**Hundred Python Projects**

**Project 1: Payment Receipt**

Create a python program that does the following:

1. Ask a customer for the names and prices of the three items he/she is buying
2. Then output a receipt similar to the receipt below.
   * 1. The company name and address can be different from what is on the sample receipt
     2. The product names and prices can be different
     3. The total price should be computed by the program based on the prices entered by the user/customer
     4. The program output should look exactly like the sample receipt below showing all the borders (\*\*) .



# *Source: Connor P. Milliken (2020) Python Projects for Beginners : A Ten-Week Bootcamp Approach to Python Programming*

Project 2

Timed Recipe

Print a given recipe such that:

1. The ingredients are first displayed
2. Each step of the recipe is displayed such that after each step there will be a 5 seconds lapse before the subsequent step is printed out . For example if the recipe has three steps, step 1 will be printed and then there will be a 5 seconds pause before step 2 is printed.

Hint: time.sleep() function

Project 3

Make the “High–Low” party games

At the beginning of this book, I said that programming is as easy (or difficult) as organizing a party. With that in mind, here are a couple ideas for Python programs that you can use for entertainment at your next party.

You can use the random number generator and the sleep function to make a high–low party game. The game works like this:

1. The program displays a number between 1 and 10, inclusively.
2. The program then sleeps for 20 seconds. While the program is asleep, the players are invited to decide whether the next number will be higher or lower than the number just printed. Players who choose “high” stand on the right. Players who choose “low” stand on the left.

3. The program then displays a second number between 1 and 10, and anyone who was wrong is eliminated from the game. The program is then re-run with the players that are left until you have a winner.

This game can get very tactical, with players taking a chance on an unlikely number just so that they will be one of the people to go forward to the next round.

The game could be renamed “Self-Timer,” and the winner would be the person who was best at keeping track of time. The sequence of actions the program must follow are:

1. Set the time to remain standing to a random number.
2. Display the time to remain standing.
3. Sleep for the time to remain standing.
4. Display the “Time Up” message

The program will need to use a variable to store the random number of seconds for which the players must remain standing. The name stand\_time would work well for such a variable.

Project 4

Create a simple calculator which can perform basic arithmetic operations like addition, subtraction, multiplication or division depending upon the user input.  
**Approach :**

 User choose the desired operation. Options 1, 2, 3 and 4 are valid.

 Two numbers are taken and an if…elif…else branching is used to execute a particular section.

 Using functions add(), subtract(), multiply() and divide() evaluate respective operations.

<https://www.geeksforgeeks.org/make-simple-calculator-using-python/>

<https://techvidvan.com/tutorials/python-calculator-program/>

**Given weight and height of a person and we have to find the BMI (Body Mass Index) using Python.**

Project: Amusement park ride selector program

Use decisions to make an application

Now that you know how to make decisions in your programs, you can start to make more useful software. Let’s say your next-door neighbor is the owner of a theme park and has a job for you. Some rides at the theme park are restricted to people by age, and he wants to install some computers around his theme park so that people can find out which rides they may go on. He needs some software for the computers, and he’s offering a season pass to the park if you can come up with the goods, which is a very tempting proposition. He provides you with the following information about the rides at his park:

|  |  |
| --- | --- |
| RIDE NAME MININUM AGE REQUIREMENT | |
| Scenic River Cruise None | |
| Carnival Carousel At least 3 years old | |
| Jungle Adventure Water Splash At least 6 years old | |
| Downhill Mountain Run At least 12 years old | |
| The Regurgitator (a super scary roller coaster) | Must be at least 12 years old and less than 70 |

You discuss with him the design of the program. Users will select the ride they want to go on. The program will ask for their ages and then display a message indicating whether they can go on this ride. For now, your customer is happy with text input, but later he would like to move to a graphical user interface with touch buttons.

Output

Welcome to our Theme Park

These are the available rides:

1. Scenic River Cruise  
2. Carnival Carousel 3. Jungle Adventure Water Splash  
4. Downhill Mountain Run  
5. The Regurgitator

Please enter the ride number you want: 1

You have selected the Scenic River Cruise

There are no age limits for this ride

#### [**Anagram Game**](https://favtutor.com/blog-details/anagram-game-using-python)

#### **https://favtutor.com/blog-details/7-Python-Projects-For-Beginners**

An Anagram is a word formed by shuffling the letters of another word. The words ***silent*** and ***listen*** are anagrams of each other. Implement an anagram game that prompts the user to guess the correct word from its given anagram. For instance, the correct answer to the anagram of ***hpotyn*** is ***python***. The in-built module random will be helpful in implementing this project.

**Concepts**: Console Input/Output, Strings, Conditional & Control Flow  
**Advanced**: Implement a scoring system based on the number of attempts taken to arrive at the correct answer.

#### **Login System**

<https://favtutor.com/blog-details/7-Python-Projects-For-Beginners>

Various forms of authentication are present everywhere. The most popular authentication method for software applications is the username-password paradigm. For a user to be able to be authenticated, their details must be stored in some form of a database.  
Implement a project that must be able to register a new user by storing their username and password details in a text file. Meanwhile, existing users must be authenticated by comparing their usernames with the corresponding password stored in the text file mentioned earlier.

**Concepts**: Console Input/Output, Text File Handling  
**Advanced**: Storing passwords in their text-form is a security vulnerability. Implement some kind of hashing concept to protect passwords as well as the text-field that contains user data.

#### **Python Excel**

<https://favtutor.com/blog-details/7-Python-Projects-For-Beginners>

Python is a very powerful language and proves to be a useful tool for manipulating spreadsheets such as Excel workbooks. Implement a personal finance manager that keeps a record of all your transactions. Inputs given from the console must be stored in an Excel workbook according to the specific column. There are several libraries that help with reading and writing Excel files such as openpyxl, xlrd, pyexcel,etc.

**Concepts**: Console Input/Output, Data Structures, Excel Workbook Handling  
**Advanced**: Perform basic data analysis on the data stored through the implementation of this project. This can include visualizing monthly expenditure or category-based spending.

#### **4. Interactive Dictionary**

Build your own Python dictionary program, which prompts a user to enter a word, and the program returns its meaning. Data of a dictionary exists as a 5MB JSON file that can be found here: [Data JSON](https://www.filehosting.org/file/details/863408/data.json). The programmer should take care of case-sensitive user inputs. The jsonmodule will be required for working with the data file for this implementation.

**Concepts**: Console Input/Output, Data Structures, Library Usage  
**Advanced**: Users may input an incorrectly spelled word or another word that is close to the one intended. The programmer can account for this using the difflibmodule. This would make a dictionary more user-friendly.

#### **5. Tic-Tac-Toe**

This project includes programming a Python logic for three in a row to win the tic-tac-toe game. Implement a 3x3 tic-tac-toe game board using a suitable data structure and allow a user to make their mark on this board. The program must then place its mark accordingly after checking the rows, columns, and diagonals. The objective of the program should be to win the game by placing three in a row before the user.

**Concepts**: Console Input/Output, Multi-Dimensional Data Structures, Conditional & Control Flow, Loops  
**Advanced**: Implement a more user-friendly interface version of the game using the pygame library.

#### **6. Python Turtle**

Python “Turtle” is a feature that lets the programmer draw over a board using a turtle through programming commands. The turtle module includes functions such as right() and left() to control the movement of the turtle and hence draw graphics on the screen. Implement various functions of the turtle module using loops and conditionals to draw graphics such as shapes and spirals on the screen.

**Concepts**: Graphics, Library Usage, Loops, Functions  
**Advanced**: Implement the “Snake” game, arguably the most popular game using the Python turtle module.

#### **7. Graphical User Interface - Calculator**

Users prefer graphical interfaces rather than text based consoles for their applications. Hence, it is important to learn how to program GUIs with Python. Design a calculator GUI that implements the four basic operations – addition, subtraction, multiplication and division on two numbers. The GUI should include the numbers and operators as buttons. PyQt5 and Tkinter are the most popular libraries in Python for building graphical user interfaces.

**Concepts**: GUI Design, Library Usage, Functions  
**Advanced**: Implement complex scientific functions similar to the default calculator applications available on mobile phones.

### 1: ODD OR EVEN?

Let’s start with a super easy project you can build right away. Write a small Python program that asks the user for a number and tells them if it is odd or even.

To help you get started, [here’s a good article about the Modulo operator](https://www.freecodecamp.org/news/the-python-modulo-operator-what-does-the-symbol-mean-in-python-solved/) in Python.

### 2: GUESSING THE NUMBER

Making a user guess a number is another easy Python program you can write in just a few lines of code. It should generate a random number between, let’s say, 0 and 100. The player then needs to guess what that number is.

Should the user guess wrong, the program should respond by telling them their guess is either too low or too high. When the user guesses right, your program should ask them if they want to play again.

For a little added challenge, you can limit the number of guesses to 5, for example.

### 3: HOROSCOPE

Create a simple horoscope program that asks the user for their star sign and outputs a fun horoscope for them. Bear in mind that your program should display an error message if the user types in their sign wrong.

### 4: ROCK PAPER SCISSORS

Rock Paper Scissors (or RPS) is one of my favorite Python projects for beginners. It’s simple enough but still challenging since you need to think about how to turn the game rules into logic for the program.

The rules are simple:

1. Paper wins over rock
2. Rock wins over scissors
3. Scissors win over paper

Have your program generate a random answer for the computer – but don’t display it. Then, ask the player for their answer.

### 5: HANGMAN

Hangman is one of the best Python projects for beginners who want a bit of a challenge. The game is about guessing a random word with a limited number of guesses.

Although Hangman may sound rather simple, you need to consider a few key points, such as:

* You need to set a maximum limit for guesses.
* The player needs to be notified about the remaining number of guesses.
* Your player needs to be able to input their guesses.

To get started, you need a way to generate random words to be guessed. The best way is to simply create a separate text file where you store the words or short phrases.

This Python project will probably take you a bit longer, but it’s great practice. You will have to think about random choice, variables, boolean values, inputs and outputs, strings, length, and much more.

To build a very basic text-based Hangman game, here’s a step-by-step tutorial video for some helpful tips:

### 6: ACRONYM

An acronym Python program will take a given phrase or text and convert it into its acronym. That is, a word that consists of the first letters of each word in the text.

To make your acronym easier to read, go ahead and turn the letters into uppercase with Python. Also, think about how the program will take text as input from the user while separating that user input from the logic itself.

### 7: PASSWORD GENERATOR

Create a Python program that generates a random password for the user. Make sure your program takes a few inputs from the user:

1. How long should the password be?
2. How many characters should there be?
3. Should it have both uppercase and lowercase letters?
4. Should it include numbers and special symbols, too?

The best part about this small Python project is that you can actually use it for generating strong passwords for your own user accounts across the Web!

### 7: WORD COUNT

Write a Python program that takes a text file as an input and outputs how many times each word occurs in the text.

You can start small by counting the number of words in a list, for example. Use the [Counter collection](https://docs.python.org/2/library/collections.html#collections.Counter) for this.

Once you understand how the collection works, move on to more advanced inputs, such as longer text files. You can access complete book scripts for free at [Project Gutenberg](https://www.gutenberg.org/) for this.

### 8: PONG GAME

Creating a simple Pong game is one of the easiest projects to learn how to use Python for simple video games. You can use the [Pygame library](https://www.pygame.org/wiki/GettingStarted" \t "_blank) to speed up your workflow for designing and writing your game.

Using Python to build simple but fun games is definitely one of the most enjoyable ways to learn the language. Moreover, you will get more ideas for developing your game further and adding new features while you’re playing.

If you’re ready to start building your first video game with Python right away, here’s a full video course from freeCodeCamp including a Pong game tutorial:

### 10: DATA ANALYSIS WITH PYTHON

Conducting data analysis with Python is a great idea for an intermediate Python project.

Python supports a number of powerful libraries you can use for analyzing, visualizing, and mining data. Small data analysis projects will familiarize you with a few popular Python libraries, such as Matplotlib, NumPy, and Pandas, for instance.

My first Python book, [*Python Crash Course*](https://mikkegoes.com/book-python-crash-course) by Eric Matthes, had a fun project where I learned how to do data analysis with Python. If you’re looking for an all-in-one Python book for absolute beginners, I definitely recommend checking it out.

If you learn better by watching videos, here’s one of my favorite videos for learning the basics of Matplotlib and Pandas for absolute beginners:

<https://www.upgrad.com/blog/python-projects-ideas-topics-beginners/>

### **Beginner Level Projects in Python**

* Hangman Game With Python
* Snake Game Using Pygame
* Scientific Calculator Using Python
* Product landing Page using Python Flask
* URL Shortener Using Python

### **Intermediate Level Projects in Python**

* Web Scraping Using Python
* Exploratory Data Analysis
* Pong Game using Kivy in Python
* Login System Using Python Flask/Django Web Frameworks
* Survival Prediction on Titanic Data

### **Advanced Level Projects in Python**

* Face Mask Detection Using OpenCV Python
* Speech Recognition Using Python
* Text To Speech Using Python
* Chatbot In Python
* Web browser Automation Using Selenium