

PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award
of Bachelor of Technology Degree in Information Technology

By
ASWIN KUMAR.K

38120015



DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING
SATHYABAMA INSTITUTE OF SCIENCE AND
TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119. TAMILNADU

January 2021



SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the Bonafide work of who carried out the project entitled "ANCIENT GAMES IN MODERN COMPUTER SYSTEM" under our supervision from August 2020 to November 2020.

Internal Guide

Dr. V.MARIA ANU M.E.,PH.D.

Head of the Department

Dr. R. Subhashini M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I hereby declare that the Project Report entitled “**Ancient Games In Modern Computer System**“ done by me under the guidance of **Dr. V.MARIA ANU M.E., Ph.D.**, at Sathyabama Institute of Science and Technology (Deemed to be University), Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600119 is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology.

Place: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge our sincere thanks to Board of management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey our thanks to **Dr. T. SASIKALA M.E., Ph.D., Dean, School of Computing** and **Dr. R. SUBHASHINI M.E., Ph.D., Head of the Department, Department of Information Technology** for providing us the necessary support and details at the right time during the progressive reviews.

I would like to express our sincere and deep sense of gratitude to our Project **Dr. V. MARIA ANU M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express our thanks to all Teaching and Non-teaching staff members of the Department of **INFORMATION TECHNOLOGY** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



RETECH
SOLUTIONS PVT. LTD.
An ISO 9001:2015 certified company

INT: RE20S035

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **ASWIN KUMAR K** student of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY, BTECH/IT** has undergone the Internship Training through online in our concern entitled **PYTHON** from **19th Aug 2020** to **21st Aug 2020** in relevant departments related to their academic studies.

During the above period, the performance was good & we wish great success in all your future endeavours.

Technology drives the world. We are the people who are behind the technology.

Authorised Signature



No-31, 1st Floor, Alagesan Street, Tambaram West, Chennai-45.

Office : 044-42077204 | +91-8122266653

www.retechsolutions.in

COURSE CERTIFICATE



ABSTRACT

Ancient Games In Modern Computer System

There are tons of games out there with random number generators of varying complexity that all run on your everyday, humble, 6-sided dice.

some dice that you roll one by one, decide when to stop, and get the final die roll as the outcome of the game.

Giving it a somewhat structured, mathematical notation, our game is noted as $G(n)$, where n is the potential maximum number of dice rolls. The game is played in a series of rounds, k number of rounds to be precise, where $k \leq n$. We start every round with a random state, noted by S , and the two indicators of the state are the current reward, $d(i)$, and the remaining rounds after the current state, i . Note that are “counting from the end”, i is the potential number of rounds remaining, rather than the rounds that have already passed. In every single state, $d(i)$ is a value of a random variable with discrete uniform distribution between 1 and 6 (i.e., the result of a completely balanced die roll). This will make more sense later on. $A(i)$ is the action the agent makes, based on information from the current state, $S(i, d(i))$. The two possible actions are *Continue* and *Stop*. If the chosen action is *Continue*, we go through the same process again, starting with a fresh die roll. If the chosen action is *Stop*, the game ends, with the current die roll as the result.

It probably think it is a bit of over-engineering, but building our problem in this structured way will make our life much easier going forward, when we come to the simulations and reinforcement learning.



FIGURE NO	FIGURE NAME	PAGE NO
CHAPTER NO	TITLE NAME	PAGE NO
	ABSTRACT	7
	LIST OF FIGURES	9
	LIST OF ABBREVIATION	10
1	INTRODUCTION	11
	1.1 GENERAL	11
	1.2 OUTLINE OF THE PROJECT	11
2	AIM AND SCOPE	12
	2.1 AIM	12
	2.2 PROBLEM STATEMENT	12
	2.3 SCOPE	12
3	SYSTEM ANALYSIS AND DESIGN	13
	3.1 GENERAL	13
	3.2 ALGORITHM	14
	3.3 FEASIBILITY STUDY	17
4	RESULT AND DISCUSSION	19
5	CONCLUSION AND FUTURE WORK	20
	5.1 CONCLUSION	20
	5.2 FUTURE WORK	20
	5.3 REFERENCE	20
	APPENDIX	21
	A.SCREENSHOTS	21
	B.SOURCE CODE	23

3.1	DATA FLOW	18
4.1	OUTPUT	19

LIST OF ABBREVIATIONS

ABBREVIATIONS	EXPANSION
Python	-

CHAPTER 1

INTRODUCTION

- This project concerned about developing a “ANCIENT GAMES IN MODERN COMPUTER SYSTEM”. One of the most beloved games of all time and a simple python project to test your skills.
- Start by making it player vs computer.
- Gain a better understanding of while loops.
- Snake and Ladders, Ludo and Checkers, are the most favorite games of all time. But, it feels terrible when you drop the plan of playing just because you couldn't find the dice. So, here is a fascinating option to give you a chance to make something cool.

1.2 OUTLINE OF THE PROJECT

Simple games

This section presents the implementation of some simple games based on drawing random numbers. The games can be played by two humans, but here we consider a human versus the computer.

Guessing a number

The game

The computer determines a secret number, and the player shall guess the number. For each guess, the computer tells if the number is too high or too low.

The implementation

We let the computer draw a random integer in an interval known to the player, let us say $[1,100]$. In a while loop the program prompts the player for a guess, reads the guess, and checks if the guess is higher or lower than the drawn number. An appropriate message is written to the screen. We think the algorithm can be expressed directly as executable Python code.

The following are specific objectives for this project ;

The game

The player is supposed to roll two dice, and beforehand guess the sum of the eyes. If the guess on the sum is nn and it turns out to be right, the player earns nn euros. Otherwise, the player must pay 1 euro. The machine plays in the same way, but the machine's guess of the number of eyes is a uniformly distributed number between 2 and 12. The player determines the number of rounds, rr , to play, and receives rr euros as initial capital. The winner is the one that has the largest amount of euros after rr rounds, or the one that avoids to lose all the money.

The implementation

There are three actions that we can naturally implement as functions: (i) roll two dice and compute the sum; (ii) ask the player to guess the number of eyes; (iii) draw the computer's guess of the number of eyes. One soon realizes that it is as easy to implement this game for an arbitrary number of dice as it is for two dice. Consequently we can introduce $ndice$ as the number of dice. The three functions take the following forms:

CHAPTER 2

AIM AND SCOPE

2.1 AIM

The main aim of the project is 'rolling_dice' function is a function that is executed every time a button is clicked. This is attained through the 'command=rolling_dice' parameter while defining a button.

2.2 PROBLEM STATEMENT

Here, we use **pack()** to arrange our widgets in row and column form. The 'BlankLine' label is to skip a line, whereas we use 'HeadingLabel' label to give a heading.

- **root** – the name by which we refer to the main window of the application
- **text** – text to be displayed in the HeadingLabel
- **fg** – the colour of the font used in HeadingLabel
- **bg** – background colour of the HeadingLabel
- **font** – used to give customised fonts to the HeadingLabel text
- **.pack()** – Used to pack the widget onto the root window

Step 4: Forming a list of images to be randomly displayed



2.3 SCOPE

simulating rolling 2 dice in Python, Roll the dice in the loop: `def dice(n): total = 0`
for i in range(n): total += random.randint(1, 6) return total. The += augmented
assignment operator We set two variables (min and max), lowest and highest
number of the dice. We then use a while loop, so that the user can roll the dice again.
The roll_again can be set to any value, but here it's set to "yes" or "y", but you can
also add other variations to it.

Rolling two dice and tabulating outcomes, Rolling two dice and tabulating
outcomes · python random. Here's the problem: You will write a program to simulate
the rolling of a pair of dice. We can run the program in the terminal with the command
`python3 dice_roll.py` and we will be given a random dice roll and then immediately
asked if we want to roll the dice again. If we type 'y' we will get another dice roll and
then when we type 'No' or anything other than 'yes' or 'y' the program quits. Terminal:

Roll pair of dice, It rolls two dice and gives the output of each dice. 1. `def roll():`. 2.
`import random`. 3. `a = random.randint(1, 6)`. 4. `b = random.randint(1, 6)`. 5. `print b`. 6.
`print a`. If instead of rolling n times, you need to produce n results of 2 dice rolls,

Python Project

Dice - Rolling Simulation



CHAPTER 3

SYSTEM ANALYSIS AND DESIGN

3.1 GENERAL

Python: how to simulate the roll of a dice 100 times, Creates a list of 100 rolls:
`import random print([random.randint(1,6) for _ in xrange(100)])`. indent your code properly and call `roll()` a hundred times, something like `for i in xrange(100): roll()` –
iruvar Jun 28 '13 at 15:56 What output do you expect? A list of the 100 rolls? –
dansalmo Jun 28 '13 at 16:09

Python Tutorial Chapter 5 by Marshall Brain, It rolls a simulated 6-sided dice 1,000 times and records how many times each face of the This new program uses a Python list to replicate the previous program: If you wanted to simulate a 100 face dice, you could do that too, even though What is the probability of rolling an 8, a 9, or a 10? If you roll the die 500 times, how many times do you expect to get a 5? In many games we don't roll one die, but two or more. Assume we roll 2 die. What is the probability that we roll a sum of 4? If we roll two dice 100 times, how many times do we expect to get a pair?

Rolling a Die « PD4CS, If we roll two dice 100 times, how many times do we expect to get a pair? The probabilities of some of the above questions may not be This is a classic "roll the dice" program. We will be using the random module for this,since we want to randomize the numbers

3.3 ALGORITHM

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

When you want to carry out a set of operations in Python, it's a good strategy to first write out the necessary steps in pseudocode. While pseudocode is not a programming language, it is similar to one because it helps you break down your task into instructions that can be carried out in code.

Think of pseudocode as a bridge between spoken language and code.

In English, the problem being solved is this: I want to compare the sum of two dice rolls to a guess I have made. If I guessed the sum correctly, I win. If not, I lose.

Let's put this in pseudocode. Here's what I want my code to do.

1. Simulate the roll of a die and store the result.

2. Simulate the roll of another die and store the result.
3. Add the results of the two simulations.
4. Input my guess.
5. Compare my guess to the actual answer.
6. Print out a statement declaring whether I won or lost.

While we could make this pseudocode even more technical, the steps above will hopefully give you an idea about the benefit of breaking down your process in this way. It clarifies what your code must accomplish.

As you read the steps above, it's a good idea to start thinking about what functions you will use. For example, I'll need a way to simulate randomness. I will also use the input function to enter in my own guess. I will use the sum function and the comparison operator '==' to compare my guess to the result. Finally, I will use a print statement along with a formatted string to output the result.

Use Python's random module to simulate the dice rolls.

Simulating the roll of a die in Python is done by randomly choosing an integer between 1 and 6, inclusive. This is where the random module saves the day.

This module can be imported into your IDE. Once you do that, you have access to all of the functions included in the module. It's quite an extensive list, but for our purposes, we'll be using the **random.randint()** function. This function returns a random integer based on the start and end that we specify.

The smallest value of a die roll is 1 and the largest is 6. This gives us the start and end values to use in our **random.randint()** function

I have included a print statement to do a check that my code is working how I want it to. This will not be part of the final code, but it's useful as I build out my program to make sure I don't have any bugs. Here are the results when I run my code three times.

I am getting random integers between 1 and 6, so the check is complete. Next, we repeat this step to simulate the roll of our second die.

```
second_roll = random.randint(1, 6)
```

Finally, we add the two variables to get the sum of the computer's dice roll.

```
comp_sum = first_roll + second_roll
```

Note that we cannot use **2 * random.randint(1, 6)** because this only simulates one roll and multiplies that value by two. We want to simulate two independent rolls.

Compare your answer to the result.

Next, we will write an **input()** function that prompts the user to enter their guess. Include the string you want to display in the syntax for your **input()** function.

The input always gets returned as a string data type, so to compare it with an integer, you need to convert the input type to an integer. Wrap the **input()** function with **int()** to accomplish this. Then, assign it all to the variable **guess**. Here's the syntax:

The final variable we will use is **result**. If the user guesses the correct sum, **result** will equal the string 'win'. Otherwise, **result** will equal the string 'lose'. We can use a simple **if statement** for this.

Finally, we can make our lives a little easier and our code a bit cleaner by using formatted strings in our **print statement**. This allows us to use just one set of quotation marks and include the values returned by variables by coating those variables inside of curly brackets.

Here's the finished code with the **if statement** and the **print statement**.

You are here: [Home](#) / [Code Snippets](#) / Python Game : Rolling the dice
Python Game : Rolling the dice

Last Updated: August 27, 2020

Time again for a game script.

How it works

This is a classic “roll the dice” program.

We will be using the random module for this, since we want to randomize the numbers we get from the dice.

We set two variables (min and max) , lowest and highest number of the dice.

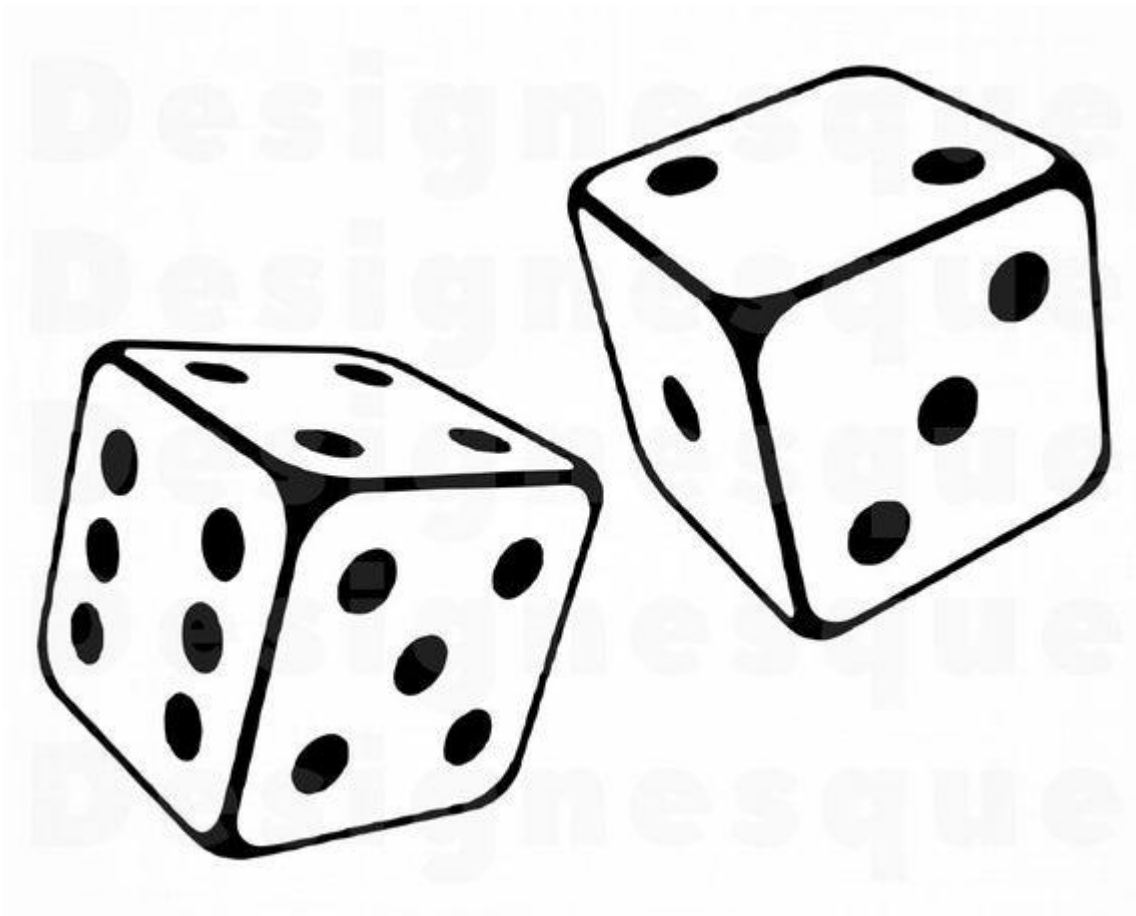
We then use a while loop, so that the user can roll the dice again.

The roll_again can be set to any value, but here it's set to "yes" or "y", but you can also add other variations to it.



APPENDIX

A.SCREENSHOTS



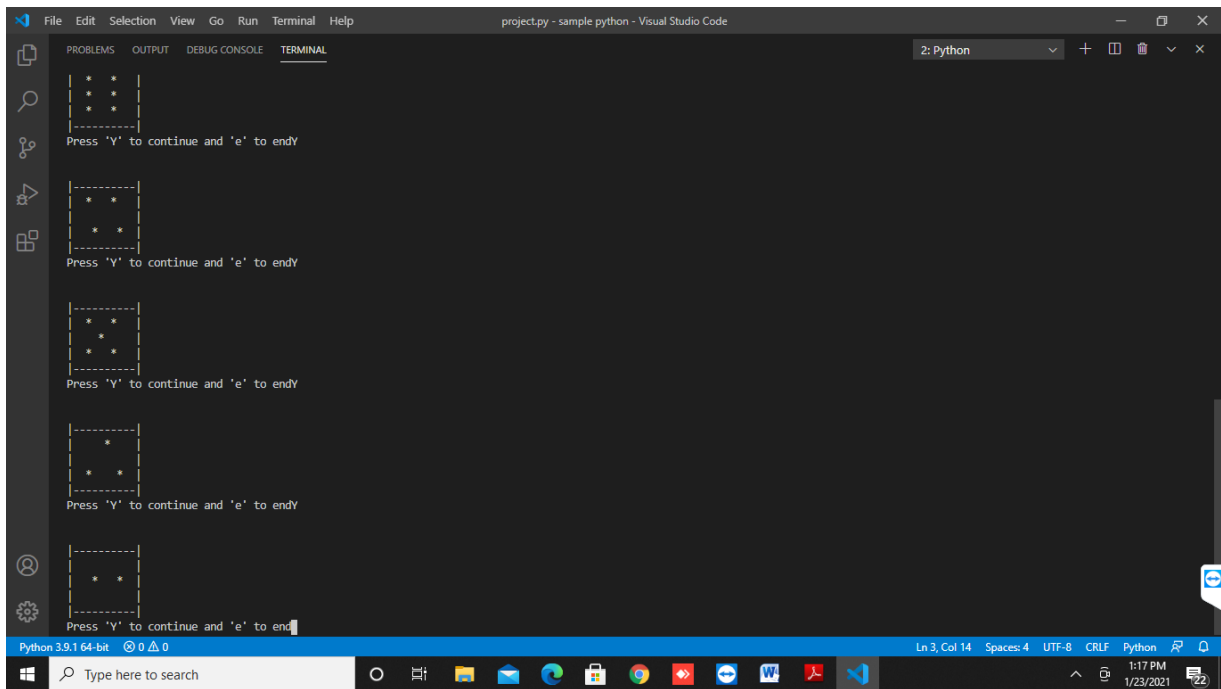
```
File Edit Selection View Go Run Terminal Help
project.py - sample python - Visual Studio Code

project.py x
project.py > ...
1 import random
2 X='Y'
3 while X=='Y':
4     no=random.randint(1,6)
5     if no==1:
6         print("|-----|")
7         print("|         |")
8         print("|      *      |")
9         print("|         |")
10        print("|-----|")
11
12    if no==2:
13        print("|-----|")
14        print("|         |")
15        print("|    * *    |")
16        print("|         |")
17        print("|-----|")
18
19    if no==3:
20        print("|-----|")
21        print("|         |")
22        print("|    * *    |")
23        print("|   * *   |")
24        print("|-----|")
25
26    if no==4:
27        print("|-----|")
28        print("|    * *    |")
29        print("|   * *   |")
30        print("|   * *   |")
31        print("|-----|")
32
33    if no==5:
34        print("|-----|")
35        print("|  * *  |")
36        print("| * * |")
37        print("|  * *  |")
38        print("|-----|")
39
40    if no==6:
41        print("|-----|")
42        print("|  * *  |")
43        print("| * * |")
44        print("| * * |")
45        print("|-----|")
46
47    x=input("Press 'Y' to continue and 'e' to end")
48    print("\n")
49

Python 3.9.1 64-bit 0 0
Ln 3, Col 14 Spaces: 4 UTF-8 CRLF Python
1:15 PM 1/23/2021
```

OUTPUT:

```
project.py - sample python - Visual Studio Code
2: Python
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\Microsoft\Desktop\sample python> & C:/Users/Microsoft/AppData/Local/Programs/Python/Python39/python.exe "c:/Users/Microsoft/Desktop/sample python/project.py"
|-----|
|  *  |
|-----|
Press 'Y' to continue and 'e' to end
Y
|-----|
|  *  |
|-----|
Press 'Y' to continue and 'e' to end
Y
|-----|
| * * |
| * * |
| * * |
|-----|
Press 'Y' to continue and 'e' to end
Y
Python 3.9.1 64-bit 0 0 0
Ln 3, Col 14 Spaces: 4 UTF-8 CRLF Python
Type here to search 1:16 PM 1/23/2021
```



B . SOURCE CODE

```
import random
X='Y'
while X=='y':
    no =random.randint(1,6)
    if no==1:
        print("|-----|")
        print("|          |")
        print("|      *      |")
        print("|          |")
        print("|-----|")

    if no==2:
        print("|-----|")
        print("|          |")
        print("|      *      |")
        print("|          |")
        print("|-----|")

    if no==3:
```

```

print(" |-----| ")
print(" |      *      | ")
print(" |              | ")
print(" |    *    *    | ")
print(" |-----| ")

if no==4:
    print(" |-----| ")
    print(" |    *    *    | ")
    print(" |              | ")
    print(" |    *    *    | ")
    print(" |-----| ")

if no==5:
    print(" |-----| ")
    print(" |    *    *    | ")
    print(" |      *      | ")
    print(" |    *    *    | ")
    print(" |-----| ")

if no==6:
    print(" |-----| ")
    print(" |    *    *    | ")
    print(" |    *    *    | ")
    print(" |    *    *    | ")
    print(" |-----| ")

x=input("Press 'y' to continue and 'e' to end")
print("\n")

```

- This can access in online or offline either from a computer or a smart phone, and fill the information and submit it for further approval.
- This game provides us with ease of access, user friendliness and transparency.

REFERENCE:

Experienced programmers will be very quick to point out that there are better ways to write a “DICE” Python game. But, for a beginner, it’s very important to be able to understand and follow the program.

THANK YOU