1.simple calculator Using Python.

```
  def add(a, b):
    return a + b


def sub(a, b):
    return a - b


def mul(a, b):
    return a * b

def div(a, b):
    if b == 0:
        return "Division by zero not possible"
    else:
        return a / b


def menudriven():
    print("Select operation:")
    print("1. Add")
    print("2. Subtract")
    print("3. Multiply")
    print("4. Divide")
    print("5. Exit")
    choice = input("Enter choice (1/2/3/4/5): ")
    return choice

def calculator():
    while True:
        choice = menudriven()

        if choice in ('1', '2', '3', '4'):
            num1 = float(input("Enter first number: "))
            num2 = float(input("Enter second number: "))

            if choice == '1':
                print(f"{num1} + {num2} = {add(num1, num2)}")
            elif choice == '2':
                print(f"{num1} - {num2} = {sub(num1, num2)}")
            elif choice == '3':
                print(f"{num1} * {num2} = {mul(num1, num2)}")
            elif choice == '4':
                print(f"{num1} / {num2} = {div(num1, num2)}")

        elif choice == '5':
            print("Exiting the calculator.")
            break

        else:
```

```
            print("Invalid input.")

calculator()
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR  DEVDB
PS D:\Rahul Python> & "C:/Program Files/Python311/python.exe" "d:/Rahul Python/calc.py"
Select operation:
1. Add
2. Subtract
3. Multiply
4. Divide
5. Exit
Enter choice (1/2/3/4/5): 1
Enter first number: 4
Enter second number: 7
4.0 + 7.0 = 11.0
Select operation:
1. Add
2. Subtract
3. Multiply
4. Divide
5. Exit
Enter choice (1/2/3/4/5):
```

2. An electric power distribution company charges domestic customers as

follows: Consumption unit Rate of charge:

1.2.1. 0-200 Rs. 0.50 per unit

1.2.2. 201-400 Rs. 0.65 per unit in excess of 200

1.2.3. 401-600 Rs 0.80 per unit excess of 400

1.2.4. 601 and above Rs 1.00per unit excess of 600

1.2.5. If the bill exceeds Rs. 400, then a surcharge of 15% will be charged, and the minimum bill should be Rs. 100/-

Create a Python program based on the scenario mentioned above.

```
def amount(units):
    if units <= 200:
        bill = units*0.50
    elif units <= 400:
        bill =(200*0.50)+(units-200*0.65)
    elif units <= 600:
        bill =(200*0.50)+(200*0.65)+(units-400)*0.80
    else:
        bill =(200*0.50)+(200*0.65)+(200*0.80)+(units-600)*1.00

    if bill>400:
```

```
        bill+= bill*0.15

    if bill<100:
        bill=100

    return bill

def main():
    units = float(input("Enter the number of units consumed: "))
    bill = amount(units)
    print("The total bill is: Rs.", bill)

main()
```

output:

```
PS D:\Rahul Python> & "C:/Program Files/Python311/python.exe" "d:/Rahul Python/bill.py"
Enter the number of units consumed: 150
The total bill is: Rs. 100
PS D:\Rahul Python>
```

3. Print the pyramid of numbers using for loops.

```
n = 5
for i in range(1, n + 1):

    for j in range(n - i):
        print(" ", end="")


    for j in range(1, i + 1):
        print(j, end="")


    for j in range(i - 1, 0, -1):
        print(j, end="")

    print()
```

output:

```
PS D:\Rahul Python> & "C:/Program Files/Python311/python.exe" "d:/Rahul Python/pyramid.py"
    1
   121
  12321
 1234321
123454321
```

4. Write a program to find the number and sum of all integers greater than 100

and less than 200 that are divisible by 7.

```python
def fs():
    numbers = []
    for i in range(101, 200):
        if i % 7 == 0:
            numbers.append(i)

    count = len(numbers)
    total_sum = sum(numbers)

    return count, total_sum

count, total_sum = fs()
print(f"Count of integers divisible by 7 between 101 and 199: {count}")
print(f"Sum of integers divisible by 7 between 101 and 199: {total_sum}")

```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/AppData/Local/Micro
Count of integers divisible by 7 between 101 and 199: 14
Sum of integers divisible by 7 between 101 and 199: 2107
```

5. Write a recursive function to calculate the sum of numbers from 0 to 10.

recursive.py > sum_recursive

```python
def sum_recursive(n):
    if n == 0:
        return 0
    else:
        return n + sum_recursive(n - 1)
result = sum_recursive(10)
print(f"The sum of numbers from 0 to 10 is: {result}")

```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/A
The sum of numbers from 0 to 10 is: 55
PS D:\Rajagiri\python>
```

6. Write a Python program to reverse the digits of a given number and add them to the original. If the sum is not a palindrome, repeat this procedure

palindrome.py > ...

```python
def is_palindrome(n):
    return str(n) == str(n)[::-1]

def reverse_number(n):
    return int(str(n)[::-1])

def find_palindrome_sum(n):
    steps = 0
    while not is_palindrome(n):
        reversed_n = reverse_number(n)
        n += reversed_n
        steps += 1
        print(f"Step {steps}: {n} (Reversed: {reversed_n})")
    return n, steps

number = int(input("Enter a number: "))
palindrome_sum, total_steps = find_palindrome_sum(number)
print(f"\nPalindrome sum: {palindrome_sum} (Total steps: {total_steps})")
```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/AppData/Local/Microsoft/WindowsAp
Enter a number: 123
Step 1: 444 (Reversed: 321)

Palindrome sum: 444 (Total steps: 1)
PS D:\Rajagiri\python> 121
121
```

7. Write a menu-driven program that performs the following operations on strings

7.1. Check if the String is a Substring of Another String

7.2. Count Occurrences of Character

7.3. Replace a substring with another substring

7.4. Convert to Capital Letters

capital.py > ...

```python
def check_substring(main_string, sub_string):
    return sub_string in main_string

def count_occurrences(main_string, char):
    return main_string.count(char)

def replace_substring(main_string, old_sub, new_sub):
    return main_string.replace(old_sub, new_sub)

def convert_to_capital(main_string):
    return main_string.upper()

def display_menu():
    print("Menu:")
    print("1. Check if the String is a Substring of Another String")
    print("2. Count Occurrences of Character")
    print("3. Replace a Substring with Another Substring")
    print("4. Convert to Capital Letters")
    print("5. Exit")
```

```python
def main():
    while True:
        display_menu()
        choice = input("Enter your choice (1-5): ")

        if choice == '1':
            main_string = input("Enter the main string: ")
            sub_string = input("Enter the substring to check: ")
            result = check_substring(main_string, sub_string)
            print(f"'{sub_string}' is {'a' if result else 'not a'} substring of '{main_string}'")

        elif choice == '2':
            main_string = input("Enter the string: ")
            char = input("Enter the character to count: ")
            result = count_occurrences(main_string, char)
            print(f"The character '{char}' occurs {result} times in the string.")

        elif choice == '3':
            main_string = input("Enter the main string: ")
            old_sub = input("Enter the substring to replace: ")
            new_sub = input("Enter the new substring: ")
            result = replace_substring(main_string, old_sub, new_sub)
            print(f"The new string is: '{result}'")
```

```
        elif choice == '4':
            main_string = input("Enter the string: ")
            result = convert_to_capital(main_string)
            print(f"The string in capital letters is: '{result}'")

        elif choice == '5':
            print("Exiting the program.")
            break

        else:
            print("Invalid choice. Please try again.")

if __name__ == "__main__":
    main()
```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/A
Menu:
1. Check if the String is a Substring of Another Strir
2. Count Occurrences of Character
3. Replace a Substring with Another Substring
4. Convert to Capital Letters
5. Exit
Enter your choice (1-5): 1
Enter the main string: colour
Enter the substring to check: our
'our' is a substring of 'colour'
```

```
Menu:
1. Check if the String is a Substring of Another String
2. Count Occurrences of Character
3. Replace a Substring with Another Substring
4. Convert to Capital Letters
5. Exit
Enter your choice (1-5): 2
Enter the string: cherry
Enter the character to count: r
The character 'r' occurs 2 times in the string.
```

```
1. Check if the String is a Substring of Another String
2. Count Occurrences of Character
3. Replace a Substring with Another Substring
4. Convert to Capital Letters
5. Exit
Enter your choice (1-5): 3
Enter the main string: walking
Enter the substring to replace: ing
Enter the new substring: ed
The new string is: 'walked'
```

```
1. Check if the String is a Substring of Another String
2. Count Occurrences of Character
3. Replace a Substring with Another Substring
4. Convert to Capital Letters
5. Exit
Enter your choice (1-5): 4
Enter the string: done
The string in capital letters is: 'DONE'
```

8. Write a function to find the factorial of a number but also store the factorials calculated in a dictionary

factorialdic.py > ...

```python
factorial_dic = {}
def factorial(n):
    if n in factorial_dic:
        return factorial_dic[n]

    if n == 0 or n == 1:
        factorial_dic[n] = 1
    else:
        factorial_dic[n] = n * factorial(n - 1)

    return factorial_dic[n]
number = 5
result = factorial(number)
print(f"Factorial of {number} is {result}")
print(f" factorial are: {factorial_dic}")

```

Output:

```
PS D:\Rajagiri\python> & C:/Users/Rahul K Ravindran/AppData/Local/
Factorial of 5 is 120
 factorial are: {1: 1, 2: 2, 3: 6, 4: 24, 5: 120}
PS D:\Rajagiri\python>
```

9. Perform various set operations

9.1. Set Union

9.2. Set Intersection

9.3. Set Difference

set operation.py > set_union

```python
def set_union(set1, set2):
    return set1.union(set2)

def set_intersection(set1, set2):
    return set1.intersection(set2)

def set_difference(set1, set2):
    return set1.difference(set2)

def display_menu():
    print("Menu:")
    print("1. Set Union")
    print("2. Set Intersection")
    print("3. Set Difference")
    print("4. Exit")

def main():
    while True:
        display_menu()
        choice = input("Enter your choice (1-4): ")
        if choice in ['1', '2', '3']:
            set1 = set(input("Enter elements of the first set separated by space: ").split())
            set2 = set(input("Enter elements of the second set separated by space: ").split())

            if choice == '1':
                result = set_union(set1, set2)
                print(f"Union of the sets: {result}")

            elif choice == '2':
                result = set_intersection(set1, set2)
                print(f"Intersection of the sets: {result}")

            elif choice == '3':
                result = set_difference(set1, set2)
                print(f"Difference of the sets: {result}")

            elif choice == '4':
                print("Exiting the program.")
                break
            else:
                print("Invalid choice. Please try again.")

if __name__ == "__main__":
    main()
```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/AppData/Local/Micr
Menu:
1. Set Union
2. Set Intersection
3. Set Difference
4. Exit
Enter your choice (1-4): 1
Enter elements of the first set separated by space: 1 2 3 4 5
Enter elements of the second set separated by space:  2 3 5
Union of the sets: {'4', '5', '1', '3', '2'}
```

```
Menu:
1. Set Union
2. Set Intersection
3. Set Difference
4. Exit
Enter your choice (1-4): 2
Enter elements of the first set separated by space: 3 4 5
Enter elements of the second set separated by space: 2 5
Intersection of the sets: {'5'}
```

```
Menu:
1. Set Union
2. Set Intersection
3. Set Difference
4. Exit
Enter your choice (1-4): 3
Enter elements of the first set separated by space:  4 5 6
Enter elements of the second set separated by space: 1 2 4
Difference of the sets: {'5', '6'}
```

10. Create a dictionary to store the name, roll_no, and total_mark of N students. Now print the details of the student with the highest total_mark.

stud.py > main

```python
def stud():
    students = {}
    N = int(input("Enter the number of students: "))

    for _ in range(N):
        roll_no = input("Enter roll number: ")
        name = input("Enter name: ")
        total_mark = int(input("Enter total mark: "))
        students[roll_no] = {
            "name": name,
            "total_mark": total_mark
        }

    return students

def top(students):
    top_student = None
    highest_mark = -1

    for roll_no, details in students.items():
        if details["total_mark"] > highest_mark:
            highest_mark = details["total_mark"]
            top_student = (roll_no, details["name"], details["total_mark"])
    return top_student

def main():
    students = stud()
    top_student = top(students)

    if top_student:
        roll_no, name, total_mark = top_student
        print("\nDetails of the student with the highest total mark:")
        print(f"Roll Number: {roll_no}")
        print(f"Name: {name}")
        print(f"Total Mark: {total_mark}")
    else:
        print("No students found.")

if __name__ == "__main__":
    main()
```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/AppData/Local/Microsoft/WindowsApps/python3.11.exe" c
Enter the number of students: 3
Enter roll number: 1
Enter name: Rahul
Enter total mark: 45
Enter roll number: 2
Enter name: Amal
Enter total mark: 33
Enter roll number: 3
Enter name: Sarath
Enter total mark: 40

Details of the student with the highest total mark:
Roll Number: 1
Name: Rahul
Total Mark: 45
PS D:\Rajagiri\python>
```

11. Write a Python program to copy the contents of a file into another file, line by line.

fileop.py > ...

```
def copy_file(source_file, destination_file):
                with open(destination_file, 'w') as dest:
                    for line in src:
                        dest.write(line)
            print(f"Contents copied from {source_file} to {destination_file} successfully.")
        except FileNotFoundError:
            print(f"The file {source_file} does not exist.")
        except IOError as e:
            print(f"An error occurred while copying the file: {e}")

# Example usage
source_file = input("Enter the source file path: ")
destination_file = input("Enter the destination file path: ")

copy_file(source_file, destination_file)

```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/AppData/L
Enter the source file path: stud.py
Enter the destination file path: copy.py
Contents copied from stud.py to copy.py successfully.
PS D:\Rajagiri\python>
```

12. Use the OS module to perform

12.1. Create a directory

12.2. Directory Listing

12.3. Search for ".py" files

12.4. Remove a particular file

Creating Directory

osmodule.py > ...

```python
import os

def create_directory(directory_name):
    try:
        os.mkdir(directory_name)
        print(f"Directory '{directory_name}' created successfully.")
    except FileExistsError:
        print(f"Directory '{directory_name}' already exists.")
    except Exception as e:
        print(f"Error occurred while creating directory '{directory_name}': {e}")

def main():
    directory_name = input("Enter the directory name to create: ")
    create_directory(directory_name)

if __name__ == "__main__":
    main()

```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/AppData/Local
Enter the directory name to create: newpgm.py
Directory 'newpgm.py' created successfully.
PS D:\Rajagiri\python>
```

Directory listing

dirlisting.py > ...

```python
import os

def directory_listing(directory_path):
    try:
        print(f"Listing files and directories in '{directory_path}':")
        for item in os.listdir(directory_path):
            print(item)
    except FileNotFoundError:
        print(f"Directory '{directory_path}' not found.")
    except Exception as e:
        print(f"Error occurred while listing directory '{directory_path}': {e}")

def main():
    directory_path = input("Enter the directory path to list: ")
    directory_listing(directory_path)

if __name__ == "__main__":
    main()

```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/AppData/Loca
Enter the directory path to list: newpgm.py
Listing files and directories in 'newpgm.py':
PS D:\Rajagiri\python>
```

Search for py files

searchdir.py > ...

```python
import os

def search_py_files(directory_path):
    try:
        print(f"Searching for '.py' files in '{directory_path}':")
        for root, dirs, files in os.walk(directory_path):
            for file in files:
                if file.endswith(".py"):
                    print(os.path.join(root, file))
    except FileNotFoundError:
        print(f"Directory '{directory_path}' not found.")
    except Exception as e:
        print(f"Error occurred while searching for '.py' files in '{directory_path}': {e}")

def main():
    directory_path = input("Enter the directory path to search for '.py' files: ")
    search_py_files(directory_path)

if __name__ == "__main__":
    main()
```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/AppData/Local/Micr
Enter the directory path to search for '.py' files: newpgm.py
Searching for '.py' files in 'newpgm.py':
PS D:\Rajagiri\python>
```

Remove a particular file

removedir.py > ...

```python
import os

def remove_file(file_path):
    try:
        # Check file attributes
        print(f"File attributes for '{file_path}':")
        os.system(f'attrib "{file_path}"')

        # Try to remove the file
        os.remove(file_path)
        print(f"File '{file_path}' removed successfully.")
    except FileNotFoundError:
        print(f"File '{file_path}' not found.")
    except Exception as e:
        print(f"Error occurred while removing file '{file_path}': {e}")

def main():
    file_path = input("Enter the file path to remove: ")
    remove_file(file_path)

if __name__ == "__main__":
    main()
```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/AppData/Local/Microsoft/
Enter the file path to remove: D:\Rajagiri\python\newpgm.py
File attributes for 'D:\Rajagiri\python\newpgm.py':
                    D:\Rajagiri\python\newpgm.py
```

13. Create a simple banking application by using inheritance.

bankacc.py > main

```python
class BankAccount:
    def __init__(self, account_number, balance=0):
        self.account_number = account_number
        self.balance = balance

    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            print(f"Deposited {amount} into account {self.account_number}. New balance: {self.balance}")
        else:
            print("Invalid amount. Deposit failed.")

    def withdraw(self, amount):
        if 0 < amount <= self.balance:
            self.balance -= amount
            print(f"Withdrew {amount} from account {self.account_number}. New balance: {self.balance}")
        else:
            print("Insufficient funds. Withdrawal failed.")
```

```python
    def display_balance(self):
        print(f"Account {self.account_number} balance: {self.balance}")


class SavingsAccount(BankAccount):
    def __init__(self, account_number, balance=0, interest_rate=0.01):
        super().__init__(account_number, balance)
        self.interest_rate = interest_rate

    def add_interest(self):
        interest_amount = self.balance * self.interest_rate
        self.balance += interest_amount
        print(f"Interest added to account {self.account_number}. New balance: {self.balance}")


class CurrentAccount(BankAccount):
    def __init__(self, account_number, balance=0, overdraft_limit=1000):
        super().__init__(account_number, balance)
        self.overdraft_limit = overdraft_limit
```

```python
    def withdraw(self, amount):
        if 0 < amount <= self.balance + self.overdraft_limit:
            self.balance -= amount
            print(f"Withdrew {amount} from account {self.account_number}. New balance: {self.balance}")
        else:
            print("Insufficient funds. Withdrawal failed.")

    def display_balance(self):
        print(f"Current Account {self.account_number} balance: {self.balance}")

def main():
    savings_acc = SavingsAccount("RAHUL", 5000, 0.02)
    savings_acc.display_balance()
    savings_acc.deposit(2000)
    savings_acc.add_interest()
    savings_acc.withdraw(1500)
    savings_acc.display_balance()
```

```python
    current_acc = CurrentAccount("CURACC", 10000, 2000)
    current_acc.display_balance()
    current_acc.deposit(3000)
    current_acc.withdraw(12000)
    current_acc.display_balance()

if __name__ == "__main__":
    main()
```

Output:

```
PS D:\Rajagiri\python> & "C:/Users/Rahul K Ravindran/AppData/Local/Microsof
Account RAHUL balance: 5000
Deposited 2000 into account RAHUL. New balance: 7000
Interest added to account RAHUL. New balance: 7140.0
Withdrew 1500 from account RAHUL. New balance: 5640.0
Account RAHUL balance: 5640.0
Current Account CURACC balance: 10000
Deposited 3000 into account CURACC. New balance: 13000
Withdrew 12000 from account CURACC. New balance: 1000
Current Account CURACC balance: 1000
```