

Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE0405 - Modelos Probabilísticos de Señales y Sistemas
Laboratorio de programación 1: Conceptos de probabilidad
Realizado por: Jonathan Rojas Sibaja y Fabián Abarca Calderón

Profesor: Fabián Abarca Calderón

Grupo 01

I - 2020

Estudiante: Belinda Brown Ramírez

1 Definición clásica de la probabilidad

1.1 Ejemplo 1.1

Ejemplo resuelto: el espacio de resultados elementales al lanzar una moneda es:

$$E = \{1, 2, 3, 4, 5, 6\}$$

¿Cuál es la probabilidad de cada suceso? [1]

```
[ ]: espacio_muestras = [1,2,3,4,5,6]
n = len(espacio_muestras)
probabilidad = 1.0/n
print(probabilidad)
```

```
(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$ python3 DefinicionClasicaProbabilidad.py
Se posee la siguiente muestra: 1,2,3,4,5,6
La probabilidad es de: 0.16666666666666666
(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$
```

¿Cuál es la probabilidad de que salga un número par?

```
[ ]: numeros_pares = [i for i in espacio_muestras if i % 2 is 0]
h = len(numeros_pares)
probabilidadpares = float(h)/n
print(probabilidadpares)
```

```
(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$ python3 DefinicionClasicaProbabilidad.py
Se posee la siguiente muestra: 1,2,3,4,5,6
Probabilidad de que salga un número par: 0.5
(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$
```

1.2 Ejemplo 1.2

- ¿Cuál es la probabilidad de que una carta sacada de un mazo (52 cartas en cuatro palos) sea divisible por 6?

```
[ ]: # Crear espacio de muestras
palo = [i+1 for i in range(13)]

# Comprobar divisibilidad por 6
divisible_6 = [i for i in palo if i % 6 is 0]

'''
Tomando en cuenta que hay cuatro palos,
la frecuencia relativa de los números
divisibles por 6 es:
'''

probabilidad_6 = (4*float(len(divisible_6)))/(4*len(palo))
print("La probabilidad es {0:0.2f}".format(probabilidad_6))
```

```
(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$ python3 DefinicionClasicaProbabilidad.py
Tomando en cuenta que hay cuatro palos, la frecuencia relativa de los números divisibles por 6 es:
0.15
(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$
```

2 Eventos independientes

2.1 Ejemplo 2.1

Ejemplo resuelto: el siguiente es un ejemplo para la solución “elegante” de un problema común.

¿Cuál es la probabilidad de que, al lanzar dos dados, la suma de los dados sea 7? [2]

El resultado es fácil de deducir: de 36 combinaciones posibles, seis suman siete (1 + 6, 2 + 5, 3 + 4, 4 + 3, 5 + 2, 6 + 1), entonces $6/36 = 1/6 \approx 0.16667$.

Solución alternativa

Primero, el objeto defaultdict del [módulo collections](#) crea diccionarios con valores predeterminados cuando encuentra una nueva clave. Su uso práctico es el de ser un “diccionario rellenable”.

```
[ ]: from collections import defaultdict
```

Ahora, es posible crear un diccionario con todas las combinaciones posibles y la suma de cada una, con un doble bucle for (observen qué poderosa es la sintaxis de Python aquí):

```
[ ]: d = {(i,j) : i+j for i in range(1, 7) for j in range(1,7)}
print(d)
```

Seguidamente se crea un defaultdict vacío. Este implica que, más adelante, si una clave no es encontrada en el diccionario, en lugar de un KeyError se crea una nueva entrada (un nuevo par

key:value). Si no saben qué es `KeyError` o `key:value`, pueden ir a leer más sobre diccionarios en Python.

```
[ ]: dinv = defaultdict(list)
      print(dinv)
```

Es posible extraer del diccionario las combinaciones que suman 7. El método `.items()` genera una lista de pares de “tuplas” (una tupla es un conjunto ordenado e inmutable de elementos) a partir del diccionario de combinaciones creado en `d`. “Rellenamos” el `defaultdict` con los elementos en el diccionario creado anteriormente y el método `.append()`, esto con un bucle `for` en donde los índices `i, j` representan los pares de combinaciones y su suma. La ventaja es que ahora están todos agrupados.

```
[ ]: print('Antes...\n')
      print(d.items())

      for i,j in d.items():
          dinv[j].append(i)

      print('\nDespués...')
      dinv
```

El `for` anterior puede leerse como: “para cada par en la lista de ítems, en la posición `j` (la suma de las combinaciones) añada la combinación correspondiente (en `i`)”.

Extraemos los pares que suman siete y obtenemos la cantidad de estos.

```
[ ]: print('Combinaciones que suman 7:', dinv[7])
      print('Elementos:', len(dinv[7]))
```

Finalmente, y más en general, se obtiene la probabilidad para todas las sumas en forma de un solo diccionario:

```
[ ]: probabilidades = {i : len(j)/36 for i,j in dinv.items()}
      print('El vector de probabilidades de suma es =', probabilidades)
      print('La probabilidad de que la suma sea 7 es =', probabilidades[7])
```

2.1.1 Resultado — ¿Cuál es la probabilidad de que, al lanzar dos dados, la suma de los dados sea 7?

```
(base) Belindas-MacBook-Air:Python belindabrown$ python3 ProbabilidadLanzar2DadosSuma7.py

Imprimiendo combinaciones posibles de los dados y suma de esta
Lleva el siguiente formato (numero dado 1, numero dado 2): suma dados

{(1, 1): 2, (1, 2): 3, (1, 3): 4, (1, 4): 5, (1, 5): 6, (1, 6): 7, (2, 1): 3, (2, 2): 4, (2, 3): 5, (2, 4): 6, (2, 5): 7, (2, 6): 8, (3, 1): 4, (3, 2): 5, (3, 3): 6, (3, 4): 7, (3, 5): 8, (3, 6): 9, (4, 1): 5, (4, 2): 6, (4, 3): 7, (4, 4): 8, (4, 5): 9, (4, 6): 10, (5, 1): 6, (5, 2): 7, (5, 3): 8, (5, 4): 9, (5, 5): 10, (5, 6): 11, (6, 1): 7, (6, 2): 8, (6, 3): 9, (6, 4): 10, (6, 5): 11, (6, 6): 12}

Conjunto completo:      defaultdict(<class 'list'>, {})

Antes de buscar las posibles permutaciones en las que puede suceder el resultado de 7...

dict_items([(1, 1), 2], [(1, 2), 3], [(1, 3), 4], [(1, 4), 5], [(1, 5), 6], [(1, 6), 7], [(2, 1), 3], [(2, 2), 4], [(2, 3), 5], [(2, 4), 6], [(2, 5), 7], [(2, 6), 8], [(3, 1), 4], [(3, 2), 5], [(3, 3), 6], [(3, 4), 7], [(3, 5), 8], [(3, 6), 9], [(4, 1), 5], [(4, 2), 6], [(4, 3), 7], [(4, 4), 8], [(4, 5), 9], [(4, 6), 10], [(5, 1), 6], [(5, 2), 7], [(5, 3), 8], [(5, 4), 9], [(5, 5), 10], [(5, 6), 11], [(6, 1), 7], [(6, 2), 8], [(6, 3), 9], [(6, 4), 10], [(6, 5), 11], [(6, 6), 12]])

Después de realizar las distintas permutaciones obtenemos:

defaultdict(<class 'list'>, {2: [(1, 1)], 3: [(1, 2), (2, 1)], 4: [(1, 3), (2, 2), (3, 1)], 5: [(1, 4), (2, 3), (3, 2), (4, 1)], 6: [(1, 5), (2, 4), (3, 3), (4, 2), (5, 1)], 7: [(1, 6), (2, 5), (3, 4), (4, 3), (5, 2), (6, 1)], 8: [(2, 6), (3, 5), (4, 4), (5, 3), (6, 2)], 9: [(3, 6), (4, 5), (5, 4), (6, 3)], 10: [(4, 6), (5, 5), (6, 4)], 11: [(5, 6), (6, 5)], 12: [(6, 6)]})

Combinaciones que suman 7:  [(1, 6), (2, 5), (3, 4), (4, 3), (5, 2), (6, 1)]

Elementos (cantidad de combinaciones):  6

El vector de probabilidades (probabilidad de cada número en salir cuando se lanzan los dados) de suma es:
{2: 0.027777777777777776, 3: 0.05555555555555555, 4: 0.08333333333333333, 5: 0.11111111111111111, 6: 0.13888888888888889, 7: 0.16666666666666666, 8: 0.13888888888888889, 9: 0.11111111111111111, 10: 0.08333333333333333, 11: 0.05555555555555555, 12: 0.027777777777777776}

La probabilidad de que la suma sea 7 es = 0.16666666666666666
(base) Belindas-MacBook-Air:Python belindabrown$
```

2.2 Ejemplo 2.2

Con el método anterior, puede resolverse ahora un problema más extenso:

- Las placas del país tienen tres letras y tres dígitos numéricos. En cierto juego, unos amigos apuestan un fresco de tamarindo para el primero que encuentre una placa cuyos dígitos numéricos suman: 10 para A(ndrea), 15 para B(renda) y 20 para C(arlos). ¿Quién tiene más probabilidades de ganar?

```
[ ]: #Resolución de problema placas y suma
#Ejemplo 2.2
###Las placas del país tienen tres letras y
##tres dígitos numéricos. En cierto juego, unos
##amigos apuestan un fresco de tamarindo para el primero
##que encuentre una placa cuyos dígitos numéricos suman:
## 10 para A(ndrea), 15 para B(renda) y 20 para C(arlos).
##¿Quién tiene más probabilidades de ganar?

##Forma de ejecutarlo
#python3 PlacasConSumaDigitosDeterminado.py

#Importando paquetes necesarios
from collections import defaultdict

#Creando diccionario con todas las combinaciones posibles y la suma de cada una
```

```

#dado que son 0-9 los numeros y tres posibles
d = {(i,j) : i+j for i in range(0, 9) for j in range(0,9) for k in range(0,9)}

print("\n")
print("Imprimiendo combinaciones posibles de las placas y suma de esta")
print("Lleva el siguiente formato (Digito uno, Digito 2, Digito 3): suma_
    ↳digites placa\n")
print(d)

#creando un diccionario vacio para que cuando no se crea en el diccionario_
    ↳original
#en vez de error lo que haga sea crearlo, en otras palabras, es subconjunto del_
    ↳diccionario original
#una vez que rellena para ser el conjunto universal
dinv = defaultdict(list)
#se imprime
print("\nConjunto completo:      ", dinv)

print('\nAntes de buscar las posibles permutaciones\n')
print(d.items())

#Realizando las permutaciones
for i,j in d.items():
    dinv[j].append(i)

print('\nDespués de realizar las distintas permutaciones obtenemos:\n')
print(dinv)
#Calculando 10x10x10 = 1000
probabilidades = {i : len(j)/1000 for i,j in dinv.items()}
print('\nEl vector de probabilidades (probabilidad de cada número en salir en la_
    ↳placa: \n', probabilidades)

#se escoge de las permutaciones las que suman 10
print('\nCombinaciones que suman 10 (Andrea): ', dinv[10])
print('\nElementos (cantidad de combinaciones) para la suma de 10: ',_
    ↳len(dinv[10]))
print('\nLa probabilidad de que la suma sea 10 es = ', probabilidades[10])

#se escoge de las permutaciones las que suman 15
print('\nCombinaciones que suman 15 (Brenda): ', dinv[15])
print('\nElementos (cantidad de combinaciones) para la suma de 15: ',_
    ↳len(dinv[15]))
print('\nLa probabilidad de que la suma sea 15 es = ', probabilidades[15])

```

```
#se escoge de las permutaciones las que suman 20
print('\nCombinaciones que suman 20 (Carlos): ', dinv[20])
print('\nElementos (cantidad de combinaciones) para la suma de 20: ',
      len(dinv[20]))
if (len(dinv[20])==0):
    print('\nLa probabilidad de que la suma sea 20 es = 0')
else:
    print('\nLa probabilidad de que la suma sea 20 es = ', probabilidades[20])
```

```
##Últimas cifras de las placas:
# Calcule la probabilidad de que se repitan
#las dos últimas cifras de la matrícula en quince
#automóviles anotados al azar.
```

```
Imprimiendo combinaciones posibles de las placas y suma de esta
Lleva el siguiente formato (Digito uno, Digito 2, Digito 3): suma digitos placa

{0, 0}: 0, {0, 1}: 1, {0, 2}: 2, {0, 3}: 3, {0, 4}: 4, {0, 5}: 5, {0, 6}: 6, {0, 7}: 7, {0, 8}: 8, {0, 9}: 9, {1, 0}: 1, {1, 1}: 2, {1, 2}: 3, {1, 3}: 4, {1, 4}: 5, {1, 5}: 6, {1, 6}: 7, {1, 7}: 8, {1, 8}: 9, {1, 9}: 10, {2, 0}: 2, {2, 1}: 3, {2, 2}: 4, {2, 3}: 5, {2, 4}: 6, {2, 5}: 7, {2, 6}: 8, {2, 7}: 9, {2, 8}: 10, {2, 9}: 11, {3, 0}: 3, {3, 1}: 4, {3, 2}: 5, {3, 3}: 6, {3, 4}: 7, {3, 5}: 8, {3, 6}: 9, {3, 7}: 10, {3, 8}: 11, {3, 9}: 12, {4, 0}: 4, {4, 1}: 5, {4, 2}: 6, {4, 3}: 7, {4, 4}: 8, {4, 5}: 9, {4, 6}: 10, {4, 7}: 11, {4, 8}: 12, {4, 9}: 13, {5, 0}: 5, {5, 1}: 6, {5, 2}: 7, {5, 3}: 8, {5, 4}: 9, {5, 5}: 10, {5, 6}: 11, {5, 7}: 12, {5, 8}: 13, {5, 9}: 14, {6, 0}: 6, {6, 1}: 7, {6, 2}: 8, {6, 3}: 9, {6, 4}: 10, {6, 5}: 11, {6, 6}: 12, {6, 7}: 13, {6, 8}: 14, {6, 9}: 15, {7, 0}: 7, {7, 1}: 8, {7, 2}: 9, {7, 3}: 10, {7, 4}: 11, {7, 5}: 12, {7, 6}: 13, {7, 7}: 14, {7, 8}: 15, {7, 9}: 16, {8, 0}: 8, {8, 1}: 9, {8, 2}: 10, {8, 3}: 11, {8, 4}: 12, {8, 5}: 13, {8, 6}: 14, {8, 7}: 15, {8, 8}: 16, {8, 9}: 17

Conjunto completo: defaultdict(, {})

Antes de buscar las posibles permutaciones

dict.items([(0, 0), (0, 1), (0, 2), (0, 3), (0, 4), (0, 5), (0, 6), (0, 7), (0, 8), (0, 9), (1, 0), (1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (1, 7), (1, 8), (1, 9), (2, 0), (2, 1), (2, 2), (2, 3), (2, 4), (2, 5), (2, 6), (2, 7), (2, 8), (2, 9), (3, 0), (3, 1), (3, 2), (3, 3), (3, 4), (3, 5), (3, 6), (3, 7), (3, 8), (3, 9), (4, 0), (4, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), (4, 7), (4, 8), (4, 9), (5, 0), (5, 1), (5, 2), (5, 3), (5, 4), (5, 5), (5, 6), (5, 7), (5, 8), (5, 9), (6, 0), (6, 1), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6), (6, 7), (6, 8), (6, 9), (7, 0), (7, 1), (7, 2), (7, 3), (7, 4), (7, 5), (7, 6), (7, 7), (7, 8), (7, 9), (8, 0), (8, 1), (8, 2), (8, 3), (8, 4), (8, 5), (8, 6), (8, 7), (8, 8), (8, 9), (9, 0), (9, 1), (9, 2), (9, 3), (9, 4), (9, 5), (9, 6), (9, 7), (9, 8), (9, 9)])

Después de realizar las distintas permutaciones obtenemos:

defaultdict(, {0: [(0, 0)], 1: [(0, 1), (1, 0)], 2: [(0, 2), (1, 1), (2, 0)], 3: [(0, 3), (1, 2), (2, 1), (3, 0)], 4: [(0, 4), (1, 3), (2, 2), (3, 1), (4, 0)], 5: [(0, 5), (1, 4), (2, 3), (3, 2), (4, 1), (5, 0)], 6: [(0, 6), (1, 5), (2, 4), (3, 3), (4, 2), (5, 1), (6, 0)], 7: [(0, 7), (1, 6), (2, 5), (3, 4), (4, 3), (5, 2), (6, 1), (7, 0)], 8: [(0, 8), (1, 7), (2, 6), (3, 5), (4, 4), (5, 3), (6, 2), (7, 1), (8, 0)], 9: [(0, 9), (1, 8), (2, 7), (3, 6), (4, 5), (5, 4), (6, 3), (7, 2), (8, 1), (9, 0)], 10: [(1, 0), (2, 1), (3, 2), (4, 3), (5, 4), (6, 5), (7, 6), (8, 7), (9, 8)], 11: [(1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (6, 6), (7, 7), (8, 8), (9, 9)], 12: [(1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9)], 13: [(1, 3), (2, 4), (3, 5), (4, 6), (5, 7), (6, 8), (7, 9)], 14: [(1, 4), (2, 5), (3, 6), (4, 7), (5, 8), (6, 9)], 15: [(1, 5), (2, 6), (3, 7), (4, 8), (5, 9)], 16: [(1, 6), (2, 7), (3, 8), (4, 9)], 17: [(1, 7), (2, 8), (3, 9)], 18: [(1, 8), (2, 9)], 19: [(1, 9), (2, 0)], 20: [(2, 0), (1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (6, 6), (7, 7), (8, 8), (9, 9)], 21: [(2, 1), (1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9)], 22: [(2, 2), (1, 3), (2, 4), (3, 5), (4, 6), (5, 7), (6, 8), (7, 9)], 23: [(2, 3), (1, 4), (2, 5), (3, 6), (4, 7), (5, 8), (6, 9)], 24: [(2, 4), (1, 5), (2, 6), (3, 7), (4, 8), (5, 9)], 25: [(2, 5), (1, 6), (2, 7), (3, 8), (4, 9)], 26: [(2, 6), (1, 7), (2, 8), (3, 9)], 27: [(2, 7), (1, 8), (2, 9)], 28: [(2, 8), (1, 9), (2, 0)], 29: [(2, 9), (1, 0), (2, 1)], 30: [(3, 0), (2, 1), (3, 2), (4, 3), (5, 4), (6, 5), (7, 6), (8, 7), (9, 8)], 31: [(3, 1), (2, 2), (3, 3), (4, 4), (5, 5), (6, 6), (7, 7), (8, 8), (9, 9)], 32: [(3, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9)], 33: [(3, 3), (2, 4), (3, 5), (4, 6), (5, 7), (6, 8), (7, 9)], 34: [(3, 4), (2, 5), (3, 6), (4, 7), (5, 8), (6, 9)], 35: [(3, 5), (2, 6), (3, 7), (4, 8), (5, 9)], 36: [(3, 6), (2, 7), (3, 8), (4, 9)], 37: [(3, 7), (2, 8), (3, 9)], 38: [(3, 8), (2, 9), (3, 0)], 39: [(3, 9), (2, 0), (3, 1)], 40: [(4, 0), (3, 1), (4, 2), (5, 3), (6, 4), (7, 5), (8, 6), (9, 7)], 41: [(4, 1), (3, 2), (4, 3), (5, 4), (6, 5), (7, 6), (8, 7), (9, 8)], 42: [(4, 2), (3, 3), (4, 4), (5, 5), (6, 6), (7, 7), (8, 8), (9, 9)], 43: [(4, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9)], 44: [(4, 4), (3, 5), (4, 6), (5, 7), (6, 8), (7, 9)], 45: [(4, 5), (3, 6), (4, 7), (5, 8), (6, 9)], 46: [(4, 6), (3, 7), (4, 8), (5, 9)], 47: [(4, 7), (3, 8), (4, 9)], 48: [(4, 8), (3, 9), (4, 0)], 49: [(4, 9), (3, 0), (4, 1)], 50: [(5, 0), (4, 1), (5, 2), (6, 3), (7, 4), (8, 5), (9, 6)], 51: [(5, 1), (4, 2), (5, 3), (6, 4), (7, 5), (8, 6), (9, 7)], 52: [(5, 2), (4, 3), (5, 4), (6, 5), (7, 6), (8, 7), (9, 8)], 53: [(5, 3), (4, 4), (5, 5), (6, 6), (7, 7), (8, 8), (9, 9)], 54: [(5, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9)], 55: [(5, 5), (4, 6), (5, 7), (6, 8), (7, 9)], 56: [(5, 6), (4, 7), (5, 8), (6, 9)], 57: [(5, 7), (4, 8), (5, 9)], 58: [(5, 8), (4, 9), (5, 0)], 59: [(5, 9), (4, 0), (5, 1)], 60: [(6, 0), (5, 1), (6, 2), (7, 3), (8, 4), (9, 5)], 61: [(6, 1), (5, 2), (6, 3), (7, 4), (8, 5), (9, 6)], 62: [(6, 2), (5, 3), (6, 4), (7, 5), (8, 6), (9, 7)], 63: [(6, 3), (5, 4), (6, 5), (7, 6), (8, 7), (9, 8)], 64: [(6, 4), (5, 5), (6, 6), (7, 7), (8, 8), (9, 9)], 65: [(6, 5), (5, 6), (6, 7), (7, 8), (8, 9)], 66: [(6, 6), (5, 7), (6, 8), (7, 9)], 67: [(6, 7), (5, 8), (6, 9)], 68: [(6, 8), (5, 9), (6, 0)], 69: [(6, 9), (5, 0), (6, 1)], 70: [(7, 0), (6, 1), (7, 2), (8, 3), (9, 4)], 71: [(7, 1), (6, 2), (7, 3), (8, 4), (9, 5)], 72: [(7, 2), (6, 3), (7, 4), (8, 5), (9, 6)], 73: [(7, 3), (6, 4), (7, 5), (8, 6), (9, 7)], 74: [(7, 4), (6, 5), (7, 6), (8, 7), (9, 8)], 75: [(7, 5), (6, 6), (7, 7), (8, 8), (9, 9)], 76: [(7, 6), (6, 7), (7, 8), (8, 9)], 77: [(7, 7), (6, 8), (7, 9)], 78: [(7, 8), (6, 9), (7, 0)], 79: [(7, 9), (6, 0), (7, 1)], 80: [(8, 0), (7, 1), (8, 2), (9, 3)], 81: [(8, 1), (7, 2), (8, 3), (9, 4)], 82: [(8, 2), (7, 3), (8, 4), (9, 5)], 83: [(8, 3), (7, 4), (8, 5), (9, 6)], 84: [(8, 4), (7, 5), (8, 6), (9, 7)], 85: [(8, 5), (7, 6), (8, 7), (9, 8)], 86: [(8, 6), (7, 7), (8, 8), (9, 9)], 87: [(8, 7), (7, 8), (8, 9)], 88: [(8, 8), (7, 9), (8, 0)], 89: [(8, 9), (7, 0), (8, 1)], 90: [(9, 0), (8, 1), (9, 2), (0, 3)], 91: [(9, 1), (8, 2), (9, 3), (0, 4)], 92: [(9, 2), (8, 3), (9, 4), (0, 5)], 93: [(9, 3), (8, 4), (9, 5), (0, 6)], 94: [(9, 4), (8, 5), (9, 6), (0, 7)], 95: [(9, 5), (8, 6), (9, 7), (0, 8)], 96: [(9, 6), (8, 7), (9, 8), (0, 9)], 97: [(9, 7), (8, 8), (9, 9), (0, 0)], 98: [(9, 8), (8, 9), (9, 0), (0, 1)], 99: [(9, 9), (8, 0), (9, 1), (0, 2)]})

El vector de probabilidades (probabilidad de cada número en salir en la placa:
{0: 0.001, 1: 0.002, 2: 0.003, 3: 0.004, 4: 0.005, 5: 0.006, 6: 0.007, 7: 0.008, 8: 0.009, 9: 0.01, 10: 0.007, 11: 0.006, 12: 0.005, 13: 0.004, 14: 0.003, 15: 0.002, 16: 0.001})

Combinaciones que suman 10 (Andrea): [(2, 8), (3, 7), (4, 6), (5, 5), (6, 4), (7, 3), (8, 2)]

Elementos (cantidad de combinaciones) para la suma de 10: 7

La probabilidad de que la suma sea 10 es = 0.007

Combinaciones que suman 15 (Brenda): [(7, 8), (8, 7)]

Elementos (cantidad de combinaciones) para la suma de 15: 2

La probabilidad de que la suma sea 15 es = 0.002

Combinaciones que suman 20 (Carlos): []

Elementos (cantidad de combinaciones) para la suma de 20: 0

La probabilidad de que la suma sea 20 es = 0

(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$
```

- **Últimas cifras de las placas:** Calcule la probabilidad de que se repitan las dos últimas cifras de la matrícula en quince automóviles anotados al azar.

```
[ ]: # Ejemplo 2. 2
#Parte de la coincidencia de las placas
#Últimas cifras de las placas: Calcule la
#probabilidad de que se repitan las dos últimas cifras
#de la matrícula en quince automóviles anotados al azar.
```

```
#Utilizando el método de LA REGLA DE LAPLACE Y LA COMBINATORIA
```

```

#Recuerde que como es coincidencia de días nos fijamos en los 0-9  dígitos qu
→tiene la placa = 10
print("\nEste programa funciona para calcular la coincidencia en los dos últimos
→dígitos de una placa\n")
print("Se calcula para la segunda pregunta en el ejemplo 2.2 \n")
#probabilidad 1
probabilidad15 = 1.0
#personas presentes = asistentes en este caso son 50 para
carros = 15
print("Numero de placas analizadas (carros vistos):  ", carros)

#son dos dígitos combinados 10x10
#calculando la probabilidad
for i in range(carros):
    probabilidad15 = probabilidad15 * (100-i)/100
print("\nProbabilidad de que coincida los dos últimos dígitos de la placa es de
→{0:.2f}" .format(1 - probabilidad15))

```

```

(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$ python3 probabilidadcoincidencia2cifrasplaca.py
Este programa funciona para calcular la coincidencia en los dos últimos dígitos de una placa
Se calcula para la segunda pregunta en el ejemplo 2.2
Numero de placas analizadas (carros vistos):  15
Probabilidad de que coincida los dos últimos dígitos de la placa es de 0.67
(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$

```

2.3 Ejemplo 2.3

- **La coincidencia de cumpleaños:** En una fiesta a la que concurren un total de 50 personas, una amiga intrépida afirma que en la fiesta debe haber por lo menos dos personas que cumplen años el mismo día. ¿Deberíamos creerle? [3] (Plantee la solución para N asistentes a la fiesta).

```

[ ]: # Ejemplo 2. 3
#La coincidencia de cumpleaños:
###En una fiesta a la que concurren un total de 50 personas,
# una amiga intrépida afirma que en la fiesta debe haber por
#lo menos dos personas que cumplen años el mismo día. ¿Deberíamos creerle?

#Utilizando el método de LA REGLA DE LAPLACE Y LA COMBINATORIA

#Recuerde que como es coincidencia de días nos fijamos en los 365  días que
→tiene el año

```

```

print("\nEste programa funciona para calcular la coincidencia en el día de_\n
    ↳ cumpleaños en un evento con N personas\n")
print("Inicialmente se calcula para la primera pregunta en el ejemplo 2.3 \n")
#probabilidad 1
probabilidad50 = 1.0
#personas presentes = asistentes en este caso son 50 para
asistentes = 50
print("Numero de asistentes en la fiesta:  ", asistentes)

#calculando la probabilidad
for i in range(asistentes):
    probabilidad50 = probabilidad50 * (365-i)/365
print("\nProbabilidad de que coincida una misma fecha de cumpleaños es {0:.2f}" .
    ↳format(1 - probabilidad50))

npersonasAsistentes = int(input("Digite el número personas asistentes a la_\n
    ↳fiesta:\n"))
probabilidadN = 1.0
print("\nCorroborando internamente el número de asistentes:  ",_\n
    ↳npersonasAsistentes)

for iterador in range(npersonasAsistentes):
    probabilidadN = probabilidadN * (365-iterador)/365
print("\nProbabilidad de que coincida una misma fecha de cumpleaños es {0:.2f}" .
    ↳format(1 - probabilidadN))

```

```

(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$ python3 CoincidenciaCumpleanosNpersonas.py
Este programa funciona para calcular la coincidencia en el día de cumpleaños en un evento con N personas
Inicialmente se calcula para la primera pregunta en el ejemplo 2.3
Numero de asistentes en la fiesta:  50
Probabilidad de que coincida una misma fecha de cumpleaños es 0.97
Digite el número personas asistentes a la fiesta:
5
Corroborando internamente el número de asistentes:  5
Probabilidad de que coincida una misma fecha de cumpleaños es 0.03
(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$ █

```

3 Teorema de Bayes

El teorema de Bayes, también conocido como la “regla de probabilidad condicional inversa”, tiene la forma general

$$P(A | B) = \frac{P(B | A)P(A)}{P(B)}$$

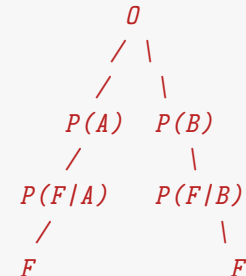
Es útil cuando se cuestiona la premisa de un resultado.

3.1 Ejemplo 3.1

- Un moderno edificio (la Escuela de Ingeniería Eléctrica) tiene dos ascensores para uso de los estudiantes. El primero de los ascensores es usado el 45% de las ocasiones, mientras que el segundo es usado el resto de las ocasiones. El uso continuado de los ascensores provoca un 5% de fallos en el primero de los ascensores y un 8% en el segundo. Un día suena la alarma de uno de los ascensores porque ha fallado. Calcule la probabilidad de que haya sido el primero de los ascensores.

```
[ ]: '''
    Un primer acercamiento intuitivo a la
    solución nos dice que si el primero se
    usa menos veces y falla con menor porcentaje
    entonces la probabilidad de que haya sido el
    de la alarma será menor al 50%.
    '''

    # Datos generales
    p_ascensores = [0.45, 0.55]
    p_fallo = [0.05, 0.08]

    '''
    Con  $[P(A), P(B)] = p\_ascensores$  y
     $[P(F/A), P(F/B)] = p\_fallo$ , entonces
    
    '''

    # Probabilidad total de fallar
    p_fallar = sum([p_ascensores[i] * p_fallo[i] for i in range(len(p_ascensores))])

    '''
    Si  $P(F) = p\_fallar$  es la probabilidad de fallar
    (y de que suene la alarma), entonces el teorema
    de Bayes queda como:

    
$$P(A/F) = \frac{P(F/A) P(A)}{P(F)}$$

    '''
```

```
# Aplicación de Teorema de Bayes
p_bayes = [(p_fallo[i] * p_ascensores[i]) / p_fallar for i in
    range(len(p_ascensores))]
print("La probabilidad de que haya sido el primero de los ascensores es {A:0.2f}.
    → Y el segundo {B:0.2f}".format(A=p_bayes[0]*100, B=p_bayes[1]*100))
```

3.1.1 Resultado — Probabilidad de fallo total, teorema de Bayes

```
(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$ python3 FallosAscensoresTeoremaBayes.py
Un primer acercamiento intuitivo a la solución nos dice que si el primero se usa menos veces y falla con menor porcentaje entonces la probabilidad de que haya sido el de la alarma será menor al 50 porciento

La probabilidad de que haya sido el primero de los ascensores es 33.83. Y el segundo 66.17
(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$
```

3.2 Ejemplo 3.2

- Hay una ciudad de 50 000 habitantes, con la siguiente distribución de población:

Niñas	Niños	Mujeres	Hombres
11000	9000	16000	14000

Hay también un reporte de 9000 casos de una nueva variedad de virus que-no-debe-ser-mencionada, distribuidos de la siguiente forma:

Niñas	Niños	Mujeres	Hombres
2000	1500	3000	2500

¿Está la probabilidad de contraer el virus relacionada con la pertenencia a un sector demográfico? Esto puede estudiarse analizando las probabilidades $P(\text{tener gripe} \mid \text{pertenece a X sector})$ [4]

```
[ ]: #Ejemplo 3.2
#Utilizando el teorema de Bayes
print("\nHay una ciudad de 50 000 habitantes, con la siguiente distribución de
    →población: Niñas 11000 Niños 9000 Mujeres 16000 Hombres 14000. Hay también un
    →reporte de 9000 casos de una nueva variedad de virus
    →que-no-debe-ser-mencionada, distribuidos de la siguiente forma: Niñas 2000
    →Niños 1500 Mujeres 3000 Hombres 2500 ¿Está la probabilidad de contraer el
    →virus relacionada con la pertenencia a un sector demográfico? Esto puede
    →estudiarse analizando las probabilidades (tener gripepertenece a X sector\n")

#Forma de ejecutarlo
#python3 InfeccionPoblacionTeoremaBayes.py
```

```

#se calcula probailidad para cada sector de la poblacion no infectada
MuestraCiudad = 50000
NinasNaci = 11000
NinosNaci = 9000
MujeresNaci = 16000
HombresNaci = 14000

#Calculando La probabilidad de un evento A se define a priori
#(sin experimentación) como:
probabilidadNinasNaci = round((NinasNaci/MuestraCiudad),2)
probabilidadNinosNaci = round((NinosNaci/MuestraCiudad),2)
probabilidadMujeresNaci = round((MujeresNaci/MuestraCiudad),2)
probabilidadHombresNaci = round((HombresNaci/MuestraCiudad),2)

# print("Probabilidad de una niña en la ciudad: ", probabilidadNinasNaci )
# print("Probabilidad de un niño en la ciudad: ", probabilidadNinosNaci )
# print("Probabilidad de una mujer en la ciudad: ", probabilidadMujeresNaci )
# print("Probabilidad de hombre en la ciudad: ", probabilidadHombresNaci)
#se calcula probailidad Poblacion infectada
PoblacionInfectadaTotal = 9000

CasosNinas = 2000
CasosNinos = 1500
CasosMujeres = 3000
CasosHombres = 2500

#Calculando La probabilidad de un evento A se define a priori
#(sin experimentación) como:
probabilidadNinasInfec = round((CasosNinas/PoblacionInfectadaTotal),2)
probabilidadNinosInfec = round((CasosNinos/PoblacionInfectadaTotal),2)
probabilidadMujeresInfec = round((CasosMujeres/PoblacionInfectadaTotal),2)
probabilidadHombresInfec = round((CasosHombres/PoblacionInfectadaTotal),2)

# print("Probabilidad de una niña infectada en la ciudad: ",
→probabilidadNinasInfec )
# print("Probabilidad de un niño infectado en la ciudad: ",
→probabilidadNinosInfec )
# print("Probabilidad de una mujer infectada en la ciudad: ",
→probabilidadMujeresInfec )
# print("Probabilidad de hombre infectado en la ciudad: ",
→probabilidadHombresInfec)

# Datos generales
p_nacimientos = [probabilidadNinasNaci, probabilidadNinosNaci,
→probabilidadMujeresNaci, probabilidadHombresNaci]

```

```

#p_nacimientos = [0.22, 0.18, 0.32, 0.28]
p_infectados = [probabilidadNinasInfec, probabilidadNinosInfec,
    →probabilidadMujeresInfec, probabilidadHombresInfec]
#p_infectados = [0.22, 0.17, 0.33, 0.28]
###
####Con  $P(A), P(B)] = p\_nacimientos$  y
### $P(F|A), P(F|B)] = p\_infectados$ , entonces
#
#      0
#      / \
#      /   \
#      P(A)  P(B)
#      /     \
#      P(F|A)  P(F|B)
#      /       \
#      F         F
#####

# Probabilidad total de infectarse
p_infeccion = sum([p_nacimientos[d] * p_infectados[d] for d in
    →range(len(p_nacimientos))])

####
##Si  $P(F) = p\_infectados$  es la probabilidad de infectarse
##(y de que se den cuenta), entonces el teorema
##de Bayes queda como:
##
##       $P(F|A) P(A)$ 
## $P(A|F) = \frac{-----}{P(F)}$ 
##
##

# Aplicación de Teorema de Bayes
p_bayes = [(p_infectados[i]*p_nacimientos[i]) / p_infeccion for i in
    →range(len(p_nacimientos))]
print("La probabilidad de infección de en niñas es {A:0.2f}, el niños {B:0.2f},
    →el mujeres {C:0.2f} y el de hombres {D:0.2f}".format(A=p_bayes[0]*100,
    →B=p_bayes[1]*100, C=p_bayes[2]*100, D=p_bayes[3]*100))

```

```

(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$ python3 InfeccionPoblacionTeoremaBayes.py
Hay una ciudad de 50 000 habitantes, con la siguiente distribución de población: Niñas 11000 Niños 9000 Mujeres 16000 Hombres 14000. Hay también un reporte de 9000 casos de una nuev
a variedad de virus que-no-debe-ser-mencionada, distribuidos de la siguiente forma: Niñas 2000 Niños 1500 Mujeres 3000 Hombres 2500 ¿Está la probabilidad de contraer el virus relaci
onada con la pertenencia a un sector demográfico? Esto puede estudiarse analizando las probabilidades  $P(\text{tener gripe}|\text{pertenece a X sector})$ 
La probabilidad de infección de en niñas es 18.40, el niños 11.63, el mujeres 40.15 y el de hombres 29.81
(base) Belindas-MacBook-Air:Lab1ConceptosProbabilidad belindabrown$

```