University of Engineering & Management, Kolkata

Department of Computer Science & Engineering

DATA STRUCTURE LAB

PYTHON PROJECT LIST

Subject Code - CS392

SESSION- 2018 ODD SEM

Submission Date-29.10.2018

RULES & REGULATIONS

- **Lab Project Carries 20 marks of the final semester lab examination.**
- **Each group consists of 4 members.**
- **Each group having a Project number which is same with the group number.**
- **The project is needed to be submitted on or before 29.10.2018.There will be no extension of the date of submission.**
- **The project is needed to be run on the day of submission and a project presentation will be there group wise.**
- **To complete the project it is advised to use "PYTHON" only.**

Python Project Topics

1. Dice Rolling Simulator

The Goal: Like the title suggests, this project involves writing a program that simulates rolling dice. When the program runs, it will randomly choose a number between 1 and 6. (Or whatever other integer you prefer — the number of sides on the die is up to you.) The program will print what that number is. It should then ask you if you'd like to roll again. For this project, you'll need to set the min and max number that your dice can produce. For the average die, that means a minimum of 1 and a maximum of 6. You'll also want a function that randomly grabs a number within that range and prints it.
Concepts to keep in mind:

- Random
- Integer
- Print
- While Loops

2. Guess the Number

The Goal: Similar to the first project, this project also uses the random module in Python. The program will first randomly generate a number unknown to the user. The user needs to guess what that number is. (In other words, the user needs to be able to *input* information.) If the user's guess is wrong, the program should return some sort of indication as to how wrong (e.g. The number is too high or too low). If the user guesses correctly, a positive indication should appear. You'll need functions to check if the user input is an actual number, to see the difference between the inputted number and the randomly generated numbers, and to then compare the numbers.
Concepts to keep in mind:

- Random function
- Variables
- Integers
- Input/Output
- Print
- While loops
- If/Else statements

Jumping off the first project, this project continues to build up the base knowledge and introduces user-inputted data at its very simplest. With user input, we start to get into a little bit of variability.

3. Mad Libs Generator

The Goal: Inspired by Summer Son's Mad Libs project with Javascript. The program will first prompt the user for a series of inputs a la Mad Libs. For example, a singular noun, an adjective, etc. Then, once all the information has been inputted, the program will take that data and place them into a premade story template. You'll need prompts for user input, and to then print out the full story at the end with the input included.
Concepts to keep in mind:

- Strings
- Variables
- Concatenation
- Print

A pretty fun beginning project that gets you thinking about how to manipulate userinputted data. Compared to the prior projects, this project focuses far more on strings and concatenating. Have some fun coming up with some wacky stories for this!

4. TextBased Adventure Game

The Goal: Remember *Adventure*? Well, we're going to build a more basic version of that. A complete text game, the program will let users move through rooms based on user input and get descriptions of each room. To create this, you'll need to establish the directions in which the user can move, a way to track how far the user has moved (and therefore which room he/she is in), and to print out a description. You'll also need to set limits for how far the user can move. In other words, create "walls" around the rooms that tell the user, "You can't move further in this direction."
Concepts to keep in mind:

- Strings
- Variables
- Input/Output

- If/Else Statements
- Print
- List
- Integers

The tricky parts here will involve setting up the directions and keeping track of just how far the user has "walked" in the game. I suggest sticking to just a few basic descriptions or rooms, perhaps 6 at most. This project also continues to build on using user-inputted data. It can be a relatively basic game, but if you want to build this into a vast, complex word, the coding will get substantially harder, especially if you want your user to start interacting with actual objects within the game. That complexity could be great, if you'd like to make this into a long term project. *Hint hint.

5. Hangman

The Goal: Despite the name, the actual "hangman" part isn't necessary. The main goal here is to create a sort of "guess the word" game. The user needs to be able to input letter guesses. A limit should also be set on how many guesses they can use. This means you'll need a way to grab a word to use for guessing. (This can be grabbed from a pre-made list. No need to get too fancy.) You will also need functions to check if the user has actually inputted a single letter, to check if the inputted letter is in the hidden word (and if it is, how many times it appears), to print letters, and a counter variable to limit guesses.
Concepts to keep in mind:

- Random
- Variables
- Boolean
- Input and Output
- Integer
- Char
- String
- Length
- Print

Likely the most complex project on this list (well, depending on just how intense you went with the adventure text game), the Hangman project compiles the prior

concepts and takes them a step further. Here, outcomes are not only determined based on user-inputted data, that data needs to be parsed through, compared, and then either accepted or rejected. If you want to take this project a step further, set up a hangman image those changes!

6. Input N numbers from user and store it in a python list. Iterate through the list and if the number is odd then store it in a dictionary with the number as key and "odd" as value else, the number as key and "even" as value.(For example, L=[1,2,4,5]
   D = {"1": "odd", "2": "even", "4": "even", "5": "odd"}

7. Maintain a database as dictionary for storing the personal details along with marks v obtained. Now retrieve the elements and calculate the percentage marks for each student.

8. Trip package details for each person
   Manali :Rs. 8000, Goa : Rs. 7500, Gangtok : Rs. 7000, Puri : Rs. 3000
   Use a python dictionary to store all these data. Suppose, a family of 6 members want to book such trips. Two of them pay for Manali, One Goa, and the other three pays for Gangtok. Write a program in Python to find out the total expense of the family using the provided dictionary.

9. Represent a graph using dictionary in python.
   (For example, the edges can be taken as input for the given graph like: AB, AC, BD Now store it using a dictionary like : {"A" : ["B","C"], "B":[D]}. Now, given a node, its degree should be displayed along with the names of the adjacent nodes.)

Submission Date Remains the same as of C Projects for further queries please talk to

Prof. Arunabha Tarafdar

Prof. Satyam Raha