BITQUANTA

Python Programming Course Assignment

Learn Python from Scratch to Project Level

(Real-world Projects & Examples)

AUTHOR

Created by: Muhammad Ishfaq Khan

Email: ishfaqkhan.dev@gmail.com

YouTube Channel: @Bitquanta

GitHub: @ishfaqkhan-dev

Chapter 2: Python Basics

1. Create a program that asks the user for their name and age, then prints a greeting with their name and age.
2. Convert a float number to an integer and print both values.
3. Use different types of operators to perform calculations on two numbers (addition, subtraction, comparison).
4. Write a program that takes user input and prints it in uppercase.
5. Concatenate two strings and find the length of the final string.

Chapter 3: Control Flow

1. Write a program to check if a number is positive, negative, or zero using if-elif-else.
2. Ask the user to enter their age and print whether they are eligible to vote (age >= 18).
3. Create a program with nested if statements to determine the largest of three numbers.
4. Use a match-case structure to print the day of the week based on a number (1 to 7).
5. Write a simple login system using if-else to check username and password.

Chapter 4: Loops and Iteration

1. Print the first 10 natural numbers using a for loop.
2. Use a while loop to print even numbers between 1 and 20.
3. Use the range() function to print numbers from 5 to 50 with a step of 5.
4. Iterate over a list using enumerate() and print index and item.
5. Create a loop that skips the number 3 using continue and stops at 5 using break.

Chapter 5: Data Structures

1. Create a list of 5 numbers and print the square of each number using list comprehension.
2. Create a tuple with three values and unpack them into variables.
3. Create two sets and find their union and intersection.
4. Create a dictionary with three key-value pairs and print all keys and values.
5. Create a nested dictionary to store student info (name, age, grade) and display the data.

Chapter 6: Functions

1. Write a function that takes two numbers and returns their sum.
2. Create a function that prints the factorial of a number.
3. Define a function using *args and return the sum of all arguments.
4. Write a lambda function to multiply a number by 10.
5. Use the built-in max() function to find the largest number in a list.

Chapter 7: Error and Exception Handling

1. Write a program that handles division by zero using try-except.
2. Use a try-except block to handle invalid input (e.g. non-integer input).
3. Raise a ValueError if the input age is negative.
4. Write a function that catches multiple exceptions (e.g. IndexError, ValueError).
5. Use finally to print "Program completed" regardless of whether an error occurred.

Chapter 8: File Handling

1. Write a program to create a new text file and write some lines into it.
2. Read a file and print each line in uppercase.
3. Append a new line to an existing file without overwriting it.
4. Use the with statement to safely open and read a file.
5. Read a CSV file and print each row (use csv.reader).

Chapter 9: Object-Oriented Programming (OOP)

1. Create a class Car with attributes like brand and model, and print them using an object.
2. Add an __init__ method to initialize object data.
3. Create a class with a class variable and instance variable. Print both using an object.
4. Create a base class Animal and a derived class Dog using inheritance.
5. Demonstrate polymorphism by creating a method with the same name in two different classes.

Chapter 10: Modules and Packages

1. Import the math module and use it to calculate the square root of a number.
2. Create a custom module with a function and import it into another file.
3. Use the random module to generate a random number between 1 and 100.
4. Write a script that prints a message only if __name__ == "__main__" is true.
5. Create a package with two modules and demonstrate how to import them.