**深 圳 大 学 实 验 报 告**

**课程名称：­ 概率论与数理统计**

**实验项目名称：** Axiom of Probability in Python

**学院： 电子与信息工程学院**

**专业： 电子信息工程**

**指导教师： 陈昌盛**

**报告人： 杨烨 学号**： **2022280380**

**班级： 06**

**实验时间： 2023年9月18日**

**实验报告提交时间： 2023年9月24日**

**教务处制**

|  |
| --- |
| Aim of Experiment:   1. Axiom of Probability in Python 2. Employ probability and statistics tools to solve practical problems 3. In order to master the random number function in numpy 4. Use python to implement the operation and verification between collections |
| Experiment Content:  This experiment is divided into two modules:  Module 1: Verifying the Fairness of Coin Tossing：   1. Define a function to generate random coin toss results：   （2）Define a function to estimate the probability of getting a number of heads between k1 and k2-1 when flipping the coin 100 times:    Module 2: Evaluate the intersection and complement of sets:  (1) Evaluate the intersection and complement of sets:  (2) Calculate the union of sets and their lengths:  (3) Perform operations on three sets: |
| Experiment Process：  This experiment is divided into two modules:  Module 1: Verifying the Fairness of Coin Tossing：   1. Define a function to generate random coin toss results：   Code:  def seq\_sum(n):  flips = np.random.rand(n) # 生成 n 个随机数，范围在 [0,1) 之间  heads = np.sum(flips > 0.5) # 统计正面朝上的硬币数量  return heads    #  （2）Define a function to estimate the probability of getting a number of heads between k1 and k2-1 when flipping the coin 100 times:  Code:  def estimate\_prob(n,k1,k2,m):  """  Estimate the probability that n flips of a fair coin result in k1 to k2 heads  n: the number of coin flips (length of the sequence)  每轮投掷硬币次数：100次  k1, k2: the trial is successful if the number of heads is between k1 and k2-1  如果每轮投掷硬币正面朝上次数小于k2大于k1，则总符合数+1  m: the number of trials (number of sequences of length n)  试验的轮数（总次数）：1000轮  output: the estimated probability  """    suc = 0  for \_ in range(m):  flips = np.random.randint(2, size = n)  #生成一个长度为 n 的数组，其中每个元素是从范围 [0, 2)  内随机生成的整数  heads = np.sum(flips) # 统计正面朝上的硬币数量  if k1 <= heads < k2: # 如果每轮投掷硬币正面朝上次数小于k2大于k1，则总符合数+1  suc += 1  return suc / m # 返还概率值  Module 2: Evaluate the intersection and complement of sets:  (1) Evaluate the intersection and complement of sets:  def complement\_of\_union(A, B, U):  print(f"({A|B}, {U - (A|B)})")  # 输出A和B的并集，以及输出A和B的并集的补集  return A|B, U - (A|B)    def intersection\_of\_complements(A, B, U):  complementary\_A = U - A # A的补集  complementary\_B = U - B # B的补集  x = complementary\_A&complementary\_B  # A的补集和B的补集的交集  print(f"({complementary\_A}, {x})")  return (complementary\_A, x)  (2)Calculate the union of sets and their lengths:  def union(A, B):  union\_set = A|B  module = len(union\_set) # A和B的并集的模长  print(f"({union\_set}, {module})")  return (union\_set, module)    def inclusion\_exclusion(A, B):  module\_A = len(A)  module\_B = len(B)  inter\_set = A&B # A和B的交集  union\_set = A|B # A和B的并集  module\_inter\_AB = len(inter\_set) # A和B的交集的模长  module\_union\_AB = len(union\_set) # A和B的并集的模长    print(f"({module\_A}, {module\_B}, {module\_inter\_AB}, {module\_union\_AB})")  print(f"notice: {module\_A} + {module\_B} - {module\_inter\_AB} = {module\_union\_AB}")  return(module\_A, module\_B, module\_inter\_AB, module\_union\_AB)  (3) Perform operations on three sets:  def union3(A, B, C):  print(f"{A|B|C}, {len(A|B|C)}") # A B C三个集合的并集以及其模长  return (A|B|C, len(A|B|C))      def inclusion\_exclusion3(A, B, C):  print(f"({len(A&B&C)}, {len(A|B|C)})")  # A B C三个集合的交集以及其模长  return (len(A&B&C),len(A|B|C)) |
| Data Logging and Processing:  In the first experiment named “**Verifying the Fairness of Coin Tossing**”  I got 48 heads out of 100 coin flips：  In the experiment of flipping a coin twice in 20 rounds, the number of times the coin heads in each round was：  [2, 1, 1, 1, 0, 1, 2, 0, 2, 1, 0, 1, 2, 0, 2, 1, 2, 1, 2, 1]  In 1000 rounds (with lots of experimental data) of 100 coin flips per round, if the number of coin flips per round is between 45 and 55, the probability that I will get between 45 and 55 after 1000 rounds is: 0.692  In the first experiment, we defined several subfunctions to calculate the median, variance, true probability, display histogram, probability truth line, medium estimate standard value line, and probability plus or minus one difference line：  The resulting probabilities are as follows:  #### test no. 1  computed prob=0.954, std=0.021  ran estimator 100 times, with parameters n=100,k1=40,k2=60,m=100  median of estimates=0.960, error of median estimator=0.006, std= 0.0208405.3  normalized error of median= 0.2639299423240206 should be <1.0  #### test no. 2  computed prob=0.159, std=0.037  ran estimator 100 times, with parameters n=100,k1=55,k2=100,m=100  median of estimates=0.190, error of median estimator=0.031, std= 0.0365355.3  normalized error of median= 0.8579273897059113 should be <1.0  #### test no. 3  computed prob=0.146, std=0.035  ran estimator 100 times, with parameters n=100,k1=47,k2=49,m=100  median of estimates=0.150, error of median estimator=0.004, std= 0.0353595.3  normalized error of median= 0.0993463603009432 should be <1.0  #### test no. 4  computed prob=1.000, std=0.000  ran estimator 100 times, with parameters n=1000,k1=400,k2=600,m=100  median of estimates=1.000, error of median estimator=0.000, std= 0.0000025.3  normalized error of median= 0.0001593621193426113 should be <1.0  #### test no. 5  computed prob=0.001, std=0.003  ran estimator 100 times, with parameters n=1000,k1=550,k2=1000,m=100  median of estimates=0.000, error of median estimator=-0.001, std= 0.0027975.3  normalized error of median= 0.27987751426889984 should be <1.0  #### test no. 6  computed prob=0.446, std=0.050  ran estimator 100 times, with parameters n=1000,k1=470,k2=499,m=100  median of estimates=0.435, error of median estimator=-0.011, std= 0.0497065.3  normalized error of median= 0.2192011030123324 should be <1.0  Draw the following graph:    The calculation of the set part is in agreement with the test result. |
| Experimental Results and Analysis:  In this experiment, I have accomplished the successful implementation of a Python code to simulate 100 coin tosses, along with a rational calculation using Python functions to determine the probability of achieving consistent outcomes over 1000 rounds.  Throughout this experiment, I acquired the knowledge of employing Python's library functions for simulating coin toss experiments and computing the probabilities associated with their outcomes. I have also gained proficiency in performing set operations using Python, thus enhancing my understanding of set theory and its applications. |
| 指导教师批阅意见：  成绩评定：  指导教师签字：  年 月 日 |
| 备注： |

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后10日内。