

    return current
```

Helper function to avoid code replication

```
def CheckState(current):
    if current > 39:
        current = current - 40
    elif current < 0:
        current = current + 40
    return current
```

```
def UpdateStateVector(current, move, landings):
    if move != current:
        landings[current] += 1
```

```
from random import uniform
```

```
def CommunityChest(current):
    u = uniform(0, 1)
    goto = current
    if u < 1/16:
        goto = 0 # Move to Go
    elif u < 2/16:
        goto = 10 # Go to Jail :(
    elif u < 3/16:
        goto = 1 # Old Kent Road
    elif u < 4/16:
        goto = Chance(current)
    return goto
```

```
def Chance(current):
    u = uniform(0, 1)
```

```

goto = current # Default, do nothing
if u < 1/16:
    goto = 0 # Go
elif u < 2/16:
    goto = 24 # Trafalgar Square
elif u < 3/16:
    goto = 11 # Pall Mall
elif u < 4/16:
    goto = 10 # Jail
elif u < 5/16:
    goto = 15 # Mary Stat
elif u < 6/16:
    goto = 39 # Mayfair
elif u < 7/16:
    goto = CheckState(current - 3) # Must check, since goto may be -ve!
elif u < 8/16:
    if (current > 28) | (current < 12):
        goto = 12
    else:
        goto = 28
return goto

```

```

def SimulateMonopoly(no_of_rolls):
    landings = [0]*40
    # Start at GO
    current = 0
    for i in range(0, no_of_rolls):
        current = RollTwoDiceWithDoubles(current)
        current = CheckState(current)
        UpdateStateVector(current, -1, landings)

        if (current == 7) | (current == 22) | (current == 36): # Chance
            move = Chance(current)
            UpdateStateVector(move, current, landings)
            current = move

        if (current == 2) | (current == 17) | (current == 33): # Com Chest
            move = CommunityChest(current)
            UpdateStateVector(move, current, landings)
            current = move

        # Go To Jail. Chance could also send you here by
        # going back three places

        if current == 30:
            current = 10
            UpdateStateVector(current, -1, landings)

    return landings

```

```
## (0, 0.06)
```

```
import matplotlib.pyplot as plt
```

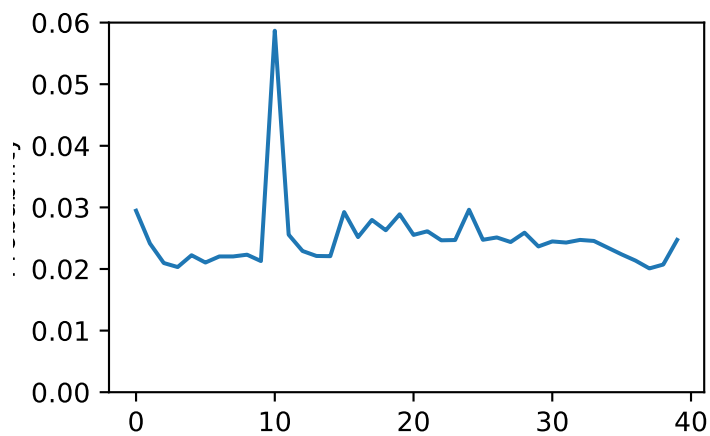


Figure 1: Distribution of resting places in a game of monopoly using 1,000,000 rolls.

```
no_of_rolls = 1000000
sim = SimulateMonopoly(no_of_rolls)
sim = sim/np.sum(sim)
plt.plot(sim)
plt.ylim(0, 0.06)
plt.show()
```