



**ITM** SKILLS  
UNIVERSITY

**INSTITUTE OF TECHNOLOGY AND MANAGEMENT  
SKILLS UNIVERSITY,  
KHARGHAR, NAVI MUMBAI**

## **PYTHON PROGRAMMING LAB**



**Prepared by:**

Name of Student: Shikha singh\_\_\_\_\_

Roll No: \_25\_\_\_\_\_

Batch: 2023-27

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

exp. No	List of Experiment
	1.1 Write a program to compute Simple Interest.
	1.2 Write a program to perform arithmetic, Relational operators.
	1.3 Write a program to find whether a given no is even & odd.
	1.4 Write a program to print first n natural number & their sum.
	1.5 Write a program to determine whether the character entered is a Vowel or not .
	1.6 Write a program to find whether given number is an Armstrong Number.
	1.7 Write a program using for loop to calculate factorial of a No.
	1.8 Write a program to print the following pattern
	i) <pre> * * * * * * * * * * * * * * *</pre>
	ii) <pre> 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5</pre>
	iii) <pre>       *     * * *   *</pre>

	2.1 Write a program that define the list of defines the list of define countries that are in BRICS.
	2.2 Write a program to traverse a list in reverse order. 1.By using Reverse method. 2.By using slicing
	2.3 Write a program that scans the email address and forms a tuple of username and domain.
	2.4 Write a program to create a list of tuples from given list having number and add its cube in tuple. i/p: c= [2,3,4,5,6,7,8,9]
	2.5 Write a program to compare two dictionaries in Python? (By using == operator)
	2.6 Write a program that creates dictionary of cube of odd numbers in the range.
	2.7 Write a program for various list slicing operation.  a= [10,20,30,40,50,60,70,80,90,100]  i. Print Complete list ii. Print 4th element of list iii. Print list from 0th to 4th index. iv. Print list -7th to 3rd element v. Appending an element to list. vi. Sorting the element of list. vii. Popping an element. viii. Removing Specified element. ix. Entering an element at specified index. x. Counting the occurrence of a specified element. xi. Extending list. xii. Reversing the list.
	3.1 Write a program to extend a list in python by using given approach. i. By using + operator. ii. By using Append ()

	iii. By using extend ()
	3.2 Write a program to add two matrices.
	3.3 Write a Python function that takes a list and returns a new list with distinct elements from the first list.
	3.4 Write a program to Check whether a number is perfect or not.
	3.5 Write a Python function that accepts a string and counts the number of upper- and lower-case letters. string_test= 'Today is My Best Day'
	4.1 Write a program to Create Employee Class & add methods to get employee details & print.
	4.2 Write a program to take input as name, email & age from user using combination of keywords argument and positional arguments (*args and **kwargs) using function,
	4.3 Write a program to admit the students in the different Departments(pgdm/btech) and count the students. (Class, Object and Constructor).
	4.4 Write a program that has a class store which keeps the record of code and price of product display the menu of all product and prompt to enter the quantity of each item required and finally generate the bill and display the total amount.
	4.5 Write a program to take input from user for addition of two numbers using (single inheritance).
	4.6 Write a program to create two base classes LU and ITM and one derived class. (Multiple inheritance).
	4.7 Write a program to implement Multilevel inheritance, Grandfather□Father-□Child to show property inheritance from grandfather to child.
	4.8 Write a program Design the Library catalogue system using inheritance take base class (library item) and derived class (Book, DVD & Journal) Each derived

	class should have unique attribute and methods and system should support Check in and check out the system. (Using Inheritance and Method overriding)
	5.1 Write a program to create my_module for addition of two numbers and import it in main script.
	5.2 Write a program to create the Bank Module to perform the operations such as Check the Balance, withdraw and deposit the money in bank account and import the module in main file.
	5.3 Write a program to create a package with name cars and add different modules (such as BMW, AUDI, NISSAN) having classes and functionality and import them in main file cars.
	6.1 Write a program to implement Multithreading. Printing “Hello” with one thread & printing “Hi” with another thread.
	7.1 Write a program to use ‘whether API’ and print temperature of any city, also print the sunrise and sunset times for the same humidity of that area.
	7.2 Write a program to use the ‘API’ of crypto currency.

**Name of Student:** Shikha singh

**Roll Number:** 25

**Experiment No:** 4.4

### **Title:**

4.4 Write a program that has a class store which keeps the record of code and price of product display the menu of all product and prompt to enter the quantity of each item required and finally generate the bill and display the total amount.

## Theory:

- **Store Class:**
  - The Store class is used to represent a store with products and prices.
  - The `__init__` method initializes an empty dictionary to store products and their prices.
  - The `add_product` method adds a product and its price to the store.
  - The `display_menu` method prints the menu of all available products and their prices.
  - The `generate_bill` method takes a dictionary of product quantities, calculates the total cost for each product, and displays the bill along with the total amount.
- **Main Function:**
  - The main function is the entry point of the program.
  - An instance of the Store class, `my_store`, is created.
  - Products and their prices are added to the store using the `add_product` method.
  - The menu is displayed using the `display_menu` method.
  - The user is prompted to enter the quantity of each product they want to purchase. The input is taken until the user enters 'exit'.
  - The entered quantities are used to generate a bill using the `generate_bill` method.

## Code:

```
class Store:
    def __init__(self):
        self.products = {}

    def add_product(self, product_name, price):
        self.products[product_name] = price

    def display_menu(self):
        print("Menu:")
        for product, price in self.products.items():
            print(f'{product} - ${price:.2f}')

    def generate_bill(self, quantities):
        total_amount = 0.0
        print("\nBill:")
        for product, quantity in quantities.items():
            if product in self.products:
                price = self.products[product]
                item_total = price * quantity
                print(f'{product} - {quantity} units - ${item_total:.2f}')
                total_amount += item_total
            else:
                print(f'Product not found: {product}')
        print(f'\nTotal Amount: ${total_amount:.2f}')

def main():

    my_store = Store()
```

```
my_store.add_product("ProductA", 10.99)
my_store.add_product("ProductB", 5.99)
my_store.add_product("ProductC", 7.49)
my_store.display_menu()
quantities = {}
while True:
    product_name = input("Enter the product name (or 'exit' to finish): ")
    if product_name.lower() == 'exit':
        break
    try:
        quantity = int(input("Enter the quantity: "))
        quantities[product_name] = quantity
    except ValueError:
        print("Invalid quantity. Please enter a valid number.")
my_store.generate_bill(quantities)
if __name__ == "__main__":
    main()
```

**Output: (screenshot)**

```
shikhasingh@SHIKHAS-MacBook-Air puthon.py % /usr/bin/python3 /Users/shikhasingh/Desktop/putho
n.py/4.4product.py
Menu:
ProductA - $10.99
ProductB - $5.99
ProductC - $7.49
Enter the product name (or 'exit' to finish): ProductA
Enter the quantity: 3
Enter the product name (or 'exit' to finish): exit

Bill:
ProductA - 3 units - $32.97

Total Amount: $32.97
shikhasingh@SHIKHAS-MacBook-Air puthon.py % %
```

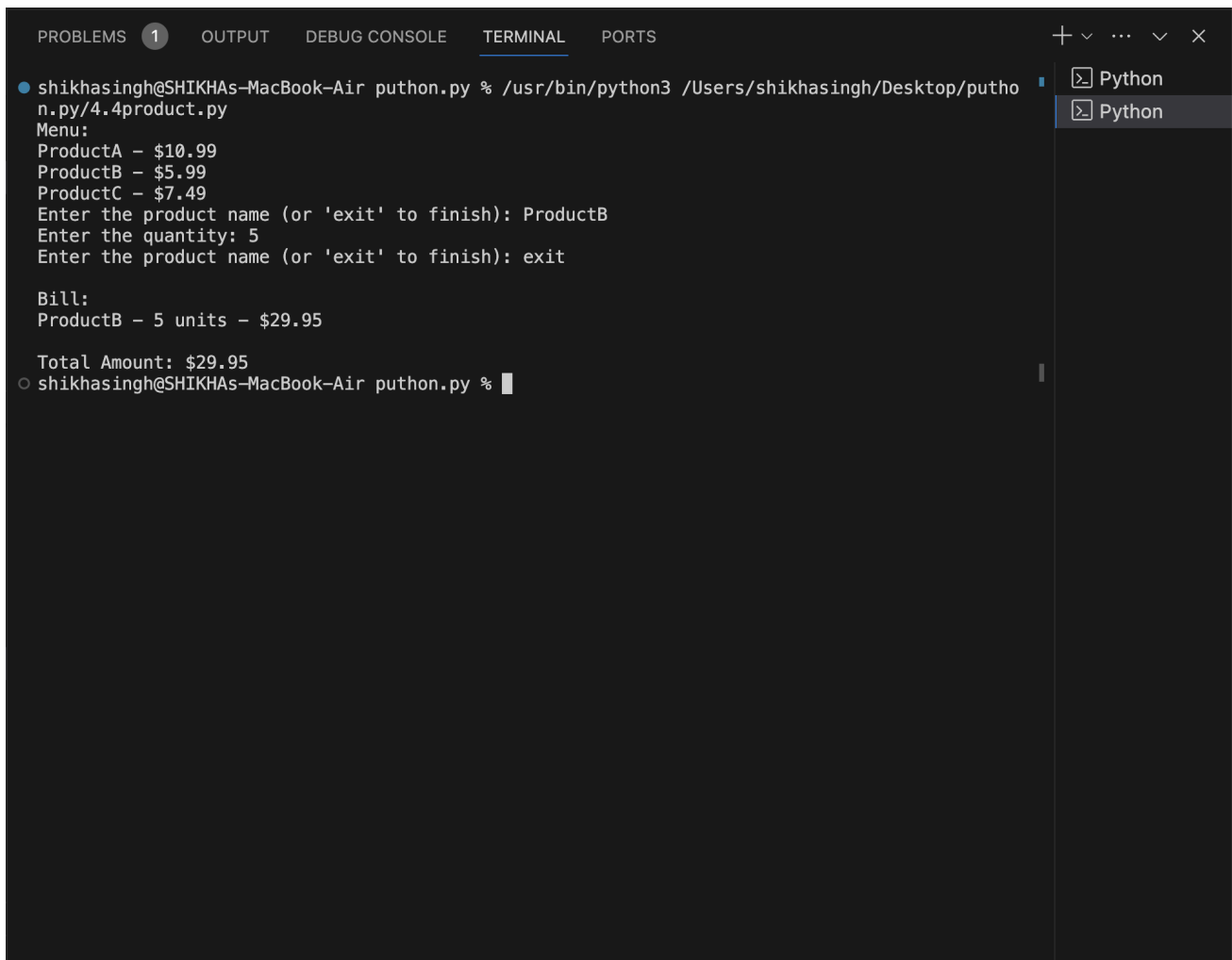
## Test Case: Any two (screenshot)

```
shikhasingh@SHIKHAS-MacBook-Air puthon.py % /usr/bin/python3 /Users/shikhasingh/Desktop/putho
n.py/4.4product.py
Menu:
ProductA - $10.99
ProductB - $5.99
ProductC - $7.49
Enter the product name (or 'exit' to finish): ProductA
Enter the quantity: 3
Enter the product name (or 'exit' to finish): exit

Bill:
ProductA - 3 units - $32.97

Total Amount: $32.97
shikhasingh@SHIKHAS-MacBook-Air puthon.py % %
```



A screenshot of a Visual Studio Code terminal window. The terminal shows the execution of a Python script named 'pthon.py' (likely a typo for 'product.py') located at '/Users/shikhasingh/Desktop/pthon.py'. The program displays a menu with three products: ProductA at \$10.99, ProductB at \$5.99, and ProductC at \$7.49. It prompts the user to enter a product name (or 'exit' to finish). The user enters 'ProductB', then the quantity '5', and finally 'exit'. The program then displays a bill for ProductB: 'ProductB - 5 units - \$29.95'. Finally, it shows the 'Total Amount: \$29.95'. The terminal interface includes tabs for PROBLEMS (1), OUTPUT, DEBUG CONSOLE, TERMINAL (active), and PORTS. On the right side, there are two Python files open in the editor.

```
shikhasingh@SHIKHAS-MacBook-Air pthon.py % /usr/bin/python3 /Users/shikhasingh/Desktop/pthon.py
n.py/4.4product.py
Menu:
ProductA - $10.99
ProductB - $5.99
ProductC - $7.49
Enter the product name (or 'exit' to finish): ProductB
Enter the quantity: 5
Enter the product name (or 'exit' to finish): exit

Bill:
ProductB - 5 units - $29.95

Total Amount: $29.95
shikhasingh@SHIKHAS-MacBook-Air pthon.py %
```

## Conclusion:

This Python program demonstrates the basic principles of object-oriented programming (OOP) by defining a simple class (Store) to model a store's functionality. It also utilizes user input and basic data structures like dictionaries to interact with the program.

The program encourages good programming practices, such as encapsulation, by encapsulating the store-related functionality within a class. It also demonstrates the use of loops (while loop), conditional statements (if statements), and exception handling (try and except) for input validation.

By breaking down the program into modular components (methods within the class), it becomes more readable, maintainable, and extensible. Overall, it serves as a beginner-friendly example of a basic inventory and billing system using Python.