Assignment 3

|  |  |  |
| --- | --- | --- |
| **3.1** | | |
| **Aim:** Write a program to create a basic calculator with functions for addition, subtraction, multiplication, and division. | | |
| **Code:**  ***def*** add(a, b):  *return* a **+** b  ***def*** subtract(a, b):  *return* a **-** b  ***def*** multiply(a, b):  *return* a **\*** b  ***def*** divide(a, b):  *if* b **==** 0:  *return* "Error: Division by zero"  *return* a **/** b  ***def*** main():  *while* True:  print("\nBasic Calculator")  print("1. Add")  print("2. Subtract")  print("3. Multiply")  print("4. Divide")  print("5. Exit")  choice **=** input("Choose an operation (1-5): ")  *if* choice **==** "5":  print("Exiting the calculator. Goodbye!")  *break*  num1 **=** float(input("Enter the first number: "))  num2 **=** float(input("Enter the second number: "))  *if* choice **==** "1":  print("Result:", *add*(num1, num2))  *elif* choice **==** "2":  print("Result:", *subtract*(num1, num2))  *elif* choice **==** "3":  print("Result:", *multiply*(num1, num2))  *elif* choice **==** "4":  print("Result:", *divide*(num1, num2))  *else*:  print("Invalid choice. Please try again.")  *if* \_\_name\_\_ **==** "\_\_main\_\_":  *main*()  **Output Screenshot:** | | |
| **3.2** | | |
| **Aim:** Write a recursive function to calculate the factorial of a number. | | |
| **Code:**  ***def*** factorial(n):  *if* n **==** 0 **or** n **==** 1:  *return* 1  *else*:  *return* n **\*** *factorial*(n **-** 1)  ***def*** main():  *while* True:  print("\nFactorial Calculator")  print("1. Calculate Factorial")  print("2. Exit")  choice **=** input("Choose an option (1-2): ")  *if* choice **==** "2":  print("Exiting the factorial calculator. Goodbye!")  *break*  number **=** int(input("Enter a number to calculate its factorial: "))  print("Factorial of", number, "is:", *factorial*(number))  *if* \_\_name\_\_ **==** "\_\_main\_\_":  *main*()  **Output Screenshot:** | | |
| **3.3** | | |
| **Aim:** Write functions to perform various list operations such as finding the maximum, minimum, sum, and average of a list of numbers. | | |
| **Code:**  ***def*** find\_max(lst):  max\_val **=** lst[0]  *for* num *in* lst:  *if* num **>** max\_val:  max\_val **=** num  *return* max\_val  ***def*** find\_min(lst):  min\_val **=** lst[0]  *for* num *in* lst:  *if* num **<** min\_val:  min\_val **=** num  *return* min\_val  ***def*** find\_sum(lst):  total **=** 0  *for* num *in* lst:  total **+=** num  *return* total  ***def*** find\_average(lst):  *return* *find\_sum*(lst) **/** len(lst)  ***def*** main():  *while* True:  print("\nList Operations")  print("1. Perform Operations")  print("2. Exit")  choice **=** input("Choose an option (1-2): ")  *if* choice **==** "2":  print("Exiting the list operations program. Goodbye!")  *break*  numbers **=** list(map(float, input("Enter a list of numbers separated by spaces: ").*split*()))  print("Max:", *find\_max*(numbers))  print("Min:", *find\_min*(numbers))  print("Sum:", *find\_sum*(numbers))  print("Average:", *find\_average*(numbers))  *if* \_\_name\_\_ **==** "\_\_main\_\_":  *main*()  **Output Screenshot:** | | |
| **3.4** | | |
| **Aim:** Write a function that generates the Fibonacci sequence up to a given number of terms using a for loop. | | |
| **Code:**  ***def*** fibonacci(n):  sequence **=** []  a, b **=** 0, 1  *for* \_ *in* range(n):  sequence.*append*(a)  a, b **=** b, a **+** b  *return* sequence  ***def*** main():  *while* True:  print("\nFibonacci Sequence Generator")  print("1. Generate Sequence")  print("2. Exit")  choice **=** input("Choose an option (1-2): ")  *if* choice **==** "2":  print("Exiting the Fibonacci generator. Goodbye!")  *break*  terms **=** int(input("Enter the number of terms for the Fibonacci sequence: "))  print("Fibonacci sequence:", *fibonacci*(terms))  *if* \_\_name\_\_ **==** "\_\_main\_\_":  *main*()  **Output Screenshot:** | | |
| **3.5** | | |
| **Aim:** Write functions to add, update, and delete key-value pairs in a dictionary, merge two dictionaries, and display the dictionary contents using loops. | | |
| **Code:**  ***def*** add\_key\_value(dictionary, key, value):  dictionary[key] **=** value  ***def*** update\_key\_value(dictionary, key, value):  *if* key **in** dictionary:  dictionary[key] **=** value  *else*:  print("Key not found")  ***def*** delete\_key\_value(dictionary, key):  *if* key **in** dictionary:  *del* dictionary[key]  *else*:  print("Key not found")  ***def*** merge\_dictionaries(dict1, dict2):  *return* {**\*\***dict1, **\*\***dict2}  ***def*** display\_dictionary(dictionary):  *for* key, value *in* dictionary.*items*():  print(***f***"{key}: {value}")  ***def*** main():  my\_dict **=** {}  *while* True:  print("\nDictionary Operations")  print("1. Add Key-Value")  print("2. Update Key-Value")  print("3. Delete Key-Value")  print("4. Merge Dictionaries")  print("5. Display Dictionary")  print("6. Exit")  choice **=** input("Choose an option (1-6): ")  *if* choice **==** "6":  print("Exiting the dictionary operations program. Goodbye!")  *break*  *if* choice **==** "1":  key **=** input("Enter key: ")  value **=** input("Enter value: ")  *add\_key\_value*(my\_dict, key, value)  *elif* choice **==** "2":  key **=** input("Enter key: ")  value **=** input("Enter new value: ")  *update\_key\_value*(my\_dict, key, value)  *elif* choice **==** "3":  key **=** input("Enter key to delete: ")  *delete\_key\_value*(my\_dict, key)  *elif* choice **==** "4":  new\_dict **=** {}  *while* True:  key **=** input("Enter key for new dictionary (or 'done' to finish): ")  *if* key **==** "done":  *break*  value **=** input(***f***"Enter value for {key}: ")  new\_dict[key] **=** value  my\_dict **=** *merge\_dictionaries*(my\_dict, new\_dict)  *elif* choice **==** "5":  *display\_dictionary*(my\_dict)  *else*:  print("Invalid choice. Please try again.")  *if* \_\_name\_\_ **==** "\_\_main\_\_":  *main*()  **Output Screenshot:** | | |
| **3.6** | | |
| **Aim:** Write a program to create a simple to-do list application that allows users to add, remove, and view tasks. | | |
| **Code:**  todo\_list **=** []  ***def*** add\_task(task):  todo\_list.*append*(task)  print(***f***"Task '{task}' added.")  ***def*** remove\_task(task):  *if* task **in** todo\_list:  todo\_list.*remove*(task)  print(***f***"Task '{task}' removed.")  *else*:  print(***f***"Task '{task}' not found.")  ***def*** view\_tasks():  *if* todo\_list:  print("Your To-Do List:")  *for* i, task *in* enumerate(todo\_list, 1):  print(***f***"{i}. {task}")  *else*:  print("Your to-do list is empty.")  ***def*** main():  *while* True:  print("\nTo-Do List Application")  print("1. Add Task")  print("2. Remove Task")  print("3. View Tasks")  print("4. Exit")  choice **=** input("Choose an option (1-4): ")  *if* choice **==** "4":  print("Exiting the to-do list application. Goodbye!")  *break*  *if* choice **==** "1":  task **=** input("Enter the task to add: ")  *add\_task*(task)  *elif* choice **==** "2":  task **=** input("Enter the task to remove: ")  *remove\_task*(task)  *elif* choice **==** "3":  *view\_tasks*()  *else*:  print("Invalid choice. Please try again.")  *if* \_\_name\_\_ **==** "\_\_main\_\_":  *main*()  **Output Screenshot:** | | |
| **3.7** | | |
| **Aim:** Write a program that accepts a list of numbers and returns a new list containing only the even numbers. | | |
| **Code:**  ***def*** filter\_even\_numbers(numbers):  *return* [num *for* num *in* numbers *if* num **%** 2 **==** 0]  ***def*** main():  *while* True:  print("\nEven Number Filter")  print("1. Filter Even Numbers")  print("2. Exit")  choice **=** input("Choose an option (1-2): ")  *if* choice **==** "2":  print("Exiting the even number filter. Goodbye!")  *break*  numbers **=** list(map(int, input("Enter a list of numbers separated by spaces: ").*split*()))  print("Even numbers:", *filter\_even\_numbers*(numbers))  *if* \_\_name\_\_ **==** "\_\_main\_\_":  *main*()  **Output Screenshot:** | | |
| **3.8** | | |
| **Aim:** Write a program that finds the largest and smallest numbers in a list without using built-in functions like max() and min() | | |
| **Code:**  ***def*** find\_largest\_smallest(numbers):  *if* **not** numbers:  *return* None, None  largest **=** smallest **=** numbers[0]  *for* num *in* numbers:  *if* num **>** largest:  largest **=** num  *if* num **<** smallest:  smallest **=** num  *return* largest, smallest  ***def*** main():  *while* True:  print("\nLargest and Smallest Number Finder")  print("1. Find Largest and Smallest")  print("2. Exit")  choice **=** input("Choose an option (1-2): ")  *if* choice **==** "2":  print("Exiting the program. Goodbye!")  *break*  numbers **=** list(map(float, input("Enter a list of numbers separated by spaces: ").*split*()))  largest, smallest **=** *find\_largest\_smallest*(numbers)  print("Largest:", largest)  print("Smallest:", smallest)  *if* \_\_name\_\_ **==** "\_\_main\_\_":  *main*()  **Output Screenshot:** | | |
| **Conclusion/Summary:**  This assignment provided a comprehensive exploration of fundamental Python programming concepts, including functions, user input, loops, conditionals, and data structures like lists and dictionaries. By implementing a variety of programs—such as a basic calculator, factorial calculator, list operations, Fibonacci sequence generator, dictionary operations, to-do list application, even number filter, and largest/smallest number finder—we gained hands-on experience in solving real-world problems using Python. Each program was designed to be user-friendly, with clear prompts and an option to exit, ensuring a smooth and interactive experience. This assignment not only reinforced core programming principles but also demonstrated the versatility and power of Python in building practical applications. Moving forward, these foundational skills will serve as a strong base for tackling more complex programming challenges. | | |
| **Student Signature & Date** | **Marks:** | **Evaluator Signature & Date** |