Assignment 07: Week 11-14 Basics of Programming
Practice of Basic Informatics – E2 (2024 Second Semester)
(Report Due: Jan. 24, 16:30)

Note:

1. In this assignment, you need to write four programs specified in the questions by using the Python programming language (Python 3).
2. You need to submit your report together with the source code of your four programs. You can either include all the source code in your report or submit the source code in separate files. In the case that you submit the source code in separate files, name your programs as YourLastName01.py, YourLastName02.py, YourLastName03.py, YourLastName04.py.
3. In your report, you need to describe how you design each program with discussions.
4. Please summarize whether you achieved your initial goals through taking this class.

Questions:

1. Write a program for the following purpose by using the `if ~ elif ~ else` statements: Generate a random score (an integer between `0` and `100`) and transform it into a category of `A~D` (`A: score>=80, B: 60<=score<80, C: 30<=score<60, D: score<30`).
   (1) Hint: Import the `random.randint` module (`from random import randint`) and then use the function `randint(0,100)` to generate a random score between `0` and `100`.
   (2) Execute your program for about five times and report the results. The execution result should be like:

```
Score: 78 (B)
Score: 19 (D)
Score: 93 (A)
Score: 63 (B)
Score: 47 (C)
```

2. Write a program for generating the `9*9` multiplication table by using the `for` loops.
   (1) The execution result should be like:

```
1*1= 1
1*2= 2 2*2= 4
1*3= 3 2*3= 6 3*3= 9
1*4= 4 2*4= 8 3*4=12 4*4=16
```

```
1*5= 5 2*5=10 3*5=15 4*5=20 5*5=25
1*6= 6 2*6=12 3*6=18 4*6=24 5*6=30 6*6=36
1*7= 7 2*7=14 3*7=21 4*7=28 5*7=35 6*7=42 7*7=49
1*8= 8 2*8=16 3*8=24 4*8=32 5*8=40 6*8=48 7*8=56 8*8=64
1*9= 9 2*9=18 3*9=27 4*9=36 5*9=45 6*9=54 7*9=63 8*9=72 9*9=81
```

(2) Hint: You will need to use multiple `for` loops in this program. Pay attention to the output format `print()`. In default, `print("Hello World")` will start a new line. Using `print("Hello World", end = '')` will not start a new line.

3. Write a program for the following purpose by using the `while` loop together with the `break` statement:

   Continuously generate three random integers between `1` and `13`, until the sum of the three integers is `21`. Print out the three integers when their sum is `21` and stop the loop.

   (1) Hint: Import the `random.randint` module (`from random import randint`) and then use the function `randint(1,13)` to generate a random integer between `1` and `13`.

   (2) Execute your program for about several times and report the results. The execution result should be like:

```
6 4 11
7 9 5
4 11 6
```

4. Write a program for simulating the Two Dice Sum Game (see Appendix 1) with user-defined functions.

   (1) First create a user-defined function `dice()` for rolling a die.

   - The `dice()` function does not have any input parameters.
   - The `dice()` function returns the random integer between 1 and 6.
   - Hint: Import the `random.randint` module (`from random import randint`) and then use the function `randint(1,6)` to roll a die.

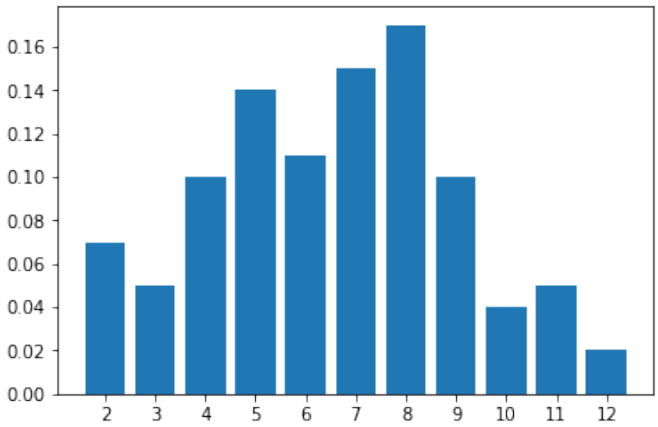
   (2) Then create a user-defined function `dicesum()` for rolling two dice and calculating their sum by using the `dice()` function created in the previous step.

   - The `dicesum()` function does not have any input parameters.
   - The `dicesum()` function returns three values: results of the two dice (each result is a random integer between `1` and `6`), and the two dice sum (an integer between `2` and `12`).

(3) Finally, create a user-defined function `dicegame(n)` for conducting the two dice sum simulation.

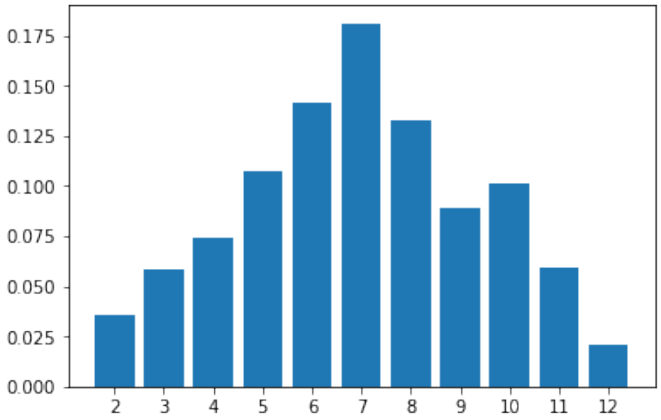
- The `dicegame(n)` function has a parameter `n`, which represents the simulation count.
- The `dicegame(n)` function does not return values. It calculates the experimental probabilities for all possible sums (from `2` to `12`) and prints out the probabilities with three digits after the decimal point for each value. Moreover, it plots the probability distribution as a bar graph.
- Hint: Use the `matplotlib.pyplot` package (`import matplotlib.pyplot as plt`) for plotting data as graph. See Appendix 2 for a sample program for plotting data.
- The execution result of `dicegame(100)`, `dicegame(1000)` and `dicegame(10000)` should be like:

```
Probability distribution of the dice sum with 100 simulations:
P(dice_sum= 2) is 0.070
P(dice_sum= 3) is 0.050
P(dice_sum= 4) is 0.100
P(dice_sum= 5) is 0.140
P(dice_sum= 6) is 0.110
P(dice_sum= 7) is 0.150
P(dice_sum= 8) is 0.170
P(dice_sum= 9) is 0.100
P(dice_sum=10) is 0.040
P(dice_sum=11) is 0.050
P(dice_sum=12) is 0.020
```

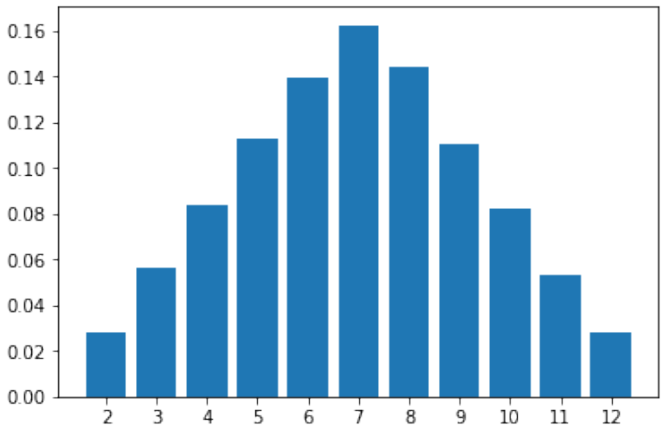



```
Probability distribution of the dice sum with 1000 simulations:
P(dice_sum= 2) is 0.036
```

P(dice_sum= 3) is 0.058
P(dice_sum= 4) is 0.074
P(dice_sum= 5) is 0.107
P(dice_sum= 6) is 0.141
P(dice_sum= 7) is 0.181
P(dice_sum= 8) is 0.133
P(dice_sum= 9) is 0.089
P(dice_sum=10) is 0.101
P(dice_sum=11) is 0.059
P(dice_sum=12) is 0.021



Probability distribution of the dice sum with 10000 simulation
s:
P(dice_sum= 2) is 0.028
P(dice_sum= 3) is 0.056
P(dice_sum= 4) is 0.084
P(dice_sum= 5) is 0.113
P(dice_sum= 6) is 0.139
P(dice_sum= 7) is 0.162
P(dice_sum= 8) is 0.144
P(dice_sum= 9) is 0.110
P(dice_sum=10) is 0.082
P(dice_sum=11) is 0.053
P(dice_sum=12) is 0.028



Appendix 1:

Two Dice Sum Game

A traditional die is a cube, with each of its six faces showing a different number of dots (pips) from 1 to 6. Pick up a pair of dice and roll them. When they stop, two faces will be showing. The dice might show 1 and 3, or maybe 2 and 4. Roll them again and there is a good chance that a different pair of faces will show up. We say that the experimental result is not reproducible and that the results from the dice fluctuate. By rolling two dice, the sum of the scores on the two dice is an integer between 2 and 12. Let's look at the theoretical probabilities for all possible sums.

Sum on dice	Pairs of dice	Probability
2	1+1	1/36 = 0.028
3	1+2, 2+1	2/36 = 0.056
4	1+3, 2+2, 3+1	3/36 = 0.083
5	1+4, 2+3, 3+2, 4+1	4/36 = 0.111
6	1+5, 2+4, 3+3, 4+2, 5+1	5/36 = 0.139
7	1+6, 2+5, 3+4, 4+3, 5+2, 6+1	6/36 = 0.167
8	2+6, 3+5, 4+4, 5+3, 6+2	5/36 = 0.139
9	3+6, 4+5, 5+4, 6+3	4/36 = 0.111
10	4+6, 5+5, 6+4	3/36 = 0.083
11	5+6, 6+5	2/36 = 0.056
12	6+6	1/36 = 0.028

Here, we're going to see the different probabilities for each integer to appear for different trial times and represent this with a simulation program written in Python.

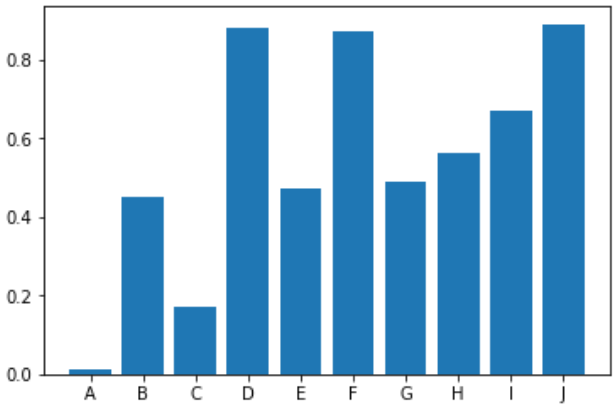
Appendix 2:

A sample of using `matplotlib.pyplot` for plotting data

Source code:

```
import matplotlib.pyplot as plt
labels = ["A", "B", "C", "D", "E", "F", "G", "H", "I", "J"]
x_pos = range(0, 10)
V = [0.01, 0.45, 0.17, 0.88, 0.47, 0.87, 0.49, 0.56, 0.67, 0.89]
plt.bar(x_pos, V, tick_label = labels)    # draw the graph
plt.show()    # show the graph
```

Execution result:



For more tutorials about `matplotlib.pyplot`, please refer to:

`https://matplotlib.org/stable/tutorials/introductory/pyplot.html`