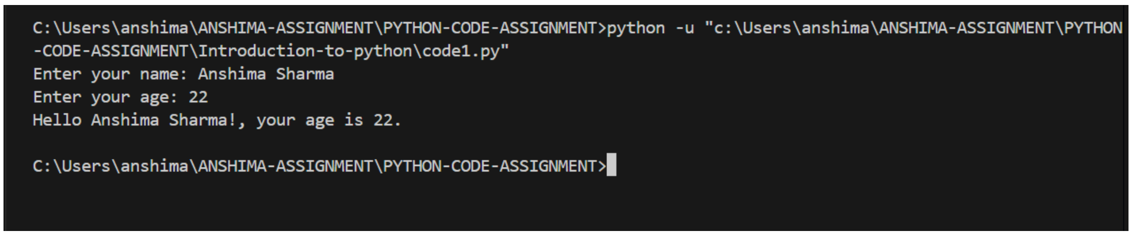
PYTHON CODING ASSIGNMENT

INTRODUCTION TO PYTHON

1. Write a Python script that accepts user input for name and age and prints a greeting message.

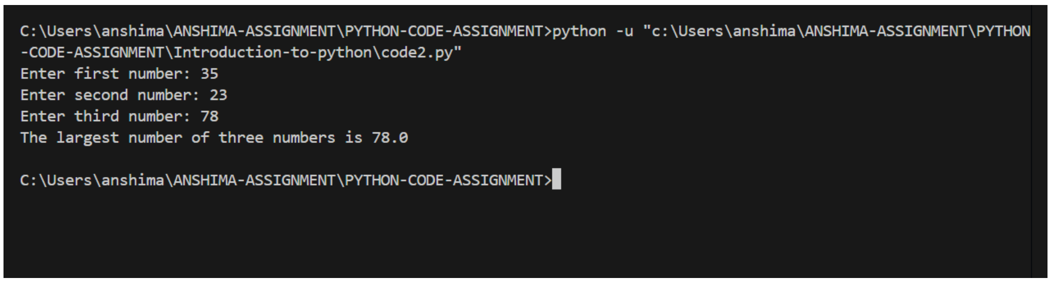
```
name = input("Enter your name: ")
age = input("Enter your age: ")
print(f"Hello {name}!, your age is {age}.")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT\Introduction-to-python\code1.py"
Enter your name: Anshima Sharma
Enter your age: 22
Hello Anshima Sharma!, your age is 22.

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

2. Create a program to find the largest of three input numbers using conditional statements.

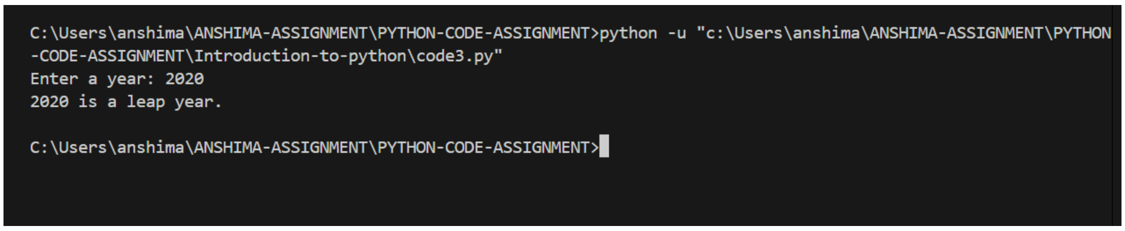
```
a = float(input("Enter first number: "))
b = float(input("Enter second number: "))
c = float(input("Enter third number: "))

if a >= b and a >= c:
        largest = a
elif b >= a and b >= c:
        largest = b
else:
        largest = c

print(f"The largest number of three numbers is {largest}")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT\Introduction-to-python\code2.py"
Enter first number: 35
Enter second number: 23
Enter third number: 78
The largest number of three numbers is 78.0

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

3. Write a script to check if a given year is a leap year.

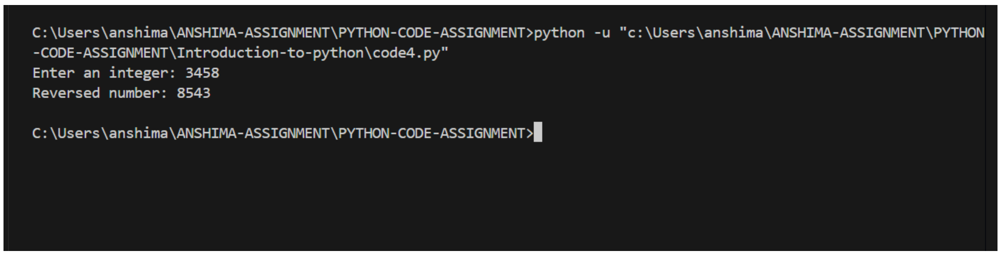
```
year = int(input("Enter a year: "))
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
            print(f"{year} is a leap year.")
else:
           print(f"{year} is not a leap year.")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Introduction-to-python\code3.py"
Enter a year: 2020
2020 is a leap year.

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

4. Develop a Python program that reverses a given integer.

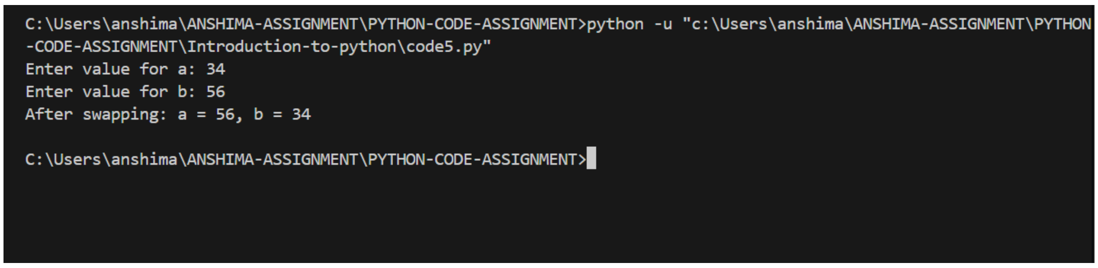
```
number = int(input("Enter an integer: "))
rev_num = 0
original = abs(number)
while original > 0:
       d = original % 10
       rev_num = rev_num * 10 + d
       original //= 10

if number < 0:
      rev_num = -rev_num

print(f"Reversed number: {rev_num}")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Introduction-to-python\code4.py"
Enter an integer: 3458
Reversed number: 8543

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

5. Write a script that swaps two variables without using a third variable.

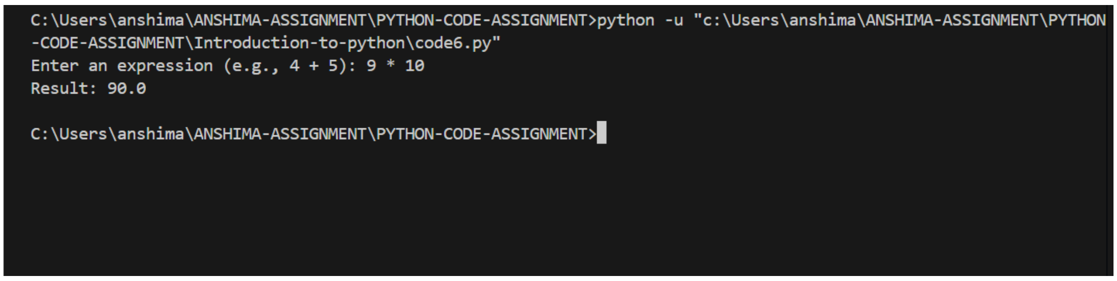
```
a = int(input("Enter value for a: "))
b = int(input("Enter value for b: "))
a, b = b, a
print(f"After swapping: a = {a}, b = {b}")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Introduction-to-python\code5.py"
Enter value for a: 34
Enter value for b: 56
After swapping: a = 56, b = 34

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

6. Create a program that simulates a simple calculator supporting +, -, *, / with input parsing.

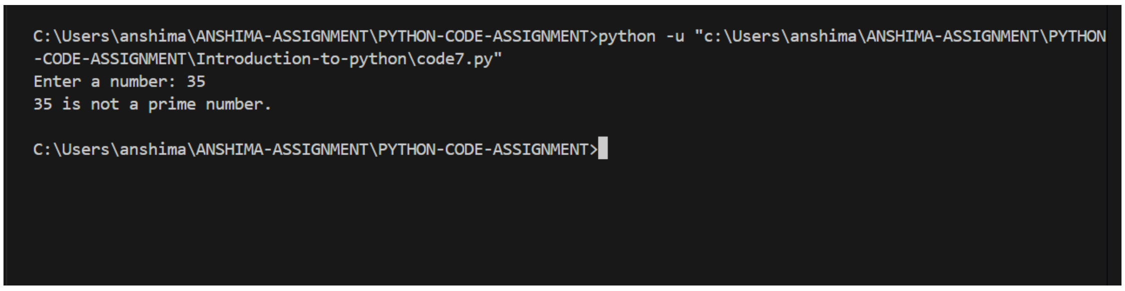
```
expr = input("Enter an expression (e.g., 4 + 5): ")
 tokens = expr.split()
 if len(tokens) == 3:
      a, op, b = tokens
      a = float(a)
      b = float(b)
 if op == '+':
       result = a + b
 elif op == '-':
       result = a - b
elif op == '*':
      result = a * b
elif op == '/':
       if b != 0:
               result = a / b
      else:
              result = "Cannot divide by zero"
      else:
             result = "Invalid operator"
      else:
             result = "Invalid input format"
print("Result:", result)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Introduction-to-python\code6.py"
Enter an expression (e.g., 4 + 5): 9 * 10
Result: 90.0

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

7. Write a Python script to determine if a given number is a prime number.

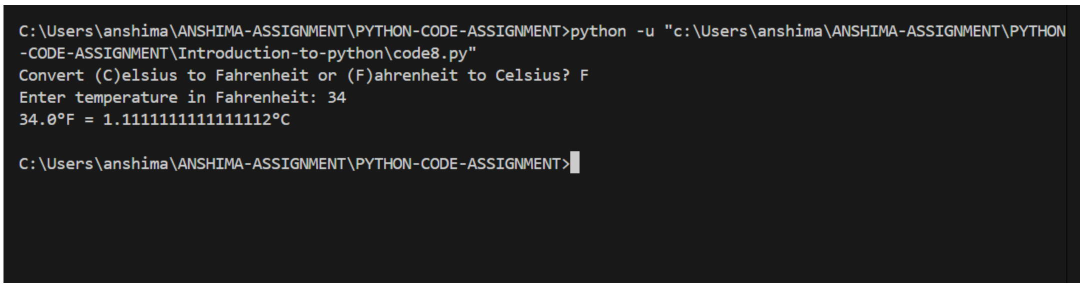
```
number = int(input("Enter a number: "))
if number <= 1:
     print(f"{number} is not a prime number.")
else:
    for i in range(2, int(number**0.5) + 1):
            if number % i == 0:
                 print(f"{number} is not a prime number.")
                 break
            else:
                 print(f"{number} is a prime number.")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Introduction-to-python\code7.py"
Enter a number: 35
35 is not a prime number.

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

8. Develop a program to convert a given temperature from Celsius to Fahrenheit and vice versa.

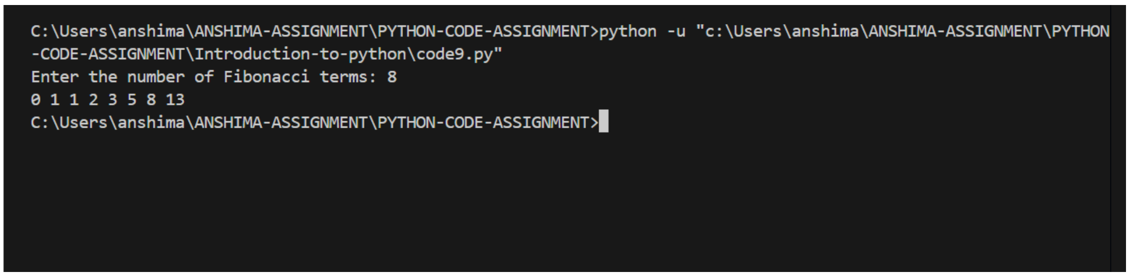
```
ch = input("Convert (C)elsius to Fahrenheit or (F)ahrenheit to Celsius? ").strip().upper()
if ch == 'C':
    c = float(input("Enter temperature in Celsius: "))
    f = (c * 9/5) + 32
    print(f"{c}°C = {f}°F")
elif ch == 'F':
    f = float(input("Enter temperature in Fahrenheit: "))
    c = (f - 32) * 5/9
    print(f"{f}°F = {c}°C")
else:
    print("Enter correct choice!")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Introduction-to-python\code8.py"
Convert (C)elsius to Fahrenheit or (F)ahrenheit to Celsius? F
Enter temperature in Fahrenheit: 34
34.0°F = 1.1111111111111112°C

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

9. Create a Python program that prints the Fibonacci sequence up to n terms using iteration.

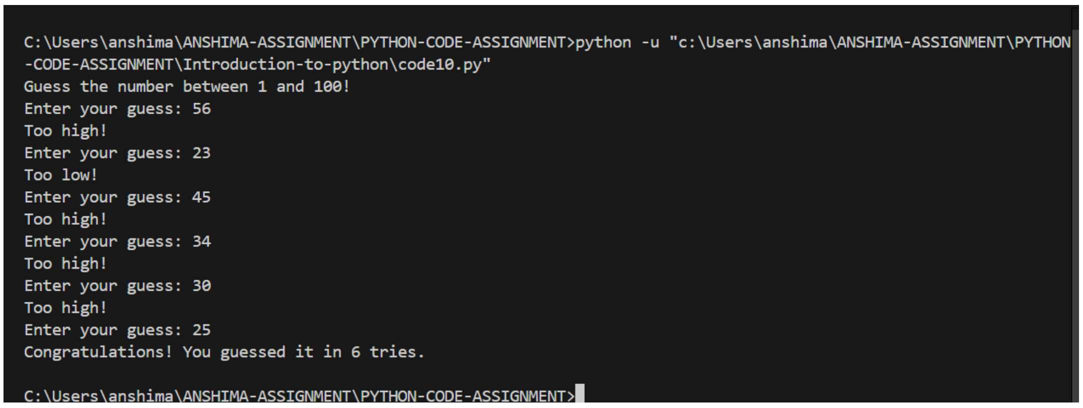
```
number = int(input("Enter the number of Fibonacci terms: "))
a, b = 0, 1
count = 0
while count < number:
    print(a, end=" ")
    a, b = b, a + b
    count += 1
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Introduction-to-python\code9.py"
Enter the number of Fibonacci terms: 8
0 1 1 2 3 5 8 13
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

10. Implement a basic number guessing game where the computer selects a random number.

```
import random
secret = random.randint(1, 100)
attempts = 0
print("Guess the number between 1 and 100!")
while True:
    guess = int(input("Enter your guess: "))
    attempts += 1
    if guess < secret:
        print("Too low!")
    elif guess > secret:
        print("Too high!")
    else:
        print(f"Congratulations! You guessed it in {attempts} tries.")
        break
```

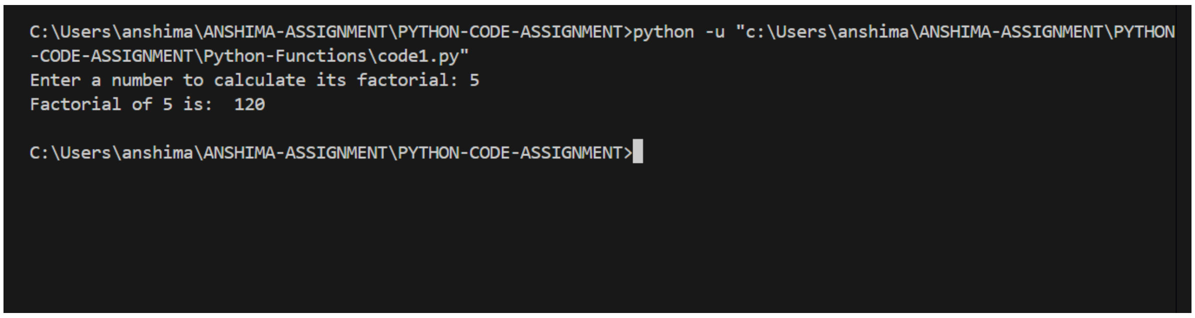


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Introduction-to-python\code10.py"
Guess the number between 1 and 100!
Enter your guess: 56
Too high!
Enter your guess: 23
Too low!
Enter your guess: 45
Too high!
Enter your guess: 34
Too high!
Enter your guess: 30
Too high!
Enter your guess: 25
Congratulations! You guessed it in 6 tries.

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

PYTHON FUNCTIONS

1. Write a function to calculate the factorial of a number (non-recursive).

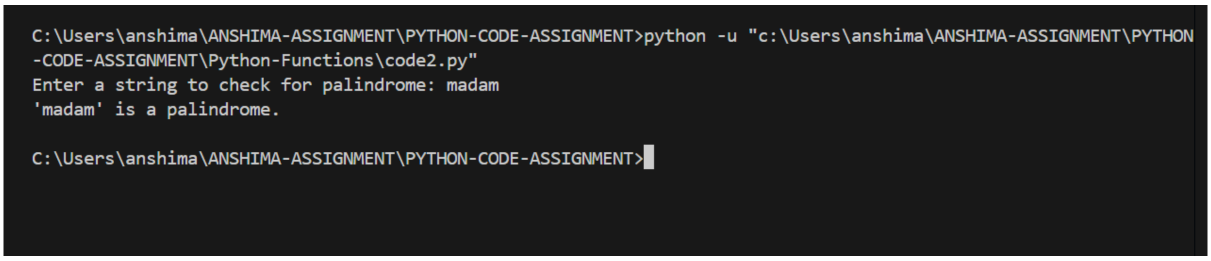
```
def factorial(n):
    result=1
    for i in range(2,n+1):
        result*=1
    return result
val=int(input("Enter a number to calculate its factorial: ")
print(f'Factorial of {val} is: ",factorial(val))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Python-Functions\code1.py"
Enter a number to calculate its factorial: 5
Factorial of 5 is:  120

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

2. Define a function that checks whether a string is a palindrome.

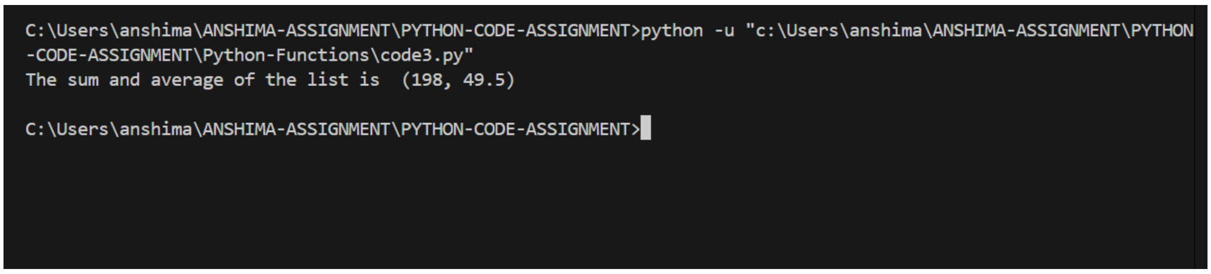
```
def sum_and_average(lst):
     total = sum(lst)
     avg = total / len(lst) if lst else 0
    return total, avg

list=[54,21,34,89]
print("The sum and average of the list is ",sum_and_average(list))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Python-Functions\code2.py"
Enter a string to check for palindrome: madam
'madam' is a palindrome.

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

3. Write a function that accepts a list and returns the sum and average of the numbers.

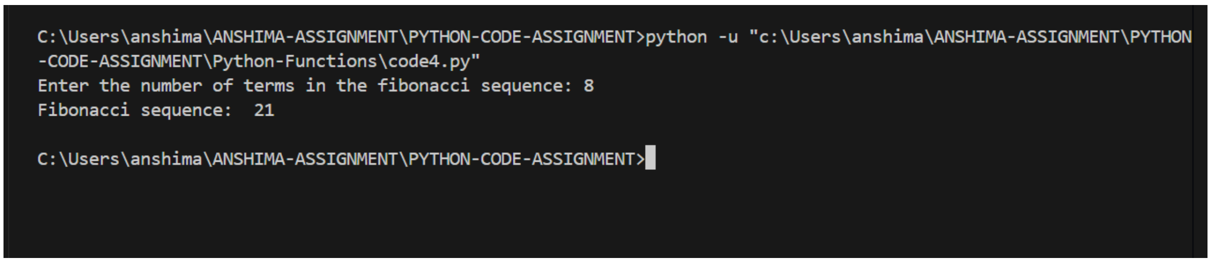
```
def fibonacci(n):
   if n <= 1:
      return n
   return fibonacci(n - 1) + fibonacci(n - 2)

val=int(input("Enter the number of terms in the fibonacci sequence: "))
print("Fibonacci sequence: ",fibonacci(val))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Python-Functions\code3.py"
The sum and average of the list is  (198, 49.5)

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

4. Create a function that returns the nth Fibonacci number using recursion.

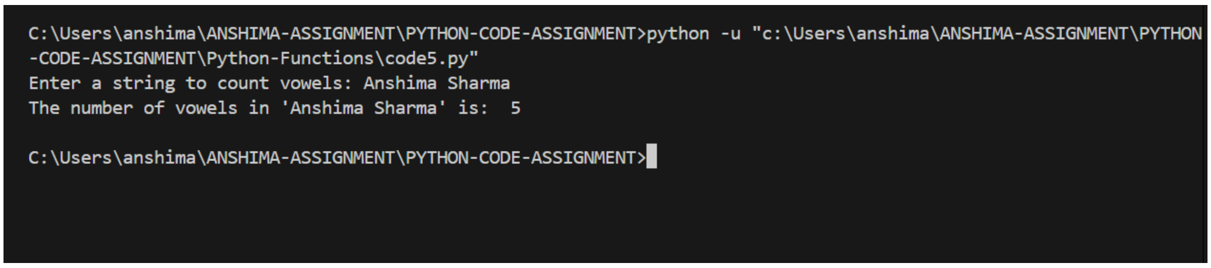
```
def count_vowels(s):
    vowels = 'aeiouAEIOU'
    return sum(1 for char in s if char in vowels)

str=input("Enter a string to count vowels: ")
print(f"The number of vowels in '{str}' is: ",count_vowels(str))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Python-Functions\code4.py"
Enter the number of terms in the fibonacci sequence: 8
Fibonacci sequence:  21

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

5. Define a function to count the number of vowels in a given string.

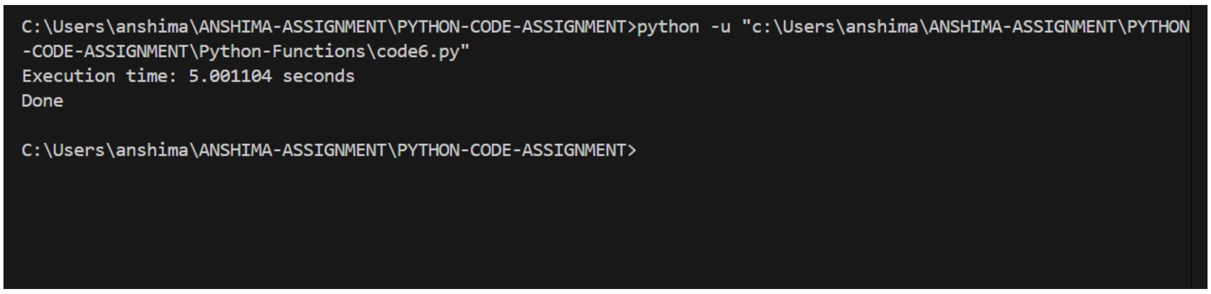
```
def count_vowels(s):
    vowels = 'aeiouAEIOU'
    return sum(1 for char in s if char in vowels)

str=input("Enter a string to count vowels: ")
print(f"The number of vowels in '{str}' is: ",count_vowels(str))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Python-Functions\code5.py"
Enter a string to count vowels: Anshima Sharma
The number of vowels in 'Anshima Sharma' is:  5

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

6. Implement a decorator that measures execution time of any function.

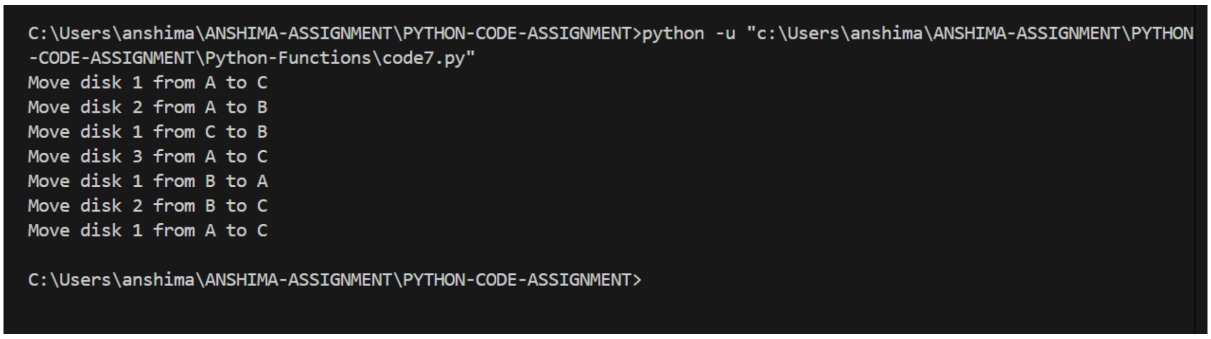
```
import time
def time_it(func):
    def wrapper(*args, **kwargs):
        start = time.time()
        result = func(*args, **kwargs)
        end = time.time()
        print(f"Execution time: {end - start:.6f} seconds")
        return result
    return wrapper

@time_it
def execution_time():
    time.sleep(5)
    return "Done"

print(execution_time())
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Python-Functions\code6.py"
Execution time: 5.001104 seconds
Done

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

7. Write a recursive function to solve the Tower of Hanoi problem.

```
def tower_of_hanoi(n, source, target, auxiliary):
    if n == 1:
        print(f"Move disk 1 from {source} to {target}")
        return
    tower_of_hanoi(n - 1, source, auxiliary, target)
    print(f"Move disk {n} from {source} to {target}")
    tower_of_hanoi(n - 1, auxiliary, target, source)

tower_of_hanoi(3, 'A', 'C', 'B')
```

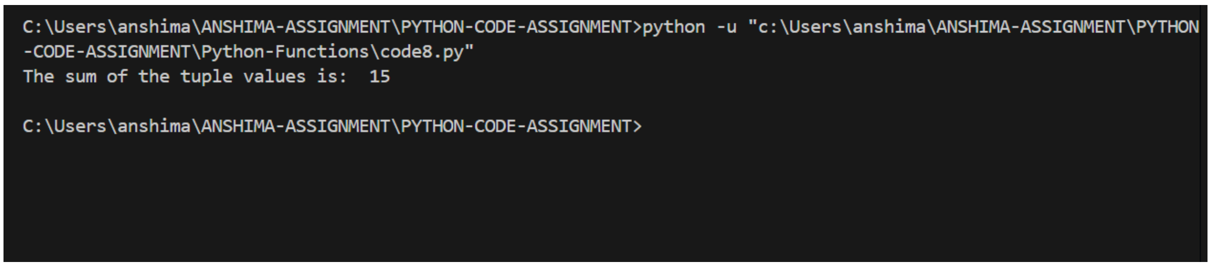


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Python-Functions\code7.py"
Move disk 1 from A to C
Move disk 2 from A to B
Move disk 1 from C to B
Move disk 3 from A to C
Move disk 1 from B to A
Move disk 2 from B to C
Move disk 1 from A to C

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

8. Implement a function that uses variable-length arguments to sum any number of

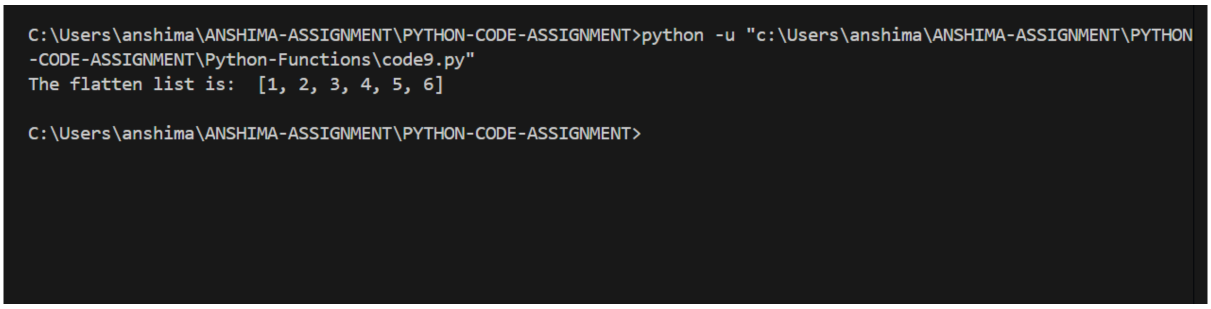
   inputs.

```
def variable_sum(*args):
    return sum(args)

tuple_values=(1,2,3,4,5)
print("The sum of the tuple values is: ", variable_sum(*tuple_values))
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Python-Functions\code8.py"
The sum of the tuple values is:  15

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

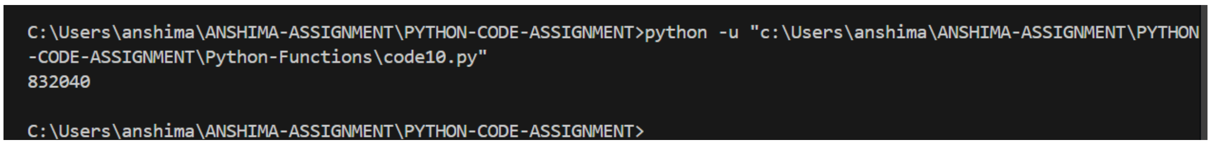
9. Write a function that flattens a nested list using recursion.

```
def flatten_list(nested):
    result = []
    for item in nested:
        if isinstance(item, list):
            result.extend(flatten_list(item))
        else:
            result.append(item)
    return result

print("The flatten list is: ",flatten_list([1, [2, [3, 4], 5], 6]))
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Python-Functions\code9.py"
The flatten list is:  [1, 2, 3, 4, 5, 6]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

10. Implement a memoized version of the Fibonacci sequence.

```
def memoized_fibonacci():
    memo = {}

    def fib(n):
        if n in memo:
            return memo[n]
        if n <= 1:
            memo[n] = n
        else:
            memo[n] = fib(n - 1) + fib(n - 2)
        return memo[n]

    return fib

fib = memoized_fibonacci()
print(fib(30))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Python-Functions\code10.py"
832040

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

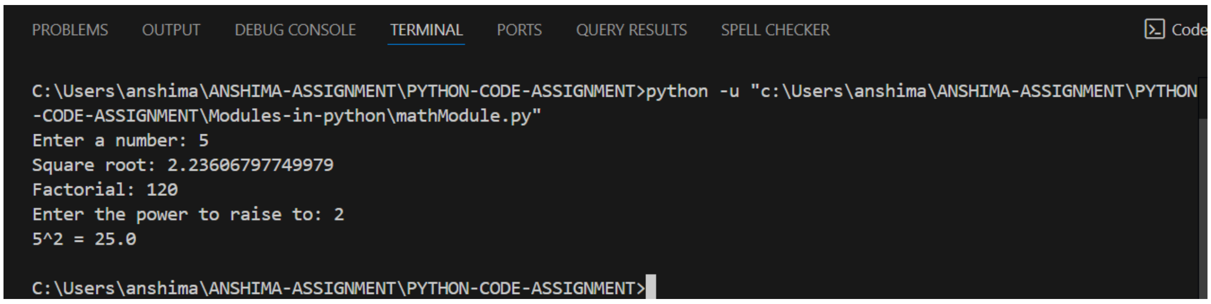
MODULES IN PYTHON

1. Create a custom module with functions to add, subtract, multiply, and divide two numbers.

```
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 a / b
    else:
        return "Cannot divide by zero"
```

2. Use the `math` module to calculate square root, factorial, and power of a number.

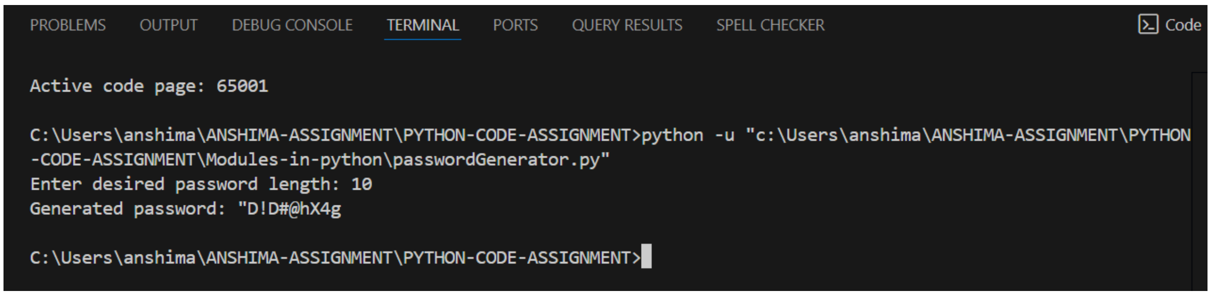
```
import math
num = int(input("Enter a number: "))
print("Square root:", math.sqrt(num))
print("Factorial:", math.factorial(num))
power = int(input("Enter the power to raise to: "))
print(f"{num}^{power} =", math.pow(num, power))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Modules-in-python\mathModule.py"
Enter a number: 5
Square root: 2.23606797749979
Factorial: 120
Enter the power to raise to: 2
5^2 = 25.0

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

3. Write a program that uses `random` to generate a password of given length.

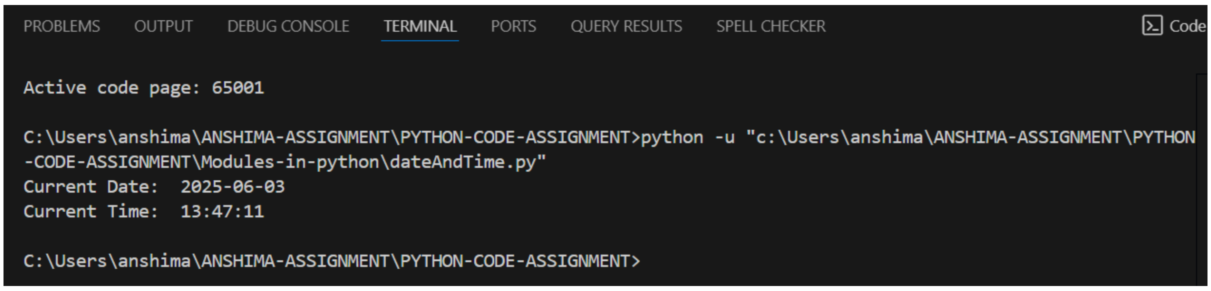
```
import random
import string

def generate_password(length):
    all_chars = string.ascii_letters + string.digits + string.punctuation
    password = ''.join(random.choice(all_chars) for _ in range(length))
    return password

length = int(input("Enter desired password length: "))
print("Generated password:", generate_password(length))
```



```
Active code page: 65001

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Modules-in-python\passwordGenerator.py"
Enter desired password length: 10
Generated password: "D!D#@hX4g

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

4. Create a program using the `datetime` module to display the current date and time.

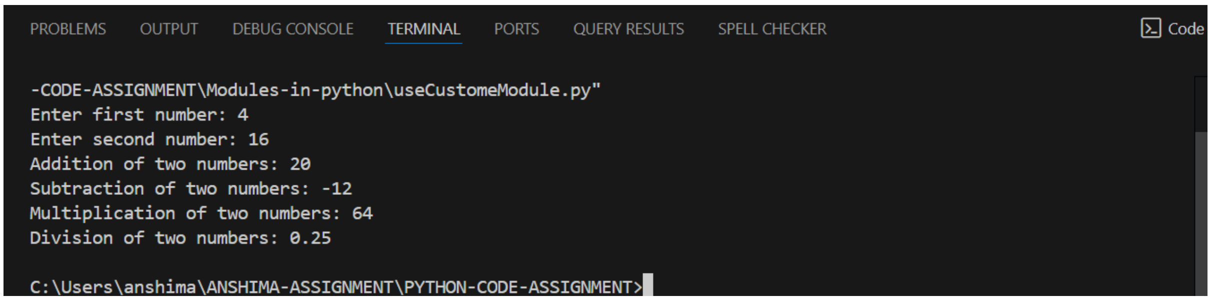
```
from datetime import datetime
now = datetime.now()
print("Current Date: ",now.strftime("%Y-%m-%d"))
print("Current Time: ",now.strftime("%H:%M:%S"))
```



```
Active code page: 65001

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Modules-in-python\dateAndTime.py"
Current Date:  2025-06-03
Current Time:  13:47:11

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

5. Import a custom module and use its functions in another script.

```
 import customModule

a = int(input("Enter first number: "))
b = int(input("Enter second number: "))

print("Addition of two numbers:", customModule.add(a, b))
print("Subtraction of two numbers:", customModule.subtract(a, b))
print("Multiplication of two numbers:", customModule.multiply(a, b))
print("Division of two numbers:", customModule.divide(a, b))
```

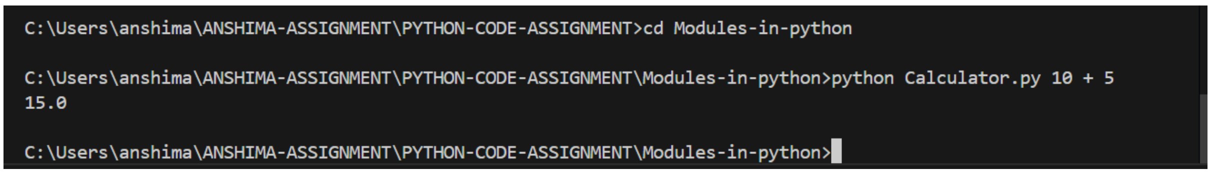


```
-CODE-ASSIGNMENT\Modules-in-python\useCustomeModule.py"
Enter first number: 4
Enter second number: 16
Addition of two numbers: 20
Subtraction of two numbers: -12
Multiplication of two numbers: 64
Division of two numbers: 0.25

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

6. Build a command-line utility using `argparse` to perform arithmetic operations.

```
import argparse

parser = argparse.ArgumentParser(description="Simple CLI Calculator")
```

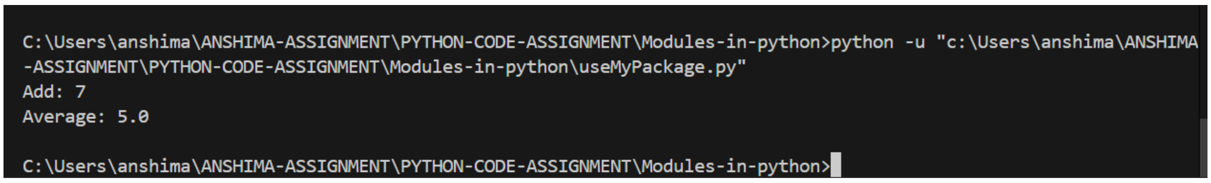
```
parser.add_argument("num1", type=float)
parser.add_argument("operator", choices=["+", "-", "*", "/"])
parser.add_argument("num2", type=float)

args = parser.parse_args()

if args.operator == "+":
    print(args.num1 + args.num2)
elif args.operator == "-":
    print(args.num1 - args.num2)
elif args.operator == "*":
    print(args.num1 * args.num2)
elif args.operator == "/":
    if args.num2 != 0:
        print(args.num1 / args.num2)
    else:
        print("Error: Division by zero")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>cd Modules-in-python

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT\Modules-in-python>python Calculator.py 10 + 5
15.0

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT\Modules-in-python>
```

7. Create and use a package with multiple modules in it.

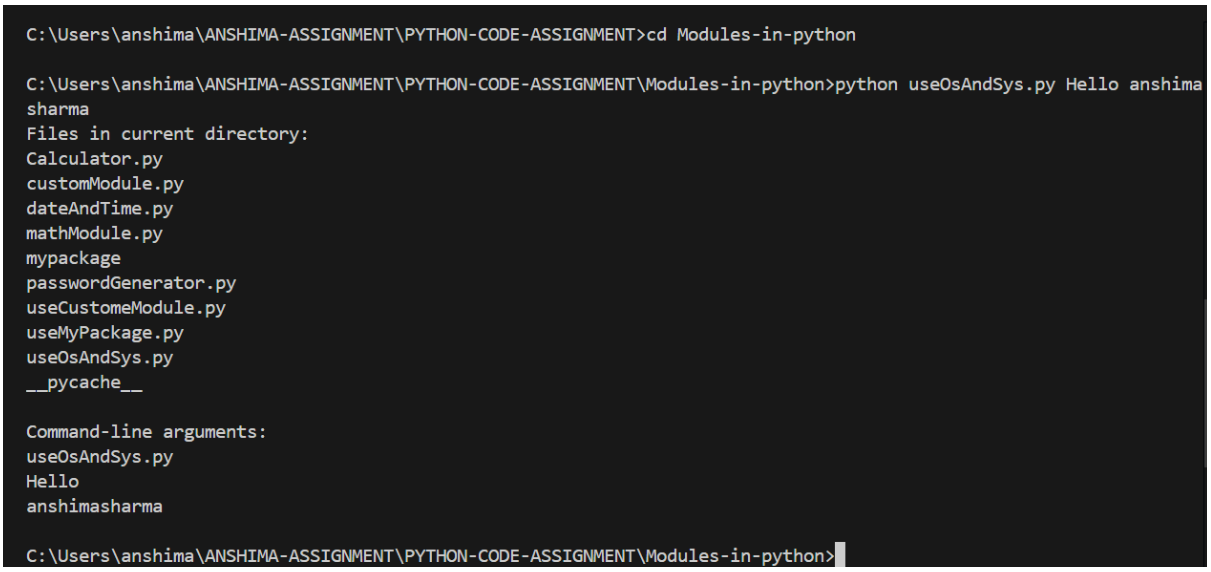
```
from mypackage import arithmetic, stats

print("Add:", arithmetic.add(3, 4))
print("Average:", stats.average([3, 5, 7]))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT\Modules-in-python>python -u "c:\Users\anshima\ANSHIMA
-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT\Modules-in-python\useMyPackage.py"
Add: 7
Average: 5.0

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT\Modules-in-python>
```

8. Develop a program that uses `os` and `sys` modules to list files and command-line args.

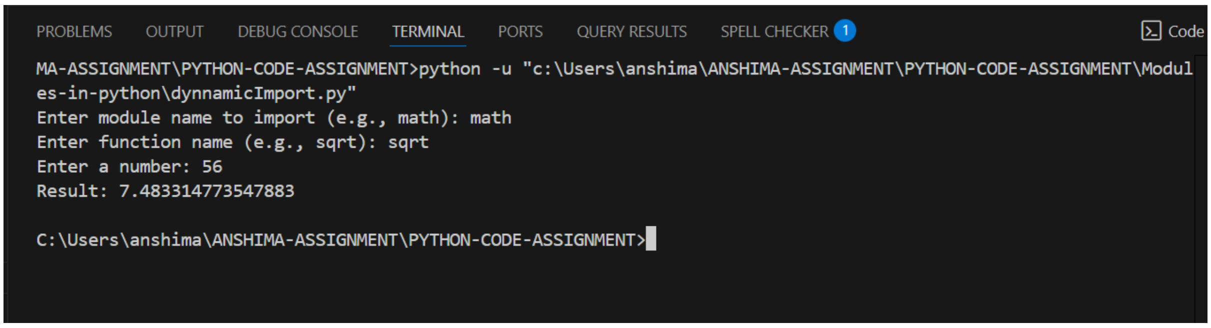
```
import os
import sys
print("Files in current directory:")
for f in os.listdir('.'):
    print(f)

print("\nCommand-line arguments:")
for arg in sys.argv:
    print(arg)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>cd Modules-in-python

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT\Modules-in-python>python useOsAndSys.py Hello anshima
sharma
Files in current directory:
Calculator.py
customModule.py
dateAndTime.py
mathModule.py
mypackage
passwordGenerator.py
useCustomeModule.py
useMyPackage.py
useOsAndSys.py
__pycache__

Command-line arguments:
useOsAndSys.py
Hello
anshimasharma

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT\Modules-in-python>
```

9. Use `importlib` to dynamically import a module and invoke a function.

```
import importlib

module_name = input("Enter module name to import (e.g., math): ")
function_name = input("Enter function name (e.g., sqrt): ")

try:
    mod = importlib.import_module(module_name)
    func = getattr(mod, function_name)
    arg = float(input("Enter a number: "))
    print("Result:", func(arg))
except (ImportError, AttributeError):
    print("Module or function not found.")
```

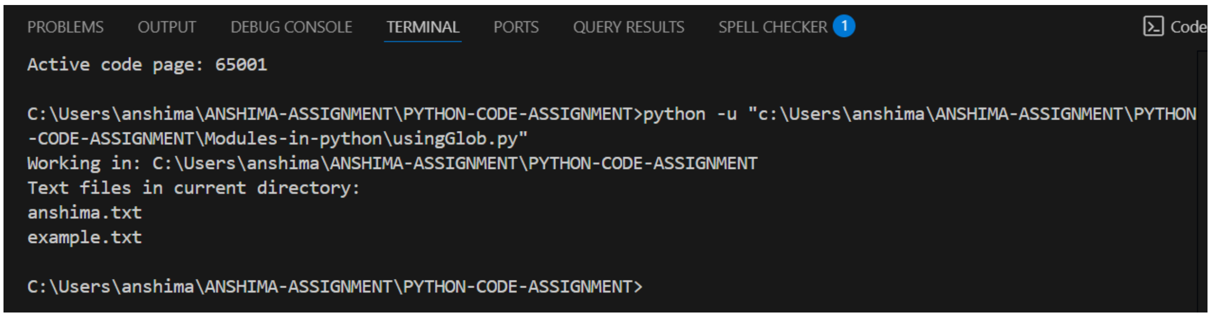


```
MA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT\Modul
es-in-python\dynnamicImport.py"
Enter module name to import (e.g., math): math
Enter function name (e.g., sqrt): sqrt
Enter a number: 56
Result: 7.483314773547883

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

10. Implement a Python script that uses `glob` to search for all `.txt` files in a directory.

```
import os
import glob

with open("example.txt", "w") as f:
    f.write("Test content")

print("Working in:", os.getcwd())
```

```
txt_files = glob.glob("*.txt")
print("Text files in current directory:")
for file in txt_files:
    print(file)
```

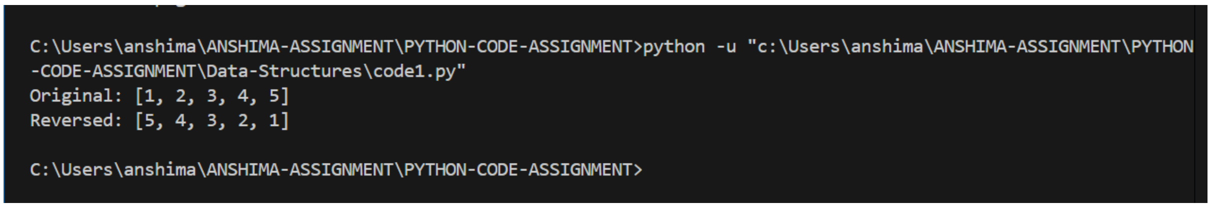


```
Active code page: 65001

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Modules-in-python\usingGlob.py"
Working in: C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT
Text files in current directory:
anshima.txt
example.txt

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

DATA STRUCTURES

1. Implement a function to reverse a list without using built-in reverse().

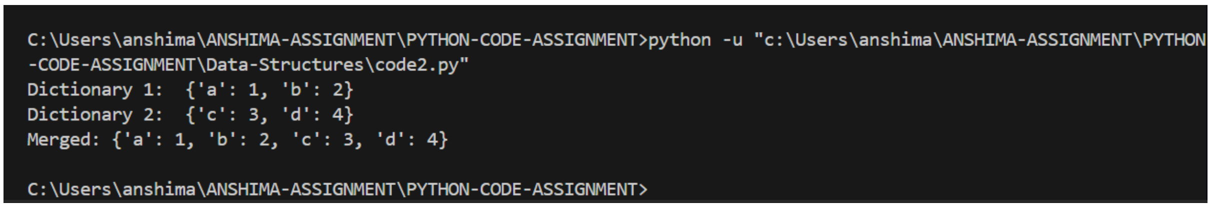
```
def reverse_list(lst):
    reversed_list = []
    for i in range(len(lst)-1, -1, -1):
        reversed_list.append(lst[i])
    return reversed_list

nums = [1, 2, 3, 4, 5]
print("Original:", nums)
print("Reversed:", reverse_list(nums))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Data-Structures\code1.py"
Original: [1, 2, 3, 4, 5]
Reversed: [5, 4, 3, 2, 1]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

2. Write a function to merge two dictionaries.

```
def merge_dicts(d1, d2):
    merged = d1.copy()
    merged.update(d2)
    return merged

dict1 = {'a': 1, 'b': 2}
dict2 = {'c': 3, 'd': 4}
print("Dictionary 1: ",dict1)
print("Dictionary 2: ",dict2)
print("Merged:", merge_dicts(dict1, dict2))
```

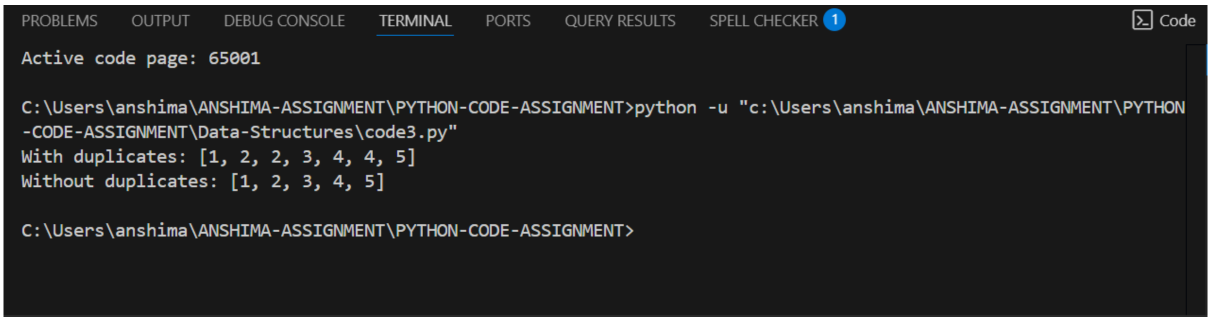


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Data-Structures\code2.py"
Dictionary 1:  {'a': 1, 'b': 2}
Dictionary 2:  {'c': 3, 'd': 4}
Merged: {'a': 1, 'b': 2, 'c': 3, 'd': 4}

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

3. Develop a function that removes duplicate elements from a list.

```
def remove_duplicates(lst):
    result = []
    for item in lst:
        if item not in result:
            result.append(item)
    return result
```

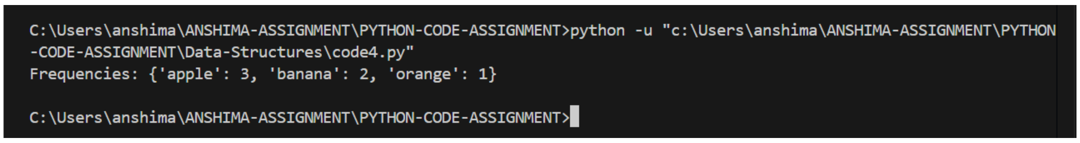
```
items = [1, 2, 2, 3, 4, 4, 5]
print("With duplicates:", items)
print("Without duplicates:", remove_duplicates(items))
```



```
Active code page: 65001

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Data-Structures\code3.py"
With duplicates: [1, 2, 2, 3, 4, 4, 5]
Without duplicates: [1, 2, 3, 4, 5]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

4. Create a function that counts the frequency of each word in a list.

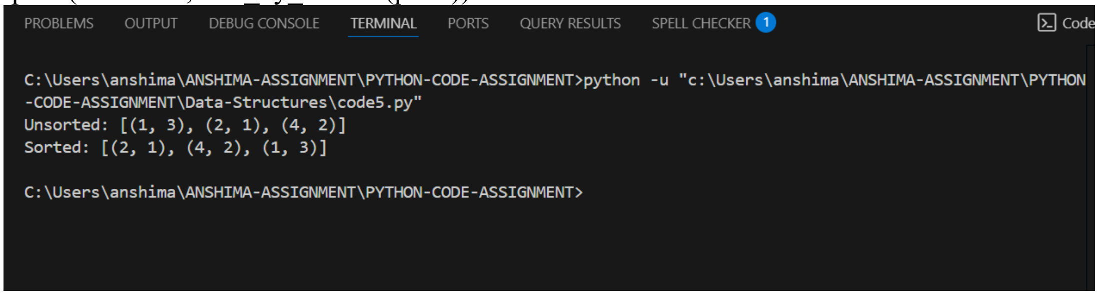
```
def count_words(words):
    freq = {}
    for word in words:
        if word in freq:
            freq[word] += 1
        else:
            freq[word] = 1
    return freq

word_list = ['apple', 'banana', 'apple', 'orange', 'banana', 'apple']
print("Frequencies:", count_words(word_list))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Data-Structures\code4.py"
Frequencies: {'apple': 3, 'banana': 2, 'orange': 1}

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

5. Write a program to sort a list of tuples based on the second element.

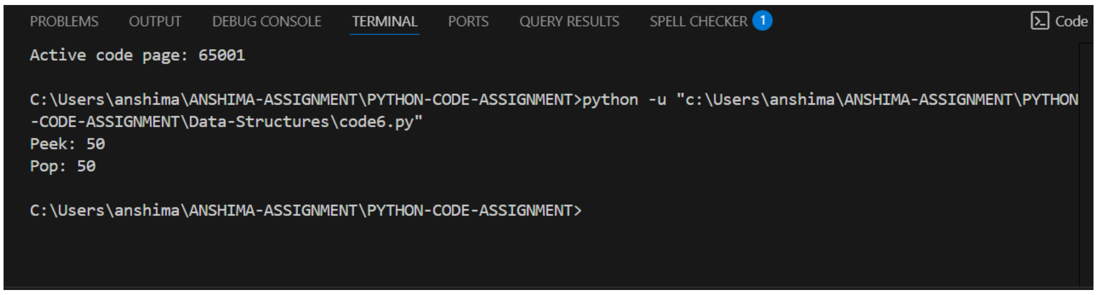
```
def sort_by_second(tuples):
    return sorted(tuples, key=lambda x: x[1])

pairs = [(1, 3), (2, 1), (4, 2)]
print("Unsorted:", pairs)
print("Sorted:", sort_by_second(pairs))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Data-Structures\code5.py"
Unsorted: [(1, 3), (2, 1), (4, 2)]
Sorted: [(2, 1), (4, 2), (1, 3)]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

6. Implement a stack using list with push, pop, and peek operations.

```
class Stack:
    def __init__(self):
        self.stack = []

    def push(self, item):
        self.stack.append(item)

    def pop(self):
        if self.stack:
            return self.stack.pop()
        return "Stack is empty"

    def peek(self):
        if self.stack:
            return self.stack[-1]
        return "Stack is empty"

s = Stack()
s.push(10)
s.push(20)
s.push(40)
s.push(50)
print("Peek:", s.peek())
print("Pop:", s.pop())
```

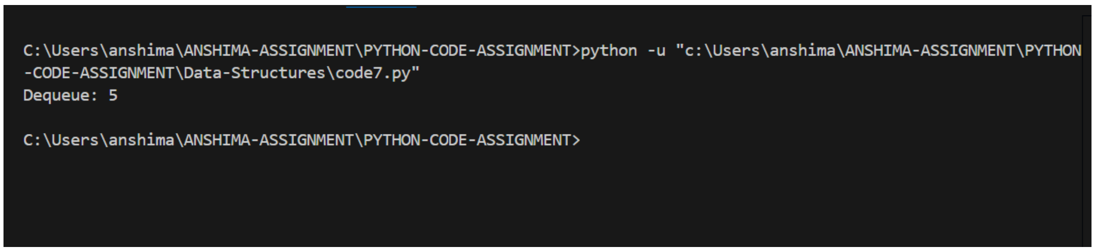


```
Active code page: 65001

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Data-Structures\code6.py"
Peek: 50
Pop: 50

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

7. Create a queue using collections.deque and implement enqueue and dequeue.

```
from collections import deque
class Queue:
    def __init__(self):
        self.queue = deque()

    def enqueue(self, item):
        self.queue.append(item)

    def dequeue(self):
        if self.queue:
            return self.queue.popleft()
        return "Queue is empty"

q = Queue()
```

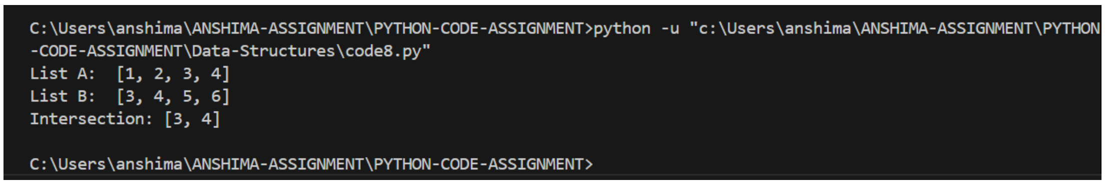
```
q.enqueue(5)
q.enqueue(10)
q.enqueue(45)
q.enqueue(34)
print("Dequeue:", q.dequeue())
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Data-Structures\code7.py"
Dequeue: 5

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

8. Write a function to find the intersection of two lists.

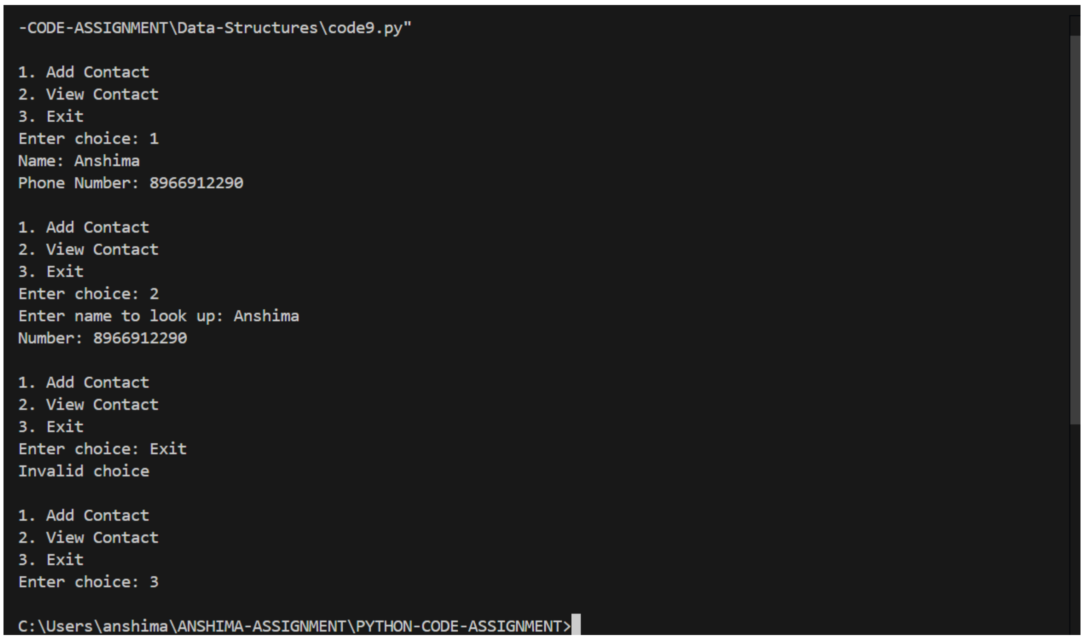
```
def find_intersection(list1, list2):
    result = []
    for item in list1:
        if item in list2 and item not in result:
            result.append(item)
    return result

a = [1, 2, 3, 4]
b = [3, 4, 5, 6]
print("List A: ",a)
print("List B: ",b)
print("Intersection:", find_intersection(a, b))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Data-Structures\code8.py"
List A:  [1, 2, 3, 4]
List B:  [3, 4, 5, 6]
Intersection: [3, 4]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

9. Create a program that uses a dictionary to implement a phonebook.

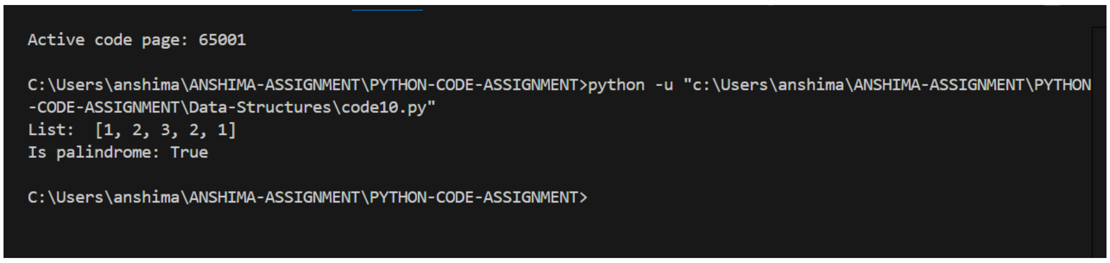
```
def phonebook_program():
    phonebook = {}
    while True:
        print("\n1. Add Contact\n2. View Contact\n3. Exit")
        choice = input("Enter choice: ")
        if choice == '1':
            name = input("Name: ")
            number = input("Phone Number: ")
            phonebook[name] = number
        elif choice == '2':
            name = input("Enter name to look up: ")
            print("Number:", phonebook.get(name, "Not found"))
        elif choice == '3':
            break
        else:
            print("Invalid choice")

phonebook_program()
```



```
-CODE-ASSIGNMENT\Data-Structures\code9.py"

1. Add Contact
2. View Contact
3. Exit
Enter choice: 1
Name: Anshima
Phone Number: 8966912290

1. Add Contact
2. View Contact
3. Exit
Enter choice: 2
Enter name to look up: Anshima
Number: 8966912290

1. Add Contact
2. View Contact
3. Exit
Enter choice: Exit
Invalid choice

1. Add Contact
2. View Contact
3. Exit
Enter choice: 3

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

10. Implement a function to check if a list is a palindrome.

```
def is_list_palindrome(lst):
    start = 0
    end = len(lst) - 1
    while start < end:
        if lst[start] != lst[end]:
            return False
        start += 1
        end -= 1
    return True
list = [1, 2, 3, 2, 1]
print("List: ",list)
print("Is palindrome:", is_list_palindrome(list))
```

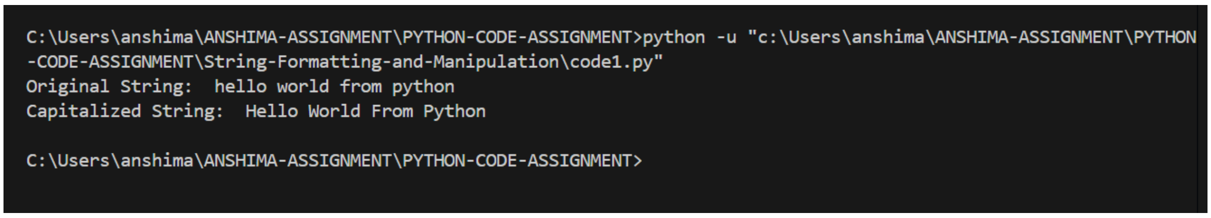


```
Active code page: 65001

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Data-Structures\code10.py"
List:  [1, 2, 3, 2, 1]
Is palindrome: True

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

STRING FORMATTING AND MANIPULATION

1. Write a function that capitalizes the first letter of each word in a string.

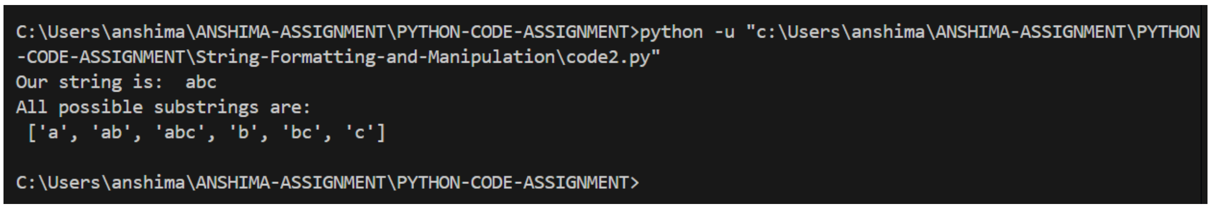
```
 def capitalize_words(text):
    words = text.split()
    capitalized = [word[0].upper() + word[1:] if word else '' for word in words]
    return ' '.join(capitalized)

s = "hello world from python"
print("Original String: ",s)
print("Capitalized String: ",capitalize_words(s))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\String-Formatting-and-Manipulation\code1.py"
Original String:  hello world from python
Capitalized String:  Hello World From Python

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

2. Create a program that finds all substrings of a given string.

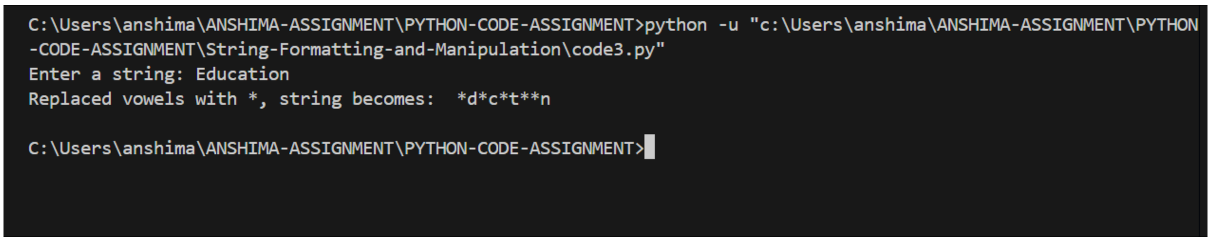
```
 def all_substrings(s):
    substrings = []
    for i in range(len(s)):
        for j in range(i + 1, len(s) + 1):
            substrings.append(s[i:j])
    return substrings

str="abc"
print("Our string is: ",str)
print("All possible substrings are: \n",all_substrings("abc"))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\String-Formatting-and-Manipulation\code2.py"
Our string is:  abc
All possible substrings are:
 ['a', 'ab', 'abc', 'b', 'bc', 'c']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

3. Write a function that replaces all vowels in a string with '*' symbol.

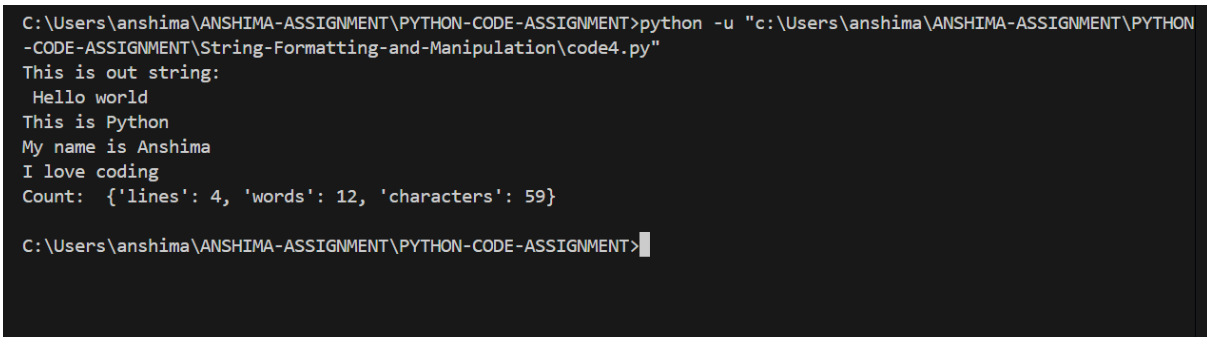
```
def replace_vowels(s):
    vowels = 'aeiouAEIOU'
    result = ''
    for char in s:
        if char in vowels:
            result += '*'
        else:
            result += char
    return result

str=input("Enter a string: ")
print("Replaced vowels with *, string becomes: ",replace_vowels(str))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\String-Formatting-and-Manipulation\code3.py"
Enter a string: Education
Replaced vowels with *, string becomes:  *d*c*t**n

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

4. Develop a function that counts words, characters, and lines in a string.

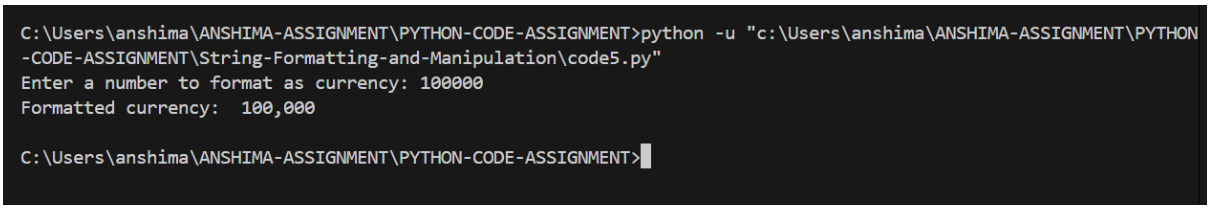
```
 def count_text_stats(text):
    lines = text.splitlines()
    words = text.split()
    characters = len(text)
    return {
        'lines': len(lines),
        'words': len(words),
        'characters': characters
    }

text = "Hello world\nThis is Python\nMy name is Anshima\nI love coding"
print("This is out string: \n",text)
print("Count: ",count_text_stats(text))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\String-Formatting-and-Manipulation\code4.py"
This is out string:
 Hello world
This is Python
My name is Anshima
I love coding
Count:  {'lines': 4, 'words': 12, 'characters': 59}

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

5. Write a script to format a number as currency (e.g., 1000000 -> 1,000,000).

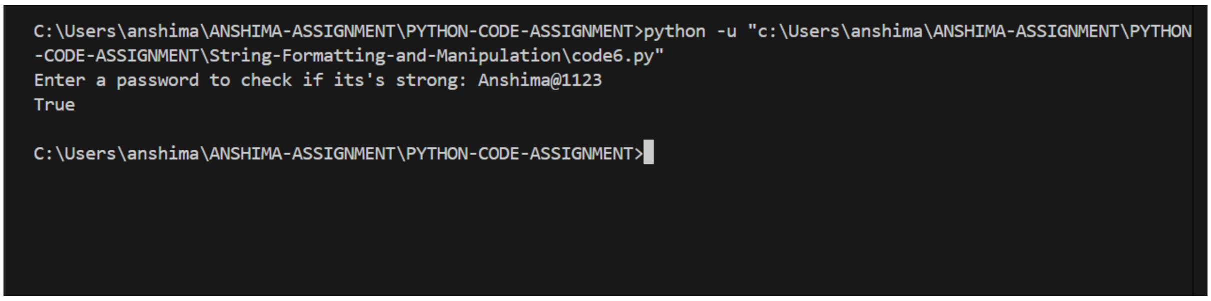
```
def format_currency(number):
    return "{:,}".format(number)

num=int(input("Enter a number to format as currency: "))
print("Formatted currency: ",format_currency(num))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\String-Formatting-and-Manipulation\code5.py"
Enter a number to format as currency: 100000
Formatted currency:  100,000

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

6. Implement a function that validates a strong password based on given criteria.

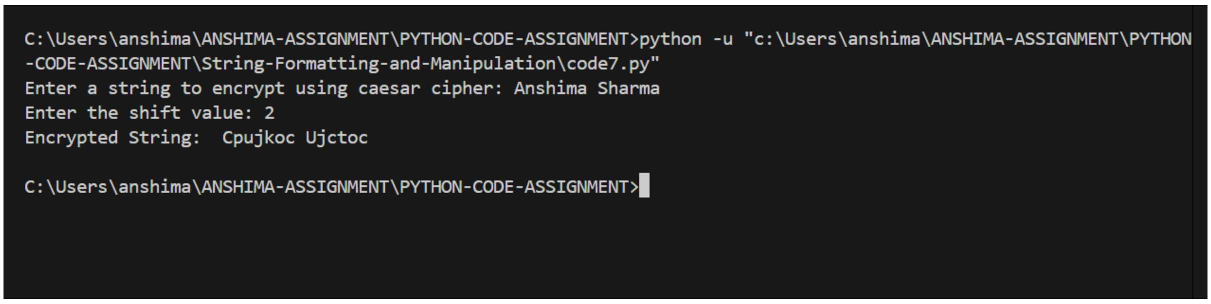
```
def is_strong_password(pwd):
   if len(pwd) < 8:
       return False
   has_upper = any(char.isupper() for char in pwd)
   has_lower = any(char.islower() for char in pwd)
   has_digit = any(char.isdigit() for char in pwd)
   has_special = any(char in "!@#$%^&*()-_+=" for char in pwd)
   return has_upper and has_lower and has_digit and has_special

str=input("Enter a password to check if its's strong: ")
print(is_strong_password(str))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\String-Formatting-and-Manipulation\code6.py"
Enter a password to check if its's strong: Anshima@1123
True

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

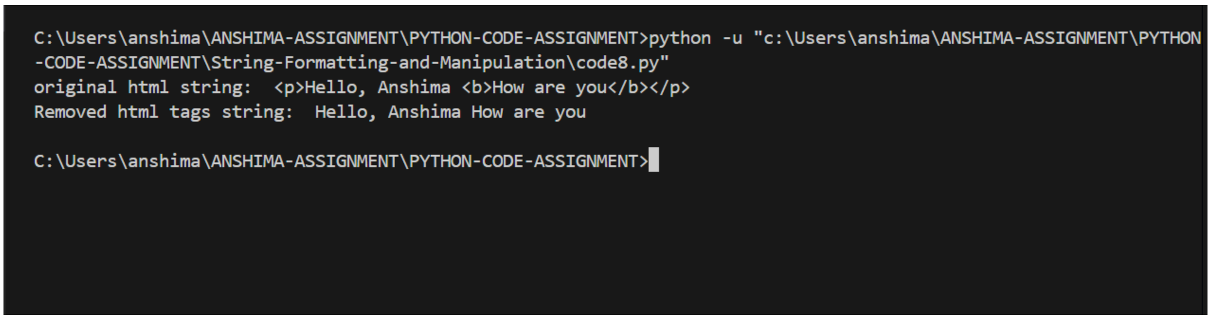
7. Write a script that encodes a string using Caesar cipher.

```
def caesar_cipher(text, shift):
    result = ''
    for char in text:
        if char.isalpha():
            base = ord('A') if char.isupper() else ord('a')
            shifted = chr((ord(char) - base + shift) % 26 + base)
            result += shifted
        else:
            result += char
    return result

str=input("Enter a string to encrypt using caesar cipher: ")
shift=int(input("Enter the shift value: "))

print("Encrypted String: ",caesar_cipher(str,shift))
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\String-Formatting-and-Manipulation\code7.py"
Enter a string to encrypt using caesar cipher: Anshima Sharma
Enter the shift value: 2
Encrypted String:  Cpujkoc Ujctoc

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

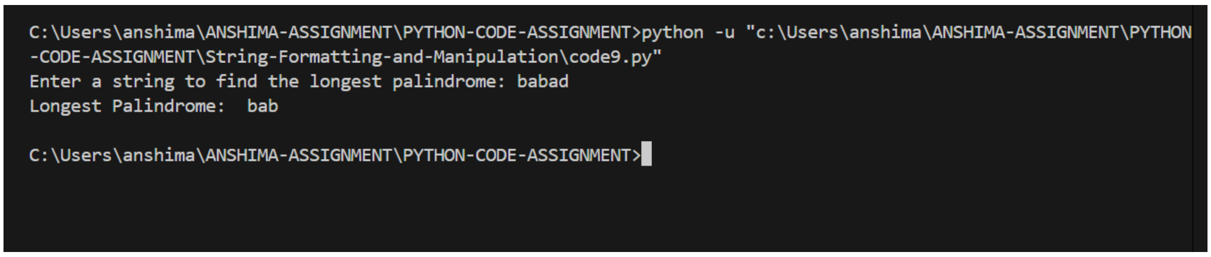
8. Create a function to remove HTML tags from a string.

```
 import re
def remove_html_tags(html):
   return re.sub(r'<[^>]+>', '', html)

html_input = "<p>Hello, Anshima <b>How are you</b></p>"
print("original html string: ",html_input)
print("Removed html tags string: ",remove_html_tags(html_input))
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\String-Formatting-and-Manipulation\code8.py"
original html string:  <p>Hello, Anshima <b>How are you</b></p>
Removed html tags string:  Hello, Anshima How are you

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

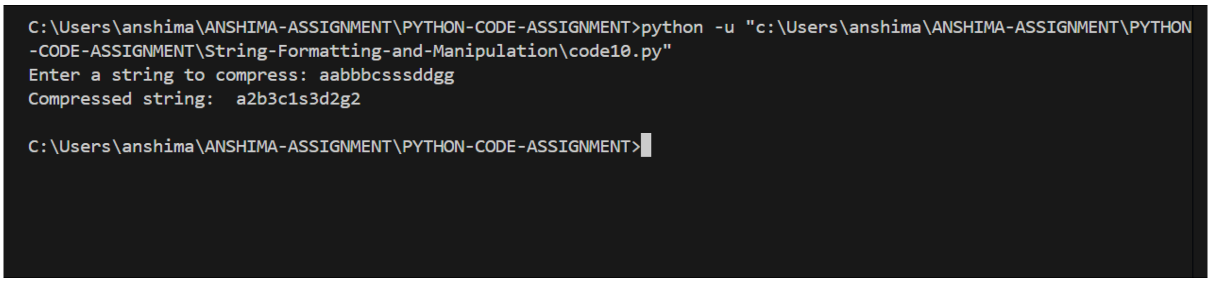
9. Develop a function that finds the longest palindromic substring.

```
 def longest_palindrome(s):
   longest = ''
   for i in range(len(s)):
       for j in range(i, len(s)):
           substr = s[i:j+1]
           if substr == substr[::-1] and len(substr) > len(longest):
               longest = substr
   return longest

str = input("Enter a string to find the longest palindrome: ")
print("Longest Palindrome: ",longest_palindrome(str))
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\String-Formatting-and-Manipulation\code9.py"
Enter a string to find the longest palindrome: babad
Longest Palindrome:  bab

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

10. Implement a string compression algorithm (e.g., aabcccccaaa -> a2b1c5a3).

```
def compress_string(s):
    if not s:
        return ""
    result = ""
    count = 1
    for i in range(1, len(s)):
        if s[i] == s[i-1]:
            count += 1
        else:
            result += s[i-1] + str(count)
            count = 1
    result += s[-1] + str(count)
    return result

valstr=input("Enter a string to compress: ")
print("Compressed string: ",compress_string(valstr))
```

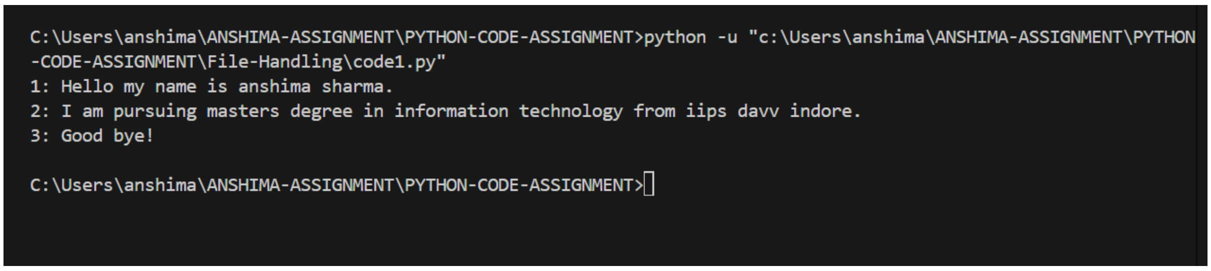


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\String-Formatting-and-Manipulation\code10.py"
Enter a string to compress: aabbbcsssddgg
Compressed string:  a2b3c1s3d2g2

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

FILE HANDLING

1. Write a script that reads a file and prints each line with line numbers.

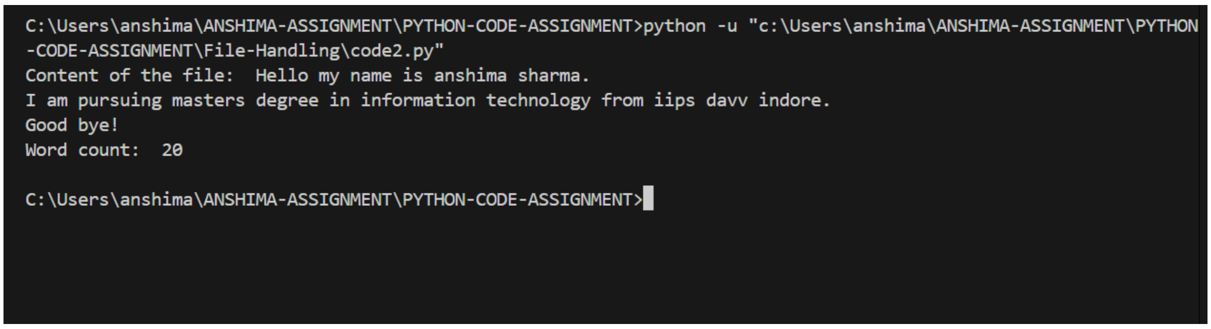
```
def print_file_with_line_numbers(filename):
    with open(filename, 'r') as file:
        for idx, line in enumerate(file, start=1):
            print(f"{idx}: {line.strip()}")

print_file_with_line_numbers("anshima.txt")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\File-Handling\code1.py"
1: Hello my name is anshima sharma.
2: I am pursuing masters degree in information technology from iips davv indore.
3: Good bye!

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

2. Create a function to count the number of words in a text file.

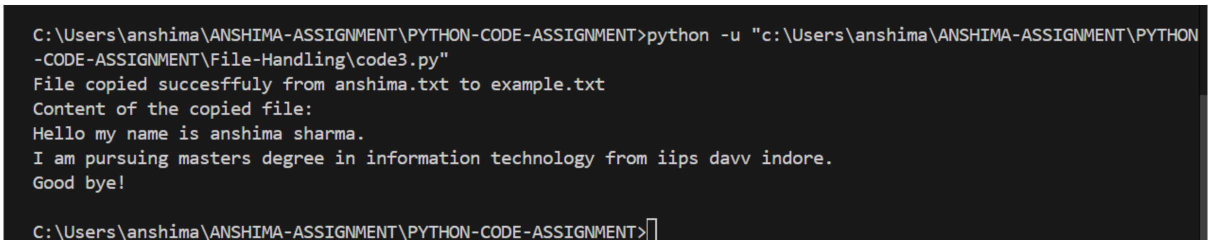
```
def count_words_in_file(filename):
    with open(filename, 'r') as file:
        text = file.read()
    words = text.split()
    print("Content of the file: ",text)
    return len(words)

print("Word count: ", count_words_in_file("anshima.txt"))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\File-Handling\code2.py"
Content of the file:  Hello my name is anshima sharma.
I am pursuing masters degree in information technology from iips davv indore.
Good bye!
Word count:  20

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

3. Write a program to copy the contents of one file into another.

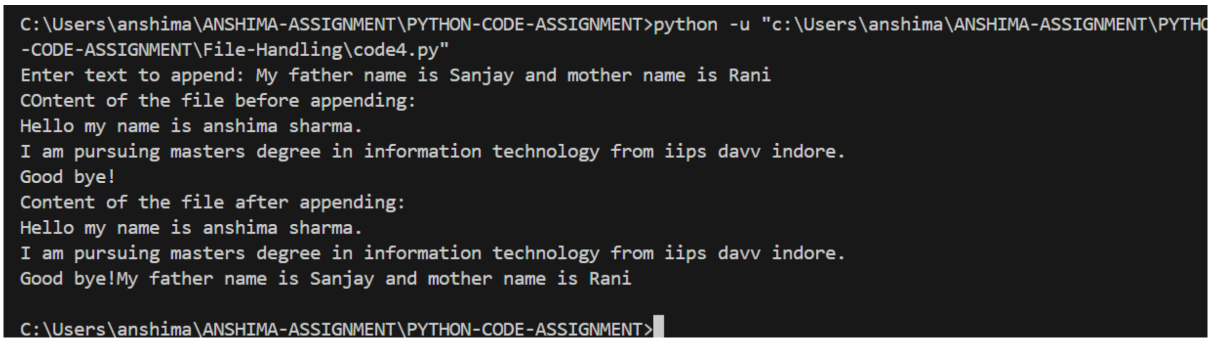
```
def copy_file(source, destination):
    with open(source, 'r') as src, open(destination, 'w') as dest:
        for line in src:
            dest.write(line)

copy_file("anshima.txt", "example.txt")
print("File copied succesffuly from", "anshima.txt", "to", "example.txt")
print("Content of the copied file: ")

with open("example.txt", 'r') as f:
    for line in f:
        print(line.strip())
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\File-Handling\code3.py"
File copied succesffuly from anshima.txt to example.txt
Content of the copied file:
Hello my name is anshima sharma.
I am pursuing masters degree in information technology from iips davv indore.
Good bye!

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

4. Implement a script that appends user input to a file.

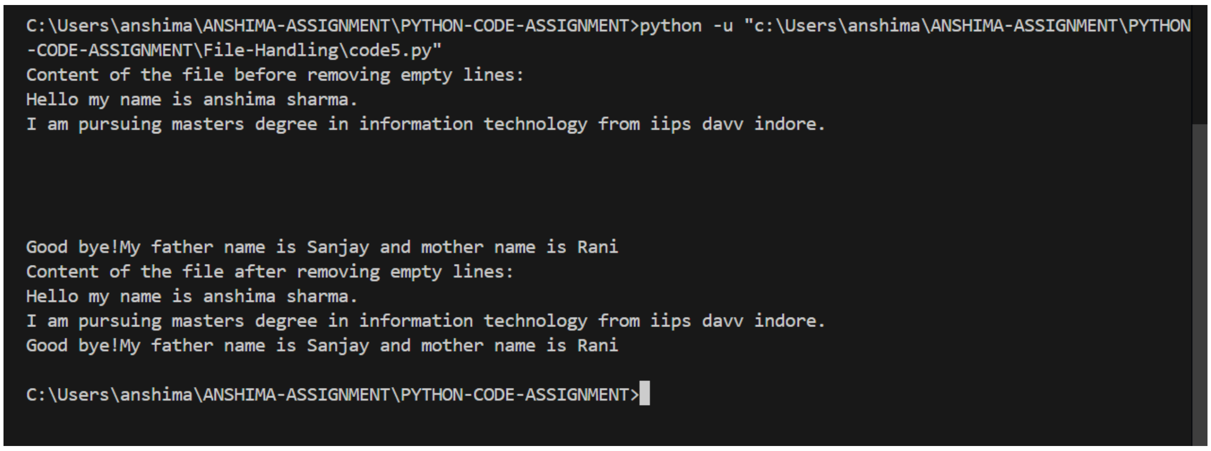
```
def append_input_to_file(filename):
    user_input = input("Enter text to append: ")
    with open(filename, 'r') as file:
        print("COntent of the file before appending: ")
        for line in file:
            print(line.strip())

    with open(filename, 'a') as file:
        file.write(user_input + '\n')

    with open(filename, 'r') as file:
        print("Content of the file after appending: ")
        for line in file:
            print(line.strip())

append_input_to_file("anshima.txt")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHO
-CODE-ASSIGNMENT\File-Handling\code4.py"
Enter text to append: My father name is Sanjay and mother name is Rani
COntent of the file before appending:
Hello my name is anshima sharma.
I am pursuing masters degree in information technology from iips davv indore.
Good bye!
Content of the file after appending:
Hello my name is anshima sharma.
I am pursuing masters degree in information technology from iips davv indore.
Good bye!My father name is Sanjay and mother name is Rani

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

5. Develop a function to read a file and remove all empty lines.

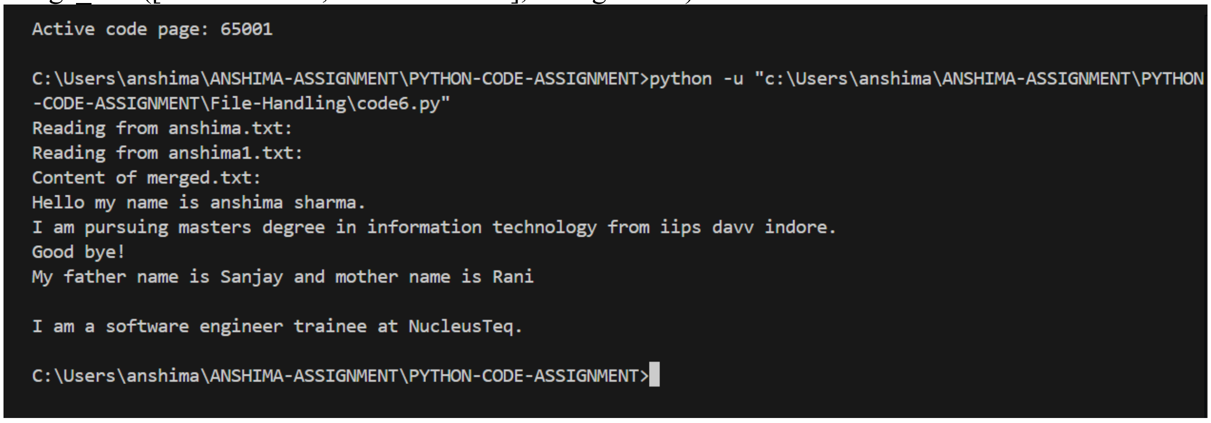
```
def remove_empty_lines(filename):
    with open(filename, 'r') as file:
        lines = file.readlines()
        print("Content of the file before removing empty lines: ")
        for line in lines:
            print(line.strip())
    non_empty_lines = [line for line in lines if line.strip()]
    with open(filename, 'w') as file:
        file.writelines(non_empty_lines)
        print("Content of the file after removing empty lines: ")
        for line in non_empty_lines:
            print(line.strip())

remove_empty_lines("anshima.txt")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\File-Handling\code5.py"
Content of the file before removing empty lines:
Hello my name is anshima sharma.
I am pursuing masters degree in information technology from iips davv indore.




Good bye!My father name is Sanjay and mother name is Rani
Content of the file after removing empty lines:
Hello my name is anshima sharma.
I am pursuing masters degree in information technology from iips davv indore.
Good bye!My father name is Sanjay and mother name is Rani

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

6. Create a script to merge multiple text files into one.

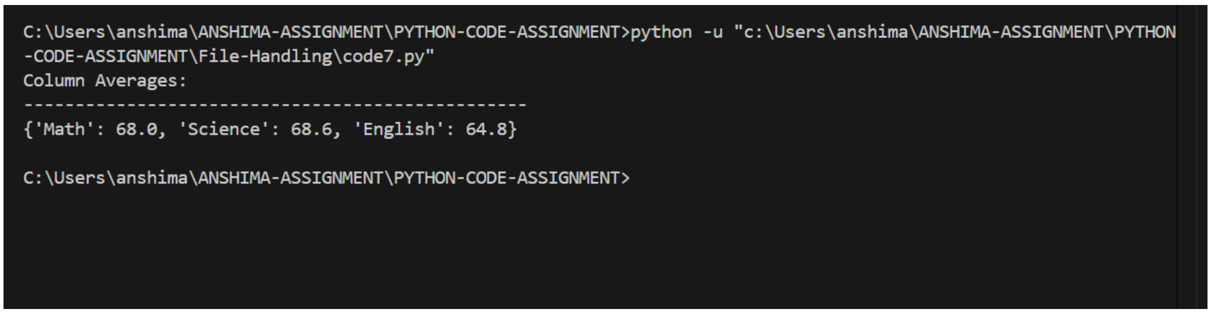
```
def merge_files(file_list, output_file):
    with open(output_file, 'w') as outfile:
        for file_name in file_list:
            with open(file_name, 'r') as infile:
                print(f"Reading from {file_name}: ")
                outfile.write(infile.read() + '\n')
    with open(output_file, 'r') as outfile:
        print(f"Content of {output_file}: ")
        for line in outfile:
            print(line.strip())

merge_files(["anshima.txt", "anshima1.txt"], "merged.txt")
```



```
Active code page: 65001

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\File-Handling\code6.py"
Reading from anshima.txt:
Reading from anshima1.txt:
Content of merged.txt:
Hello my name is anshima sharma.
I am pursuing masters degree in information technology from iips davv indore.
Good bye!
My father name is Sanjay and mother name is Rani

I am a software engineer trainee at NucleusTeq.

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

7. Write a program to read a CSV file and calculate column averages.

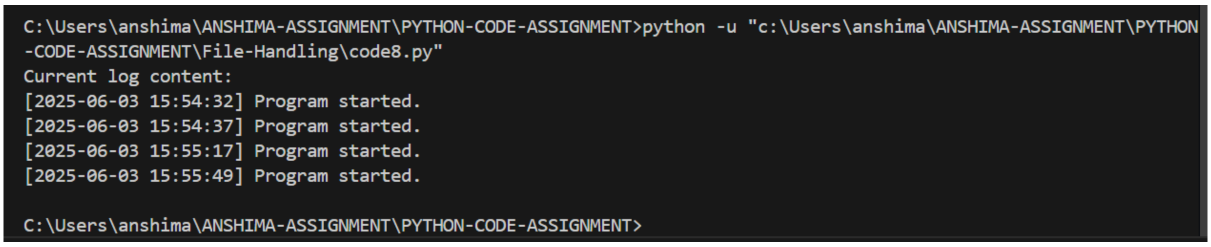
```
import csv
def column_averages(filename):
    with open(filename, newline='') as file:
        reader = csv.reader(file)
        headers = next(reader)
        sums = [0] * len(headers)
        count = 0
        for row in reader:
            for i in range(len(row)):
                try:
                    sums[i] += float(row[i])
                except ValueError:
                    pass
            count += 1
        averages = [s / count for s in sums]
        return dict(zip(headers, averages))

print("Column Averages: ")
print("-----------------------------------------------")
print(column_averages("data.csv"))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\File-Handling\code7.py"
Column Averages:
-----------------------------------------------
{'Math': 68.0, 'Science': 68.6, 'English': 64.8}

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

8. Implement a program that creates a log file with timestamped entries.

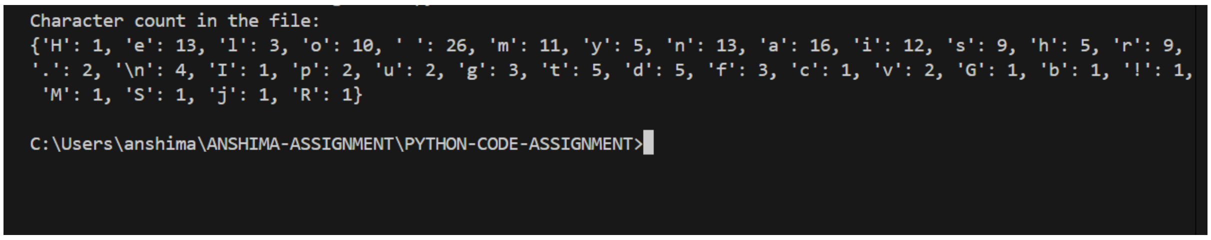
```
import datetime

def write_log(message, log_file="example.txt"):
    timestamp = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    with open(log_file, 'a') as file:
        file.write(f"[{timestamp}] {message}\n")
    with open(log_file, 'r') as file:
        print("Current log content: ")
        for line in file:
            print(line.strip())

write_log("Program started.")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\File-Handling\code8.py"
Current log content:
[2025-06-03 15:54:32] Program started.
[2025-06-03 15:54:37] Program started.
[2025-06-03 15:55:17] Program started.
[2025-06-03 15:55:49] Program started.

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

9. Write a script that reads a file and counts the frequency of each character.

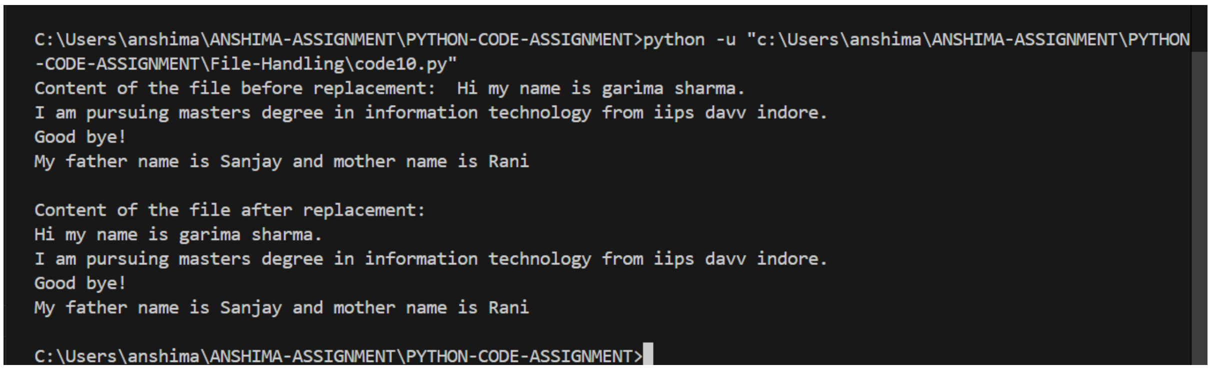
```
def count_characters(filename):
    freq = {}
    with open(filename, 'r') as file:
        text = file.read()
        for char in text:
            if char in freq:
                freq[char] += 1
            else:
                freq[char] = 1
    return freq

print("Character count in the file: ")
print(count_characters("anshima.txt"))
```



```
Character count in the file:
{'H': 1, 'e': 13, 'l': 3, 'o': 10, ' ': 26, 'm': 11, 'y': 5, 'n': 13, 'a': 16, 'i': 12, 's': 9, 'h': 5, 'r': 9,
'.': 2, '\n': 4, 'I': 1, 'p': 2, 'u': 2, 'g': 3, 't': 5, 'd': 5, 'f': 3, 'c': 1, 'v': 2, 'G': 1, 'b': 1, '!': 1,
 'M': 1, 'S': 1, 'j': 1, 'R': 1}

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

10. Create a function to replace specific words in a file with user-provided values.

```
def replace_words_in_file(filename, replacements):
    with open(filename, 'r') as file:
        content=file.read()
        print("Content of the file before replacement: ",content)

    for old_word, new_word in replacements.items():
        content = content.replace(old_word, new_word)
    with open(filename, 'w') as file:
        file.write(content)
    with open(filename, 'r') as file:
        print("Content of the file after replacement: ")
        for line in file:
            print(line.strip())

replacements = {
    "Hello": "Hi",
    "anshima": "garima"
}
replace_words_in_file("anshima.txt", replacements)
```

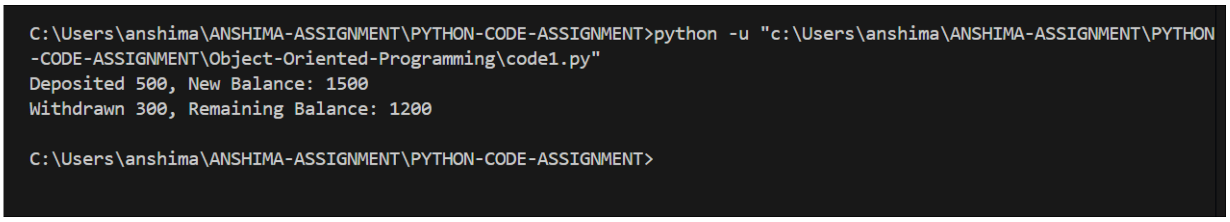


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\File-Handling\code10.py"
Content of the file before replacement:  Hi my name is garima sharma.
I am pursuing masters degree in information technology from iips davv indore.
Good bye!
My father name is Sanjay and mother name is Rani

Content of the file after replacement:
Hi my name is garima sharma.
I am pursuing masters degree in information technology from iips davv indore.
Good bye!
My father name is Sanjay and mother name is Rani

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

OOP (OBJECT-ORIENTED PROGRAMMING)

1. Create a class representing a Bank Account with deposit and withdraw methods.

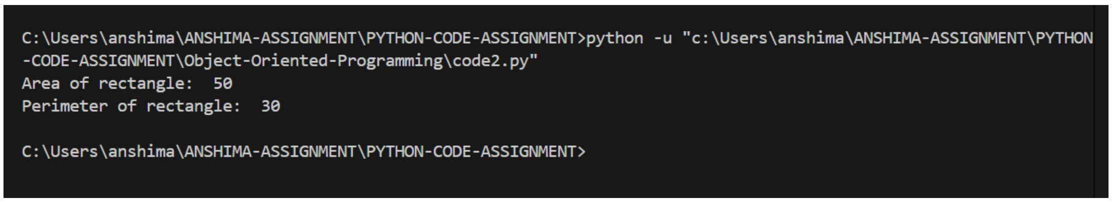
```
class BankAccount:
    def __init__(self, owner, balance=0):
        self.owner = owner
        self.balance = balance
    def deposit(self, amount):
        self.balance += amount
        print(f"Deposited {amount}, New Balance: {self.balance}")
    def withdraw(self, amount):
        if amount > self.balance:
            print("Insufficient funds.")
        else:
            self.balance -= amount
            print(f"Withdrawn {amount}, Remaining Balance: {self.balance}")

acc = BankAccount("Alice", 1000)
acc.deposit(500)
acc.withdraw(300)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Object-Oriented-Programming\code1.py"
Deposited 500, New Balance: 1500
Withdrawn 300, Remaining Balance: 1200

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

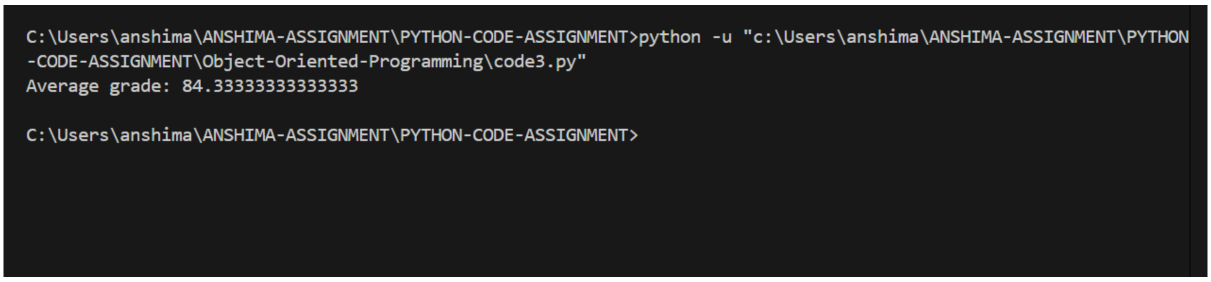
2. Implement a class for a Rectangle with methods to calculate area and perimeter.

```
class Rectangle:
    def __init__(self, length, width):
        self.length = length
        self.width = width

    def area(self):
        return self.length * self.width

    def perimeter(self):
        return 2 * (self.length + self.width)

rect = Rectangle(10, 5)
print("Area of rectangle: ", rect.area())
print("Perimeter of rectangle: ", rect.perimeter())
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Object-Oriented-Programming\code2.py"
Area of rectangle:  50
Perimeter of rectangle:  30

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

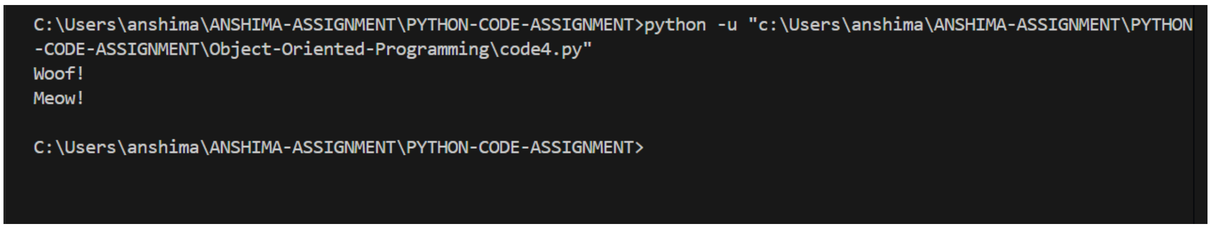
3. Create a Student class that stores name and grades, and can compute the average.

```
class Student:
    def __init__(self, name, grades):
        self.name = name
        self.grades = grades

    def average(self):
        return sum(self.grades) / len(self.grades)

s1 = Student("Bob", [85, 90, 78])
print("Average grade:", s1.average())
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Object-Oriented-Programming\code3.py"
Average grade: 84.33333333333333

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

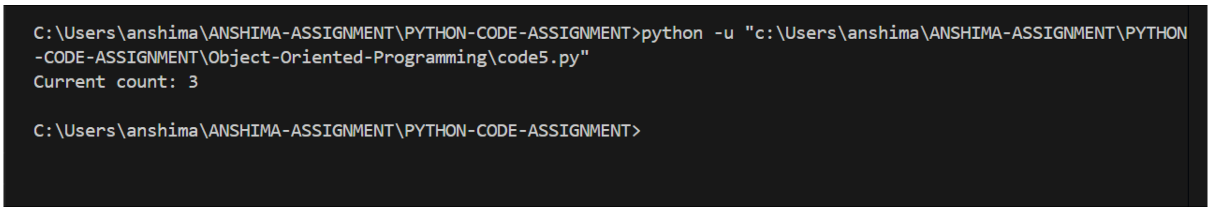
4. Implement inheritance between a base class Animal and subclasses Dog and Cat.

```
class Animal:
    def speak(self):
        print("Animal speaks")

class Dog(Animal):
    def speak(self):
        print("Woof!")

class Cat(Animal):
    def speak(self):
        print("Meow!")

dog = Dog()
cat = Cat()
dog.speak()
cat.speak()
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Object-Oriented-Programming\code4.py"
Woof!
Meow!

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

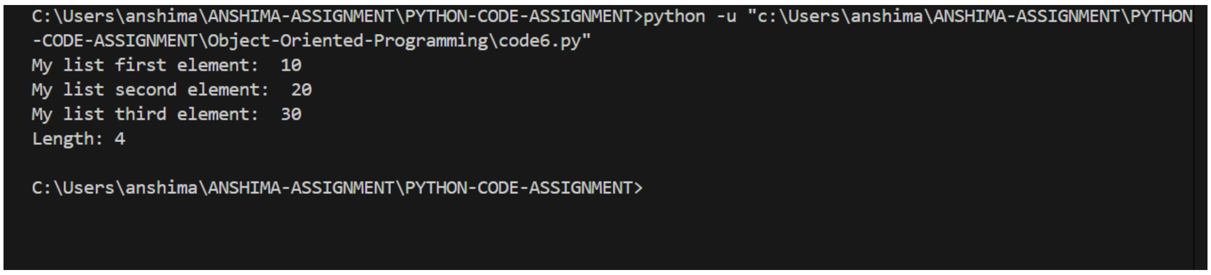
5. Write a class with a class variable shared among all instances.

```
class Counter:
    count = 0  # class variable

    def __init__(self):
        Counter.count += 1

    def show_count(self):
        print("Current count:", Counter.count)

a = Counter()
b = Counter()
c = Counter()
c.show_count()
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Object-Oriented-Programming\code5.py"
Current count: 3

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

6. Use magic methods to implement a custom class that mimics a list.

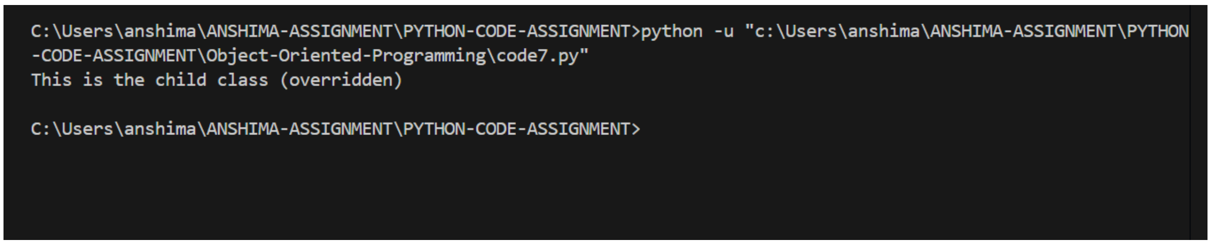
```
class MyList:
    def __init__(self):
        self.data = []

    def __getitem__(self, index):
        return self.data[index]

    def __setitem__(self, index, value):
        self.data[index] = value

    def __len__(self):
        return len(self.data)

    def append(self, value):
        self.data.append(value)

mylist = MyList()
mylist.append(10)
mylist.append(20)
mylist.append(30)
mylist.append(40)

print("My list first element: ",mylist[0])
print("My list second element: ",mylist[1])
print("My list third element: ",mylist[2])
print("Length:", len(mylist))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Object-Oriented-Programming\code6.py"
My list first element:  10
My list second element:  20
My list third element:  30
Length: 4

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

7. Implement method overriding in a subclass.

```
class Parent:
    def show(self):
        print("This is the parent class")

class Child(Parent):
    def show(self):
        print("This is the child class (overridden)")

c = Child()
c.show()
```

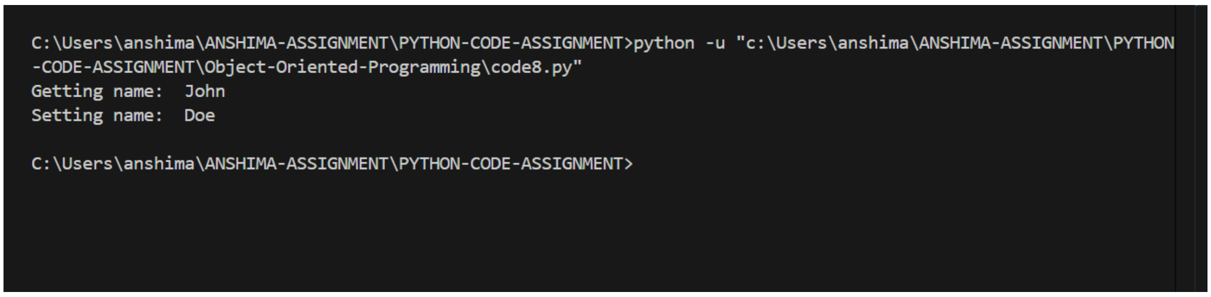


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Object-Oriented-Programming\code7.py"
This is the child class (overridden)

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

8. Create a class with private attributes and getter/setter methods.

```
class Person:
    def __init__(self, name):
        self.__name = name  # private attribute

    def get_name(self):
        return self.__name

    def set_name(self, name):
        if name:
            self.__name = name

p = Person("John")
print("Getting name: ",p.get_name())
p.set_name("Doe")
```

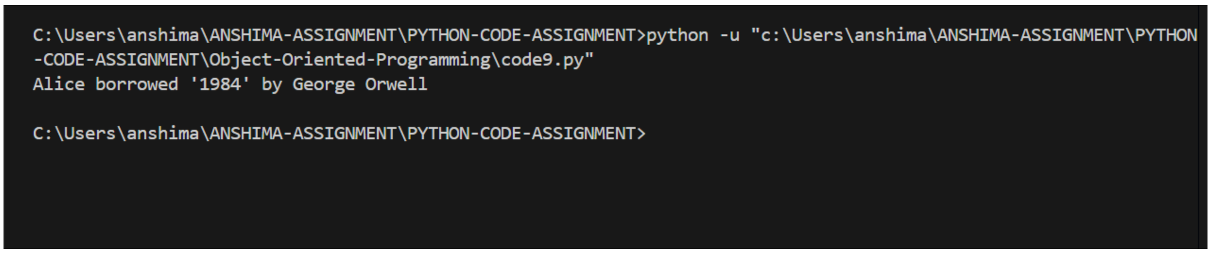
```
print("Setting name: ",p.get_name())
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Object-Oriented-Programming\code8.py"
Getting name:  John
Setting name:  Doe

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

9. Design a class structure for a library system with books and members.

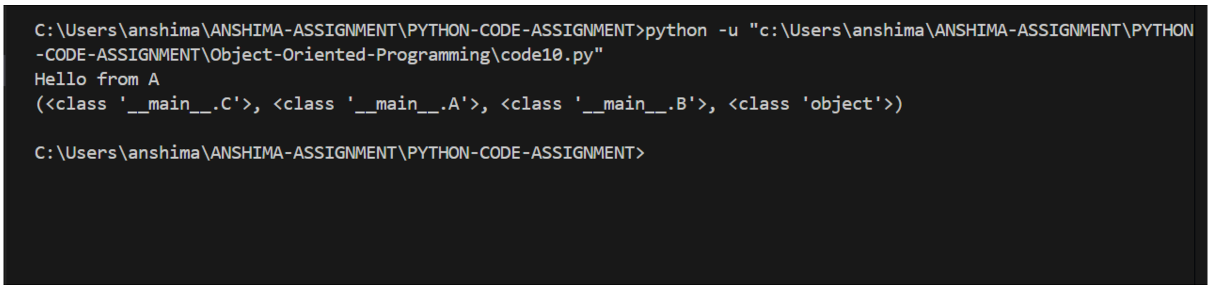
```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author

class Member:
    def __init__(self, name):
        self.name = name
        self.borrowed_books = []

    def borrow_book(self, book):
        self.borrowed_books.append(book)

book1 = Book("1984", "George Orwell")
member1 = Member("Alice")
member1.borrow_book(book1)

for book in member1.borrowed_books:
    print(f"{member1.name} borrowed '{book.title}' by {book.author}")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Object-Oriented-Programming\code9.py"
Alice borrowed '1984' by George Orwell

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

10. Implement multiple inheritance and demonstrate method resolution order.

```
class A:
    def greet(self):
        print("Hello from A")

class B:
    def greet(self):
        print("Hello from B")

class C(A, B):
    pass

obj = C()
obj.greet()
print(C.__mro__)
```

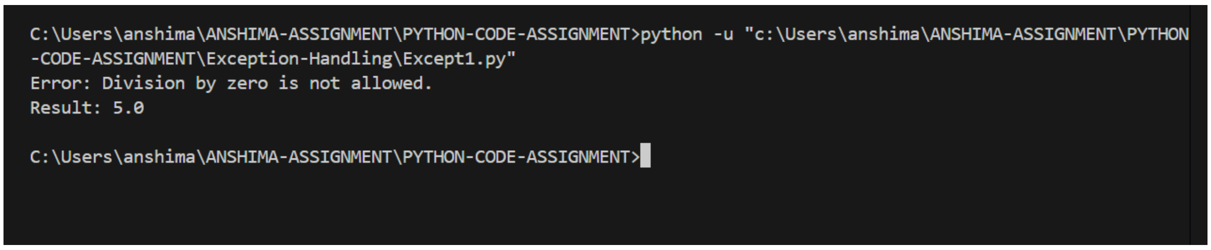


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Object-Oriented-Programming\code10.py"
Hello from A
(<class '__main__.C'>, <class '__main__.A'>, <class '__main__.B'>, <class 'object'>)

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

EXCEPTION HANDLING

1. Write a script that handles division by zero using try/except.

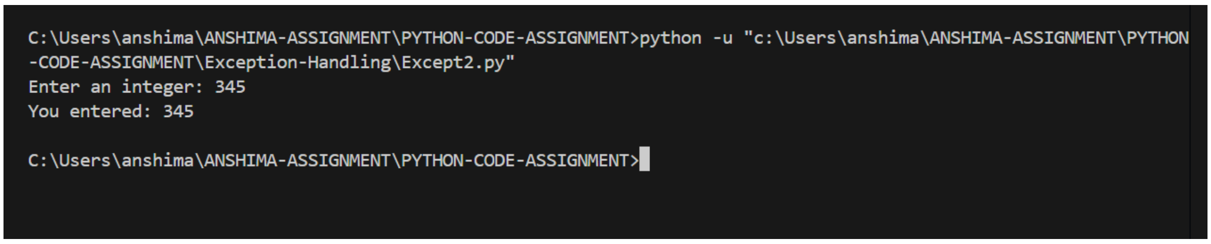
```
def divide(a, b):
    try:
        result = a / b
    except ZeroDivisionError:
        print("Error: Division by zero is not allowed.")
    else:
        print("Result:", result)

divide(10, 0)
divide(10, 2)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Exception-Handling\Except1.py"
Error: Division by zero is not allowed.
Result: 5.0

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

2. Create a program that handles invalid input from the user.

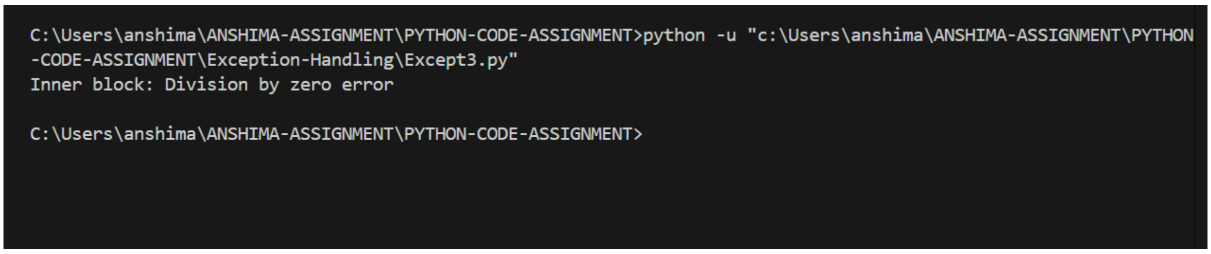
```
def get_integer():
    try:
        num = int(input("Enter an integer: "))
        print("You entered:", num)
    except ValueError:
        print("Invalid input! Please enter a valid integer.")

get_integer()
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Exception-Handling\Except2.py"
Enter an integer: 345
You entered: 345

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

3. Implement nested try/except blocks to handle multiple error types.

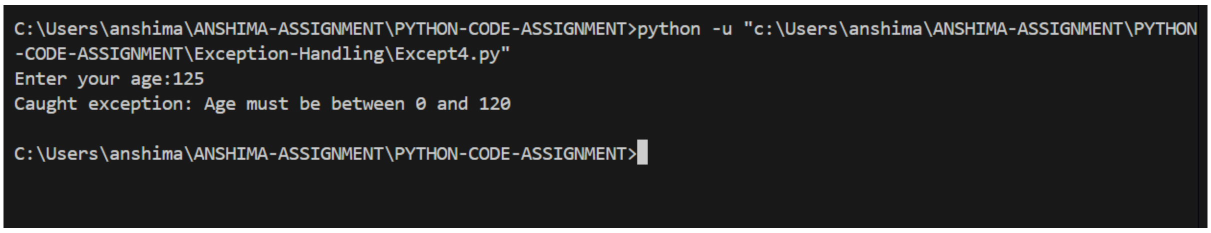
```
def nested_try():
    try:
        x = int("100")
        try:
            y = 10 / 0
        except ZeroDivisionError:
            print("Inner block: Division by zero error")
    except ValueError:
        print("Outer block: Value error")

nested_try()
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Exception-Handling\Except3.py"
Inner block: Division by zero error

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

4. Develop a function that raises a custom exception for invalid age input.

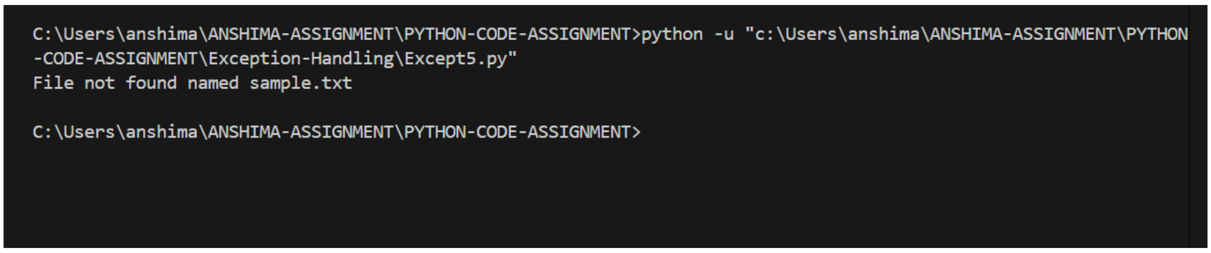
```
class InvalidAgeError(Exception):
    pass

def validate_age(age):
    if age < 0 or age > 120:
        raise InvalidAgeError("Age must be between 0 and 120")
    else:
        print("Valid age:", age)

age=int(input("Enter your age:"))
try:
    validate_age(age)
except InvalidAgeError as e:
    print("Caught exception:", e)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Exception-Handling\Except4.py"
Enter your age:125
Caught exception: Age must be between 0 and 120

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

5. Write a script using try/except/finally to open and safely close a file.

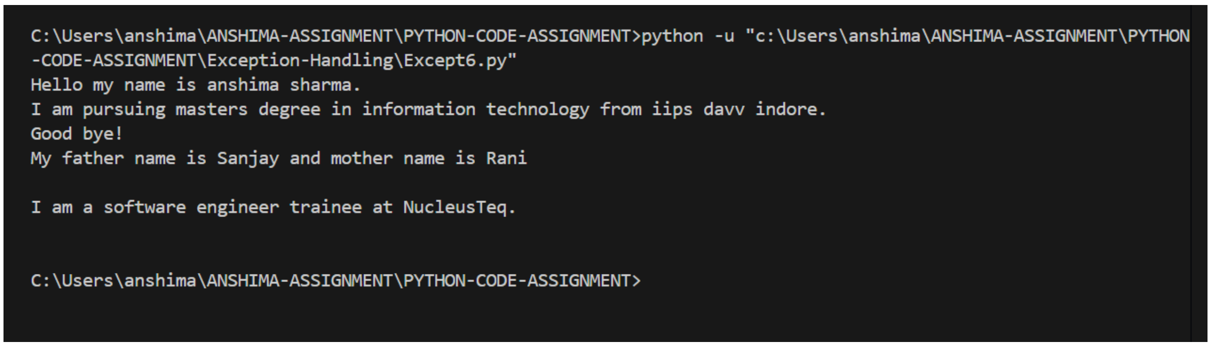
```
def read_file(filename):
    try:
        file = open(filename, "r")
        print(file.read())
    except FileNotFoundError:
        print("File not found named",filename)
    finally:
        try:
            file.close()
        except:
            pass
read_file("sample.txt")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Exception-Handling\Except5.py"
File not found named sample.txt

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

6. Create a context manager using a class to handle file open/close with exception support.

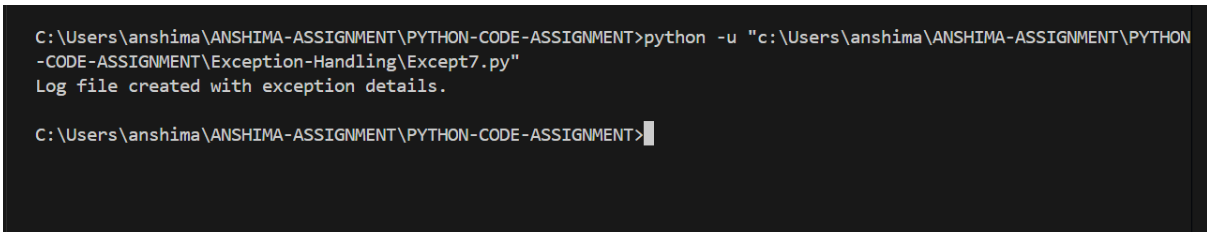
```
 class FileManager:
   def __init__(self, filename, mode):
      self.filename = filename
      self.mode = mode

   def __enter__(self):
      self.file = open(self.filename, self.mode)
      return self.file

   def __exit__(self, exc_type, exc_val, exc_tb):
      if self.file:
         self.file.close()

with FileManager("merged.txt", "r") as f:
   print(f.read())
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Exception-Handling\Except6.py"
Hello my name is anshima sharma.
I am pursuing masters degree in information technology from iips davv indore.
Good bye!
My father name is Sanjay and mother name is Rani

I am a software engineer trainee at NucleusTeq.


C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

7. Write a program that logs exceptions to a file.

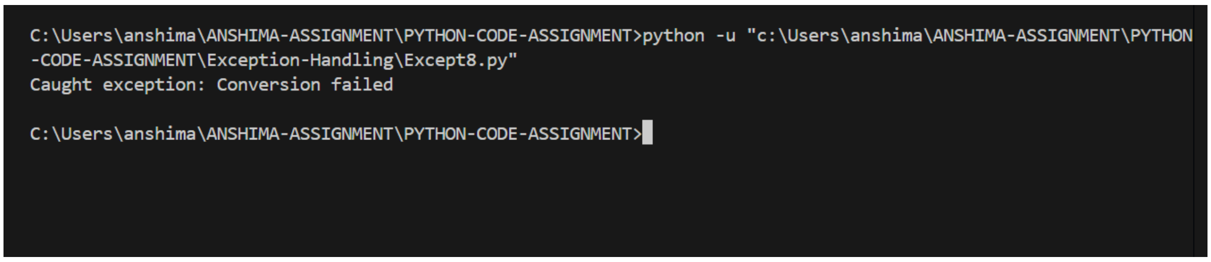
```
 def log_exceptions():
   try:
      x = 10 / 0
   except Exception as e:
      with open("log.txt", "a") as file:
         file.write(f"Exception occurred: {e}\n")
         print("Log file created with exception details.")

log_exceptions()
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Exception-Handling\Except7.py"
Log file created with exception details.

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

8. Implement exception chaining using raise from syntax.

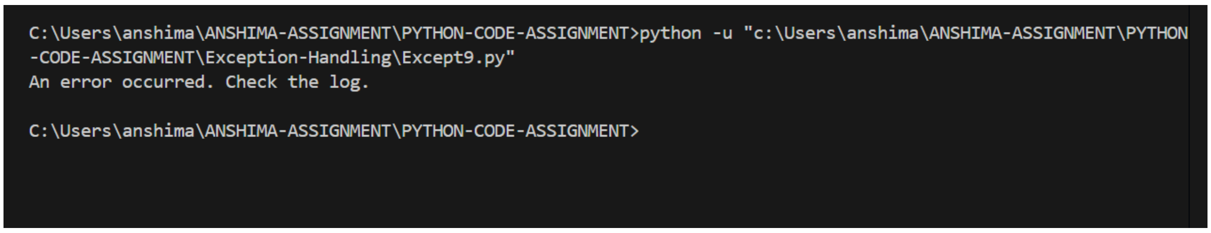
```
def convert_input(data):
    try:
        return int(data)
    except ValueError as e:
        raise RuntimeError("Conversion failed") from e

try:
    convert_input("abc")
except RuntimeError as e:
    print("Caught exception:", e)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Exception-Handling\Except8.py"
Caught exception: Conversion failed

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

9. Develop a decorator that catches and logs exceptions in any function.

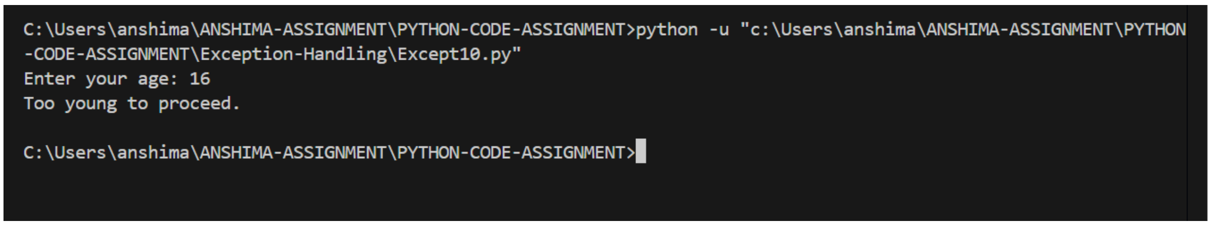
```
                                        def exception_logger(func):
    def wrapper(*args, **kwargs):
        try:
            return func(*args, **kwargs)
        except Exception as e:
            with open("decorator_log.txt", "a") as f:
                f.write(f"Error in {func.__name__}: {e}\n")
            print("An error occurred. Check the log.")
    return wrapper

@exception_logger
def risky_division(x, y):
    return x / y

risky_division(10, 0)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Exception-Handling\Except9.py"
An error occurred. Check the log.

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

10. Define multiple custom exceptions and handle them in different scenarios.

```
class TooYoungError(Exception): pass
class TooOldError(Exception): pass

def check_age(age):
    if age < 18:
        raise TooYoungError("Too young to proceed.")
    elif age > 60:
        raise TooOldError("Too old to proceed.")
    else:
        print("Age is appropriate.")

age=int(input("Enter your age: "))
try:
    check_age(age)
except TooYoungError as e:
    print(e)
except TooOldError as e:
    print(e)
```

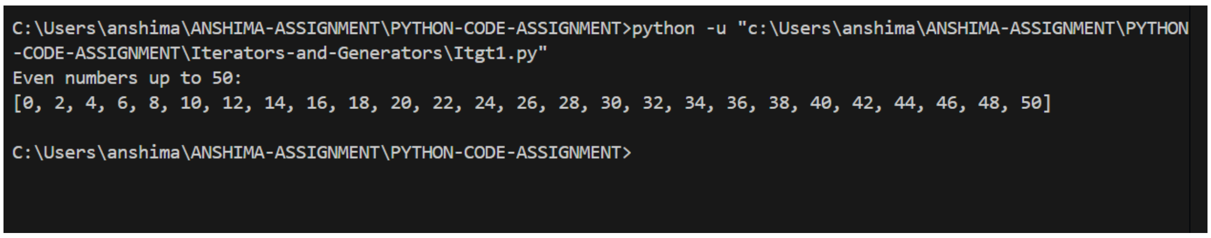


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Exception-Handling\Except10.py"
Enter your age: 16
Too young to proceed.

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

ITERATORS AND GENERATORS

1. Create a generator that yields even numbers up to a given number.

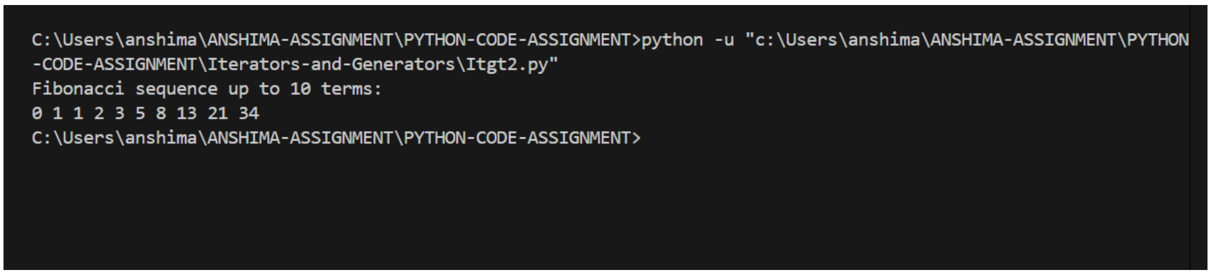
```
def even_numbers(limit):
    for num in range(0, limit + 1, 2):
        yield num

print("Even numbers up to 50:")
print(list(even_numbers(50)))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Iterators-and-Generators\Itgt1.py"
Even numbers up to 50:
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

2. Write an iterator class that returns Fibonacci numbers up to n terms.

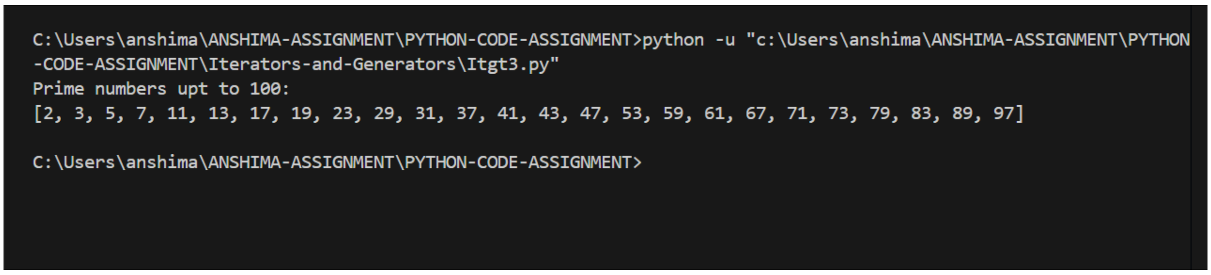
```
class FibonacciIterator:
    def __init__(self, n):
        self.n = n
        self.a, self.b = 0, 1
        self.count = 0

    def __iter__(self):
        return self

    def __next__(self):
        if self.count >= self.n:
            raise StopIteration
        self.count += 1
        result = self.a
        self.a, self.b = self.b, self.a + self.b
        return result

print("Fibonacci sequence up to 10 terms:")
for num in FibonacciIterator(10):
    print(num, end=' ')
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Iterators-and-Generators\Itgt2.py"
Fibonacci sequence up to 10 terms:
0 1 1 2 3 5 8 13 21 34
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

3. Develop a generator that yields the prime numbers under 100.

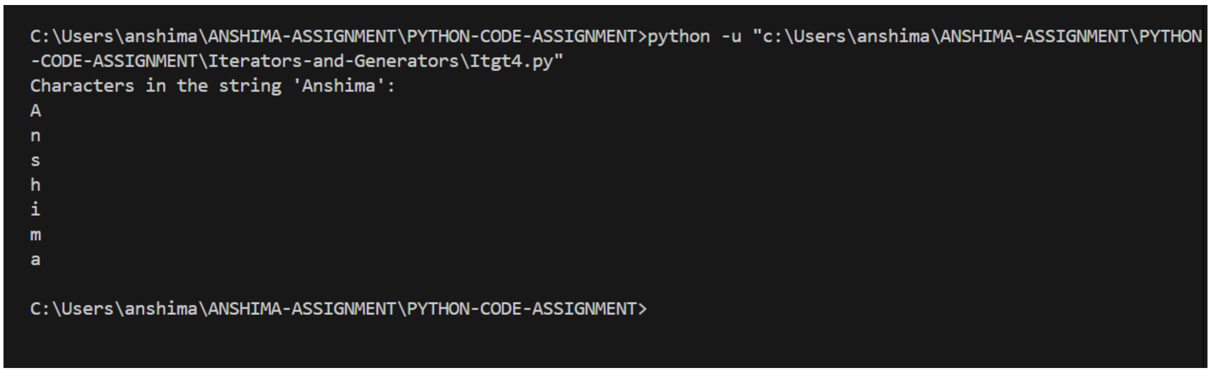
```
def is_prime(n):
   if n < 2:
      return False
   for i in range(2, int(n**0.5)+1):
      if n % i == 0:
         return False
   return True

def prime_gen():
   for i in range(2, 100):
      if is_prime(i):
         yield i

print("Prime numbers upt to 100:")
print(list(prime_gen()))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Iterators-and-Generators\Itgt3.py"
Prime numbers upt to 100:
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

4. Implement a custom iterable class for iterating characters of a string.

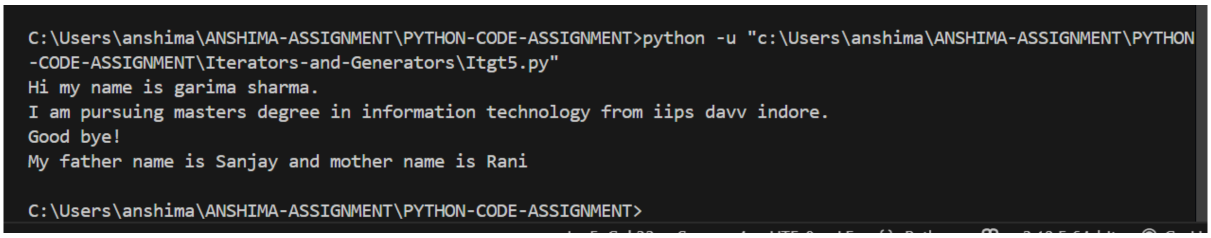
```
class CharIterator:
   def __init__(self, string):
      self.string = string
      self.index = 0

   def __iter__(self):
      return self

   def __next__(self):
      if self.index >= len(self.string):
         raise StopIteration
      ch = self.string[self.index]
      self.index += 1
      return ch

print("Characters in the string 'Anshima':")
for char in CharIterator("Anshim"):
   print(char)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Iterators-and-Generators\Itgt4.py"
Characters in the string 'Anshima':
A
n
s
h
i
m
a

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

5. Write a generator to yield lines from a file one by one.

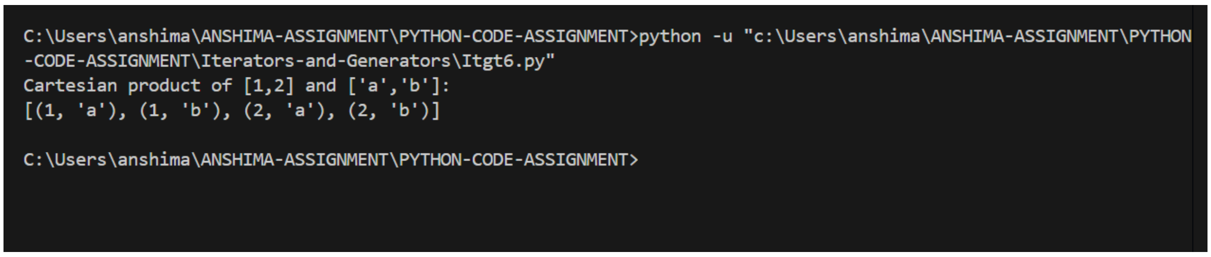
```
 def line_reader(filename):
    with open(filename, 'r') as file:
        for line in file:
            yield line.strip()
for line in line_reader('anshima.txt'):
    print(line)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Iterators-and-Generators\Itgt5.py"
Hi my name is garima sharma.
I am pursuing masters degree in information technology from iips davv indore.
Good bye!
My father name is Sanjay and mother name is Rani

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

6. Create a nested generator to yield Cartesian product of two lists.

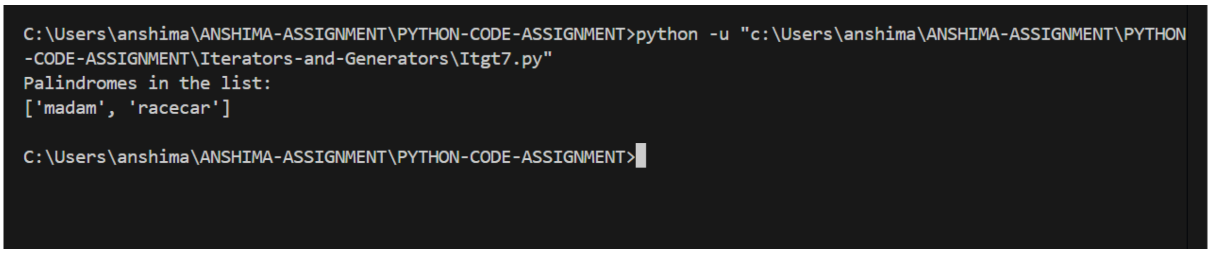
```
def cartesian_product(a, b):
    for x in a:
        for y in b:
            yield (x, y)

print("Cartesian product of [1,2] and ['a','b']:")
print(list(cartesian_product([1, 2], ['a', 'b'])))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Iterators-and-Generators\Itgt6.py"
Cartesian product of [1,2] and ['a','b']:
[(1, 'a'), (1, 'b'), (2, 'a'), (2, 'b')]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

7. Write a generator expression to filter out palindromes from a list of words.

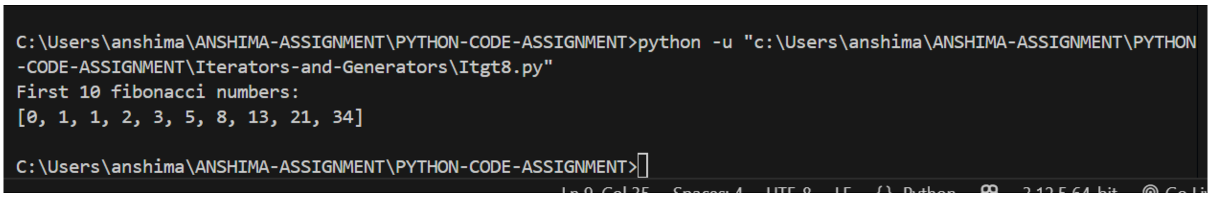
```
words = ['madam', 'hello', 'racecar', 'world']
palindromes = (word for word in words if word == word[::-1])
print("Palindromes in the list:")
print(list(palindromes))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Iterators-and-Generators\Itgt7.py"
Palindromes in the list:
['madam', 'racecar']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

8. Implement an infinite generator for Fibonacci numbers.

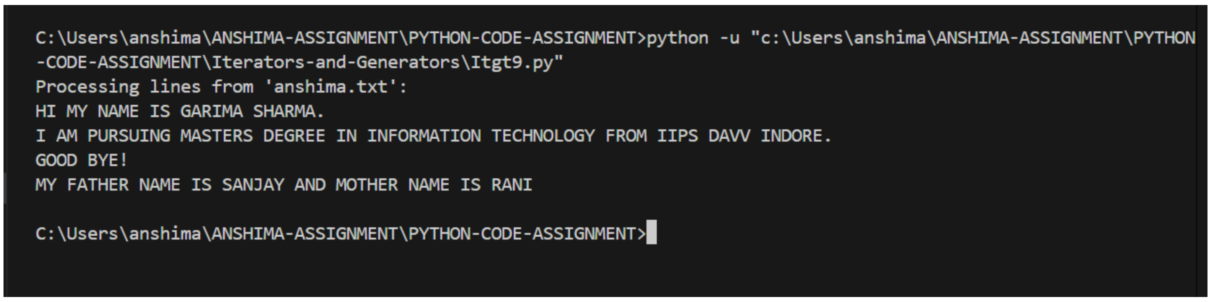
```
def infinite_fibonacci():
    a, b = 0, 1
    while True:
        yield a
        a, b = b, a + b

from itertools import islice
print("First 10 fibonacci numbers:")
print(list(islice(infinite_fibonacci(), 10)))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Iterators-and-Generators\Itgt8.py"
First 10 fibonacci numbers:
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

9. Develop a program using generator pipelines to process a text file.

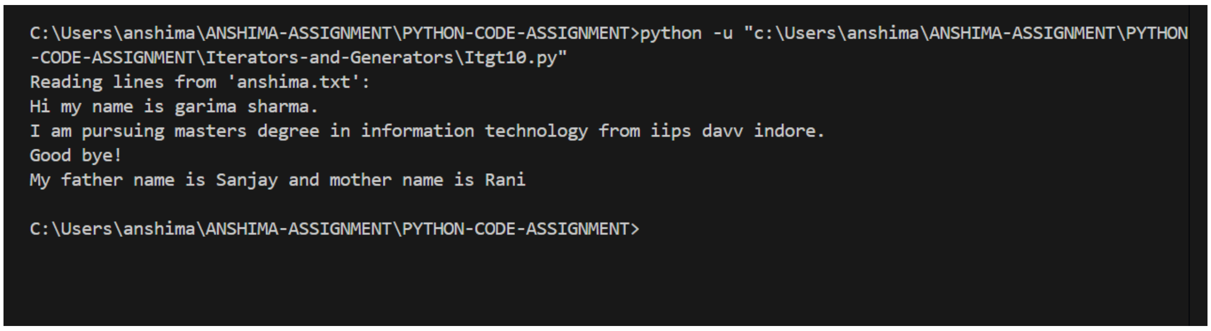
```
def read_lines(file):
    with open(file) as f:
        for line in f:
            yield line.strip()

def filter_lines(lines):
    return (line for line in lines if line)

def uppercase_lines(lines):
    return (line.upper() for line in lines)

print("Processing lines from")
pipeline = uppercase_lines(filter_lines(read_lines('anshima.txt')))
for line in pipeline:
    print(line)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Iterators-and-Generators\Itgt9.py"
Processing lines from 'anshima.txt':
HI MY NAME IS GARIMA SHARMA.
I AM PURSUING MASTERS DEGREE IN INFORMATION TECHNOLOGY FROM IIPS DAVV INDORE.
GOOD BYE!
MY FATHER NAME IS SANJAY AND MOTHER NAME IS RANI

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

10. Create a context manager that uses a generator to manage a resource.

```
from contextlib import contextmanager

@contextmanager
def open_file(name):
    f = open(name, 'r')
    try:
        yield f
    finally:
        f.close()

print("Reading lines from 'anshima.txt':")

with open_file('anshima.txt') as file:
    for line in file:
        print(line.strip())
```

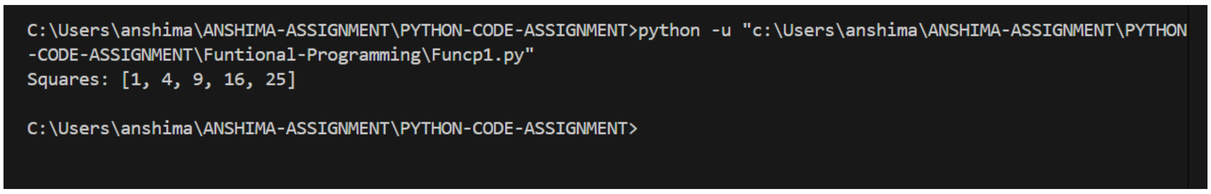


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Iterators-and-Generators\Itgt10.py"
Reading lines from 'anshima.txt':
Hi my name is garima sharma.
I am pursuing masters degree in information technology from iips davv indore.
Good bye!
My father name is Sanjay and mother name is Rani

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

FUNCTIONAL PROGRAMMING

1. Write a lambda function to compute the square of a number and use it in a list comprehension.

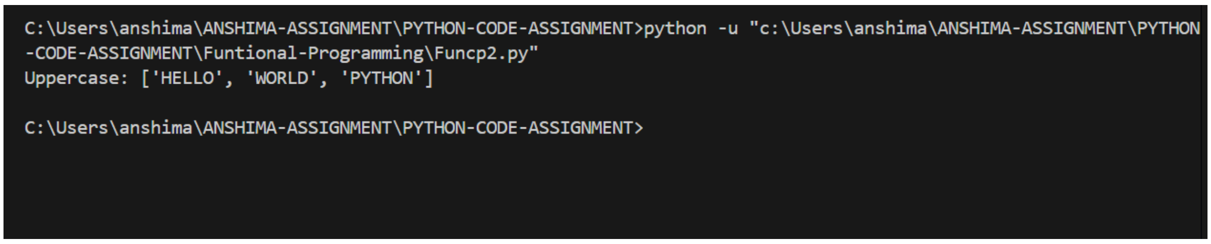
```
square = lambda x: x ** 2
squares = [square(x) for x in range(1, 6)]
print("Squares:", squares)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Funtional-Programming\Funcp1.py"
Squares: [1, 4, 9, 16, 25]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

2. Use `map()` to convert all strings in a list to uppercase.

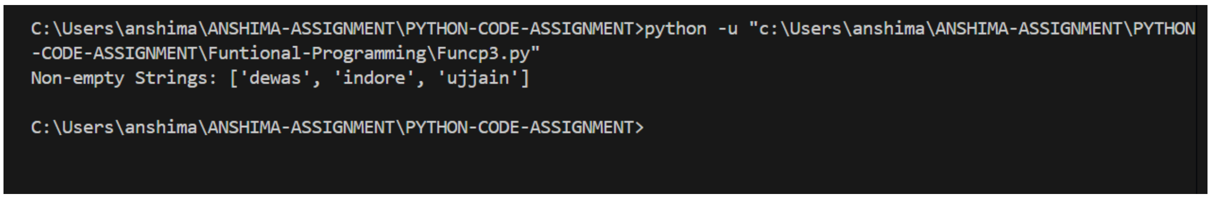
```
words = ['hello', 'world', 'python']
upper_case = list(map(str.upper, words))
print("Uppercase:", upper_case)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Funtional-Programming\Funcp2.py"
Uppercase: ['HELLO', 'WORLD', 'PYTHON']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

3. Use `filter()` to remove empty strings from a list.

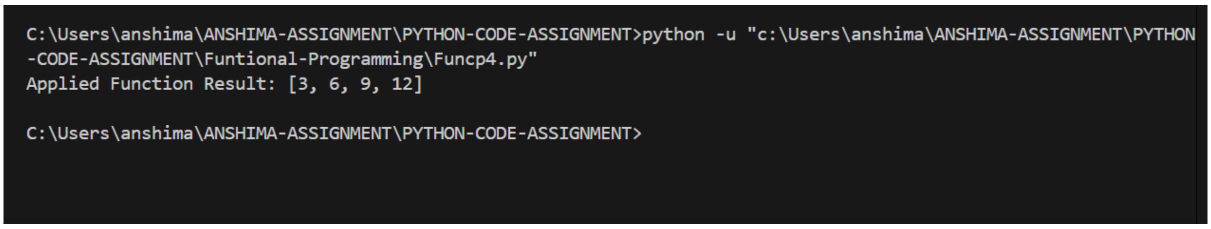
```
cities = ['dewas', '', 'indore', '', 'ujjain']
non_empty = list(filter(None, cities))
print("Non-empty Strings:", non_empty)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Funtional-Programming\Funcp3.py"
Non-empty Strings: ['dewas', 'indore', 'ujjain']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

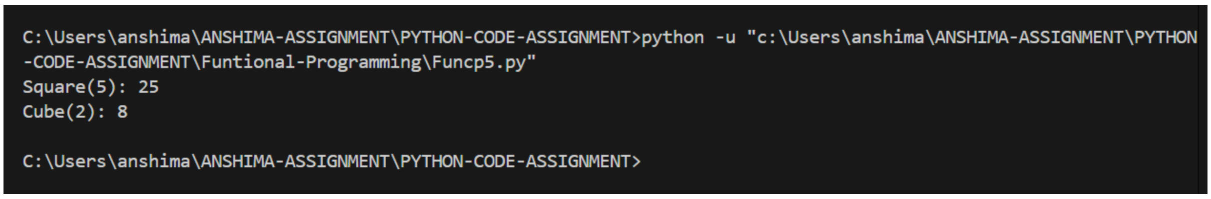
4. Write a function that takes a function and a list, and applies it to all items.

```
def apply_func(func, lst):
    return [func(x) for x in lst]

result = apply_func(lambda x: x * 3, [1, 2, 3, 4])
print("Applied Function Result:", result)
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Funtional-Programming\Funcp4.py"
Applied Function Result: [3, 6, 9, 12]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

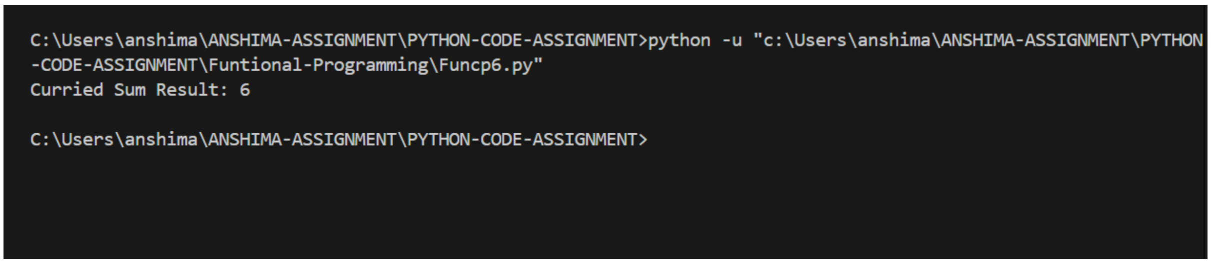
5. Create a higher-order function that returns a power function.

```
def power_fn(exp):
    return lambda x: x ** exp

square = power_fn(2)
cube = power_fn(3)
print("Square(5):", square(5))
print("Cube(2):", cube(2))
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Funtional-Programming\Funcp5.py"
Square(5): 25
Cube(2): 8

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

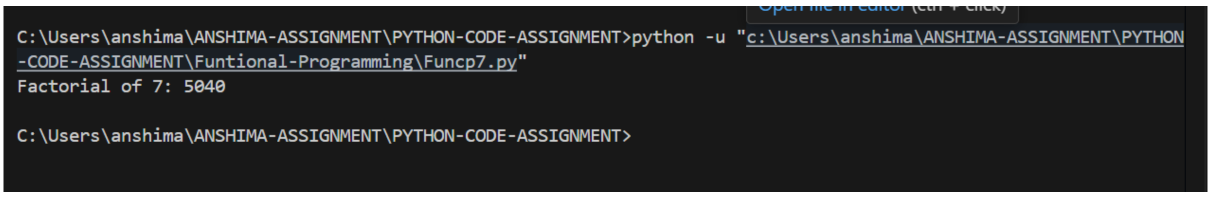
6. Implement currying using closures in Python.

```
def curried_sum(x):
    def add_y(y):
        def add_z(z):
            return x + y + z
        return add_z
    return add_y

result = curried_sum(1)(2)(3)
print("Curried Sum Result:", result)
```


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Funtional-Programming\Funcp6.py"
Curried Sum Result: 6

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

7. Use `reduce()` to calculate the factorial of a number.

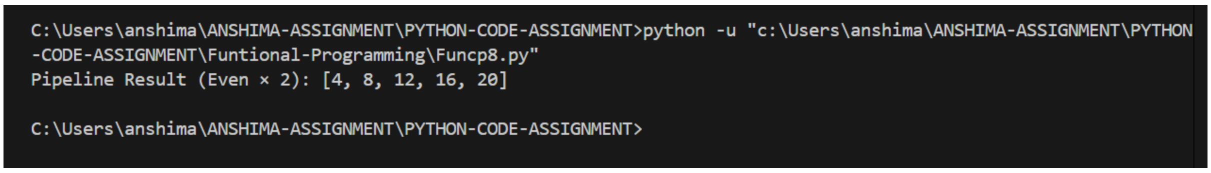
```
from functools import reduce

factorial = lambda n: reduce(lambda x, y: x * y, range(1, n + 1))
print("Factorial of 7:", factorial(7))
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Funtional-Programming\Funcp7.py"
Factorial of 7: 5040

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

8. Develop a function that uses both `map()` and `filter()` in a pipeline.

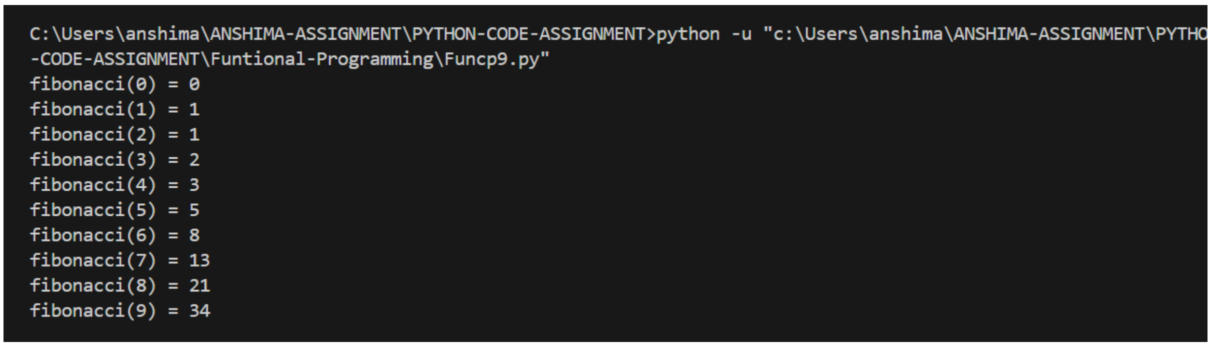
```
numbers = range(1, 11)
pipeline = list(map(lambda x: x * 2, filter(lambda x: x % 2 == 0, numbers)))
print("Pipeline Result (Even × 2):", pipeline)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Funtional-Programming\Funcp8.py"
Pipeline Result (Even × 2): [4, 8, 12, 16, 20]

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

9. Create a decorator that memoizes the result of a function.

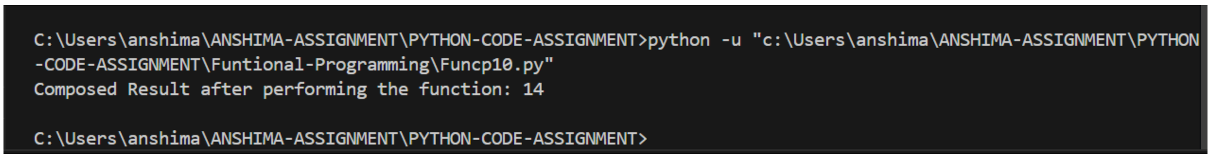
```
def memoize(func):
   cache = {}
   def wrapper(n):
      if n in cache:
         return cache[n]
      result = func(n)
      cache[n] = result
      return result
   return wrapper

@memoize
def fibonacci(n):
   if n <= 1:
      return n
   return fibonacci(n - 1) + fibonacci(n - 2)

for i in range(10):
   print(f"fibonacci({i}) = {fibonacci(i)}")
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHO
-CODE-ASSIGNMENT\Funtional-Programming\Funcp9.py"
fibonacci(0) = 0
fibonacci(1) = 1
fibonacci(2) = 1
fibonacci(3) = 2
fibonacci(4) = 3
fibonacci(5) = 5
fibonacci(6) = 8
fibonacci(7) = 13
fibonacci(8) = 21
fibonacci(9) = 34
```

10. Implement a generic compose() function that chains multiple functions.

```
def compose(*funcs):
    def composed(x):
        for f in reversed(funcs):
            x = f(x)
        return x
    return composed

add2 = lambda x: x + 2
times3 = lambda x: x * 3

composed_func = compose(add2, times3)  # (x * 3) + 2
print("Composed Result after performing the function:", composed_func(4))  # (4 * 3) + 2 =
14
```

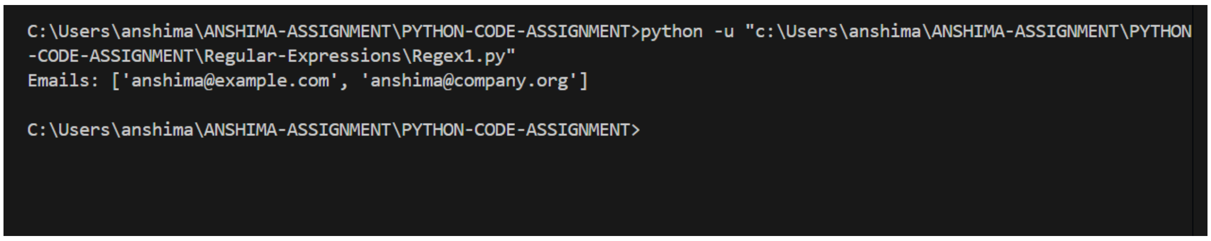


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Funtional-Programming\Funcp10.py"
Composed Result after performing the function: 14

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

REGULAR EXPRESSIONS

1. Write a regex to extract all email addresses from a string.

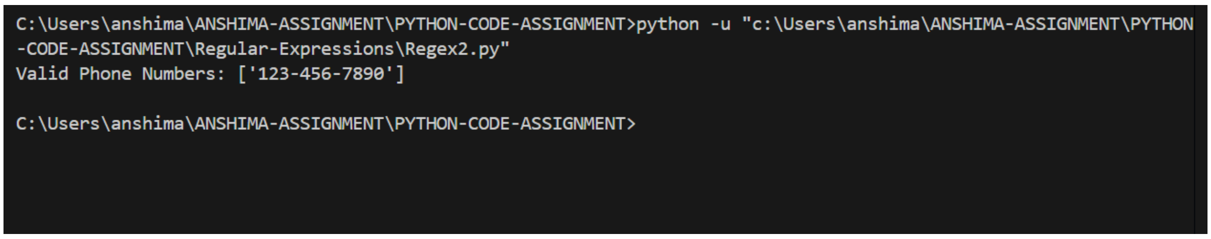
```
import re
text = "Reach out to us at anshima@example.com and anshima@company.org"
emails = re.findall(r'\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b', text)
print("Emails:", emails)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Regular-Expressions\Regex1.py"
Emails: ['anshima@example.com', 'anshima@company.org']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

2. Use regex to validate a phone number format (e.g., 123-456-7890).

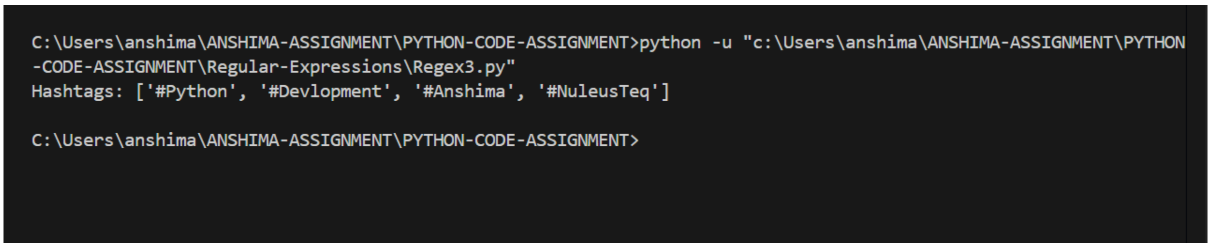
```
import re
text = "Valid: 123-456-7890, Invalid: 1234567890"
phones = re.findall(r'\b\d{3}-\d{3}-\d{4}\b', text)
print("Valid Phone Numbers:", phones)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Regular-Expressions\Regex2.py"
Valid Phone Numbers: ['123-456-7890']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

3. Create a script to extract all hashtags from a given text.

```
import re
text = "Example String #Python #Devlopment #Anshima #NuleusTeq"
hashtags = re.findall(r'#\w+', text)
print("Hashtags:", hashtags)
```

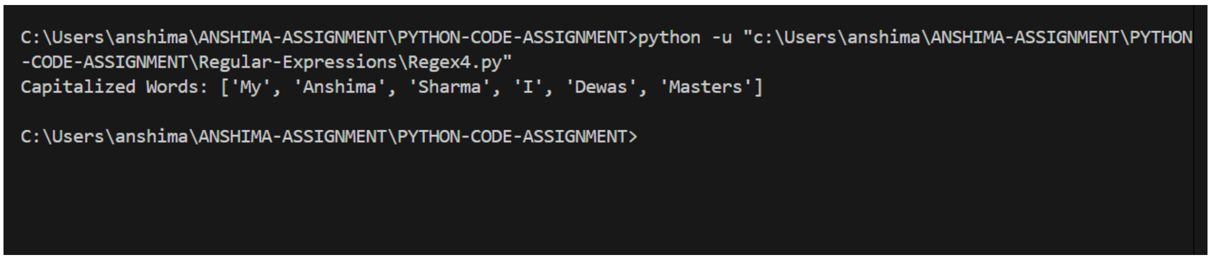


```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Regular-Expressions\Regex3.py"
Hashtags: ['#Python', '#Devlopment', '#Anshima', '#NuleusTeq']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

4. Use regex to find all words starting with a capital letter.

```
import re
```

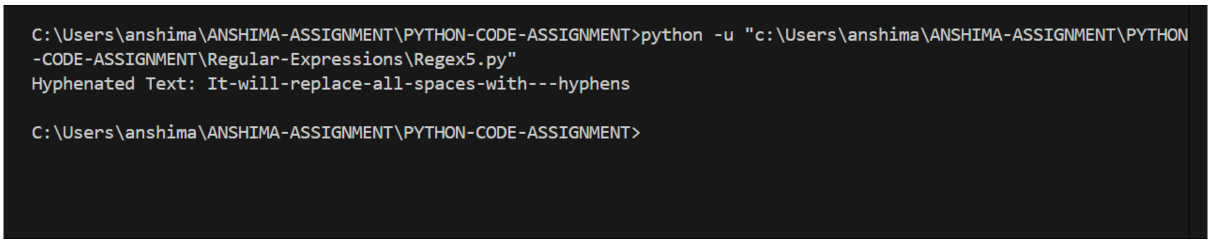
```
text = "My name is Anshima Sharma and I live in Dewas and currently pursuing Masters degree."
capital_words = re.findall(r'\b[A-Z][a-z]*\b', text)
print("Capitalized Words:", capital_words)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Regular-Expressions\Regex4.py"
Capitalized Words: ['My', 'Anshima', 'Sharma', 'I', 'Dewas', 'Masters']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

5. Write a regex to replace all whitespace with hyphens.

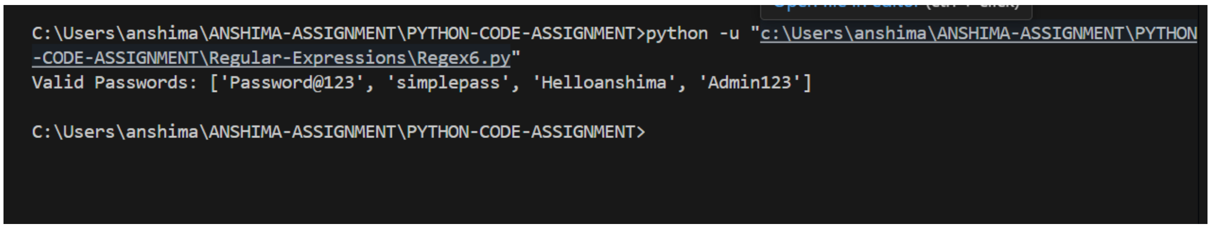
```
import re
text = "It will replace all spaces with -  hyphens"
modified = re.sub(r'\s+', '-', text)
print("Hyphenated Text:", modified)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Regular-Expressions\Regex5.py"
Hyphenated Text: It-will-replace-all-spaces-with---hyphens

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

6. Build a regex to validate complex passwords (at least 1 digit, 1 symbol, 8+ chars).

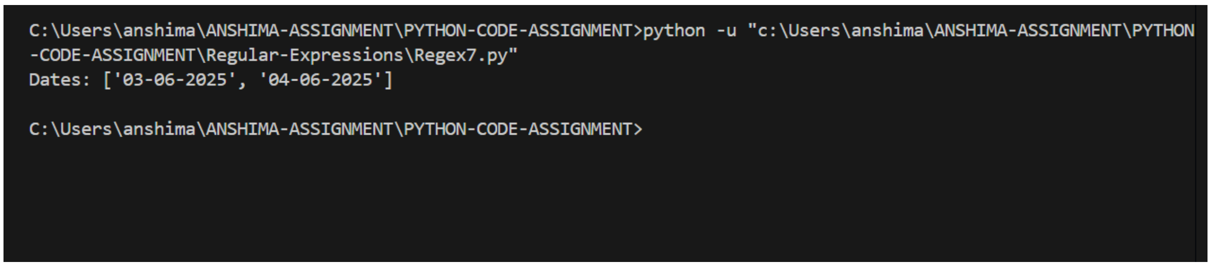
```
import re
text = "Password@123, simplepass, Helloanshima!, Admin123#"
passwords = re.findall(r'\b(?=.*\d)(?=.*[!@#$%^&*])(?=\S{8,})\S+\b', text)
print("Valid Passwords:", passwords)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Regular-Expressions\Regex6.py"
Valid Passwords: ['Password@123', 'simplepass', 'Helloanshima', 'Admin123']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

7. Write a script using regex to extract dates in dd-mm-yyyy format.

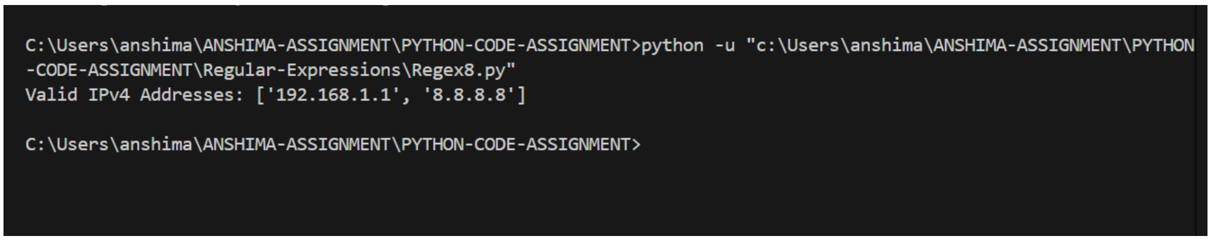
```
import re
text = "Today's date is 03-06-2025 and tomorrow is 04-06-2025"
dates = re.findall(r'\b\d{2}-\d{2}-\d{4}\b', text)
print("Dates:", dates)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Regular-Expressions\Regex7.py"
Dates: ['03-06-2025', '04-06-2025']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

8. Create a regex pattern that identifies valid IPv4 addresses.

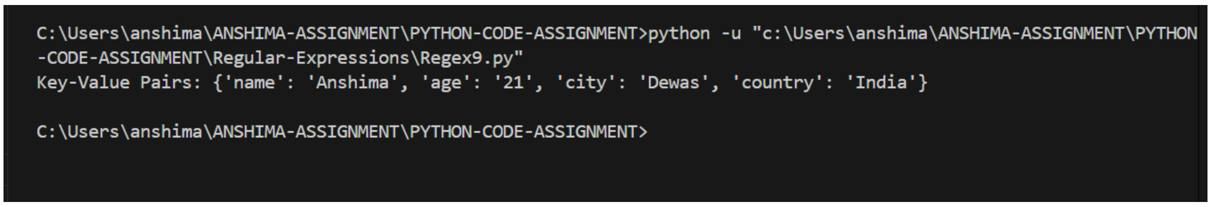
```
import re
text = "Valid: 192.168.1.1, 8.8.8.8 | Invalid: 256.300.8.1"
ip_matches = re.findall(r'\b(?:\d{1,3}\.){3}\d{1,3}\b', text)
valid_ips = [ip for ip in ip_matches if all(0 <= int(part) <= 255 for part in ip.split('.'))]
print("Valid IPv4 Addresses:", valid_ips)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Regular-Expressions\Regex8.py"
Valid IPv4 Addresses: ['192.168.1.1', '8.8.8.8']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

9. Use regex to parse and extract key-value pairs from a query string.

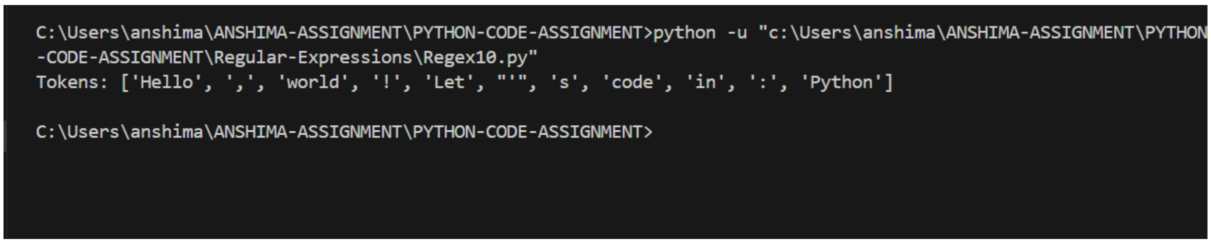
```
import re
query = "name=Anshima & age=21 & city=Dewas & country=India"
kv_pairs = dict(re.findall(r'(\w+)=([\w%]+)', query))
print("Key-Value Pairs:", kv_pairs)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Regular-Expressions\Regex9.py"
Key-Value Pairs: {'name': 'Anshima', 'age': '21', 'city': 'Dewas', 'country': 'India'}

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

10. Implement a tokenizer using regular expressions that separates punctuation from words.

```
import re
text = "Hello, world! Let's code in : Python"
tokens = re.findall(r'\b\w+\b|[^\w\s]', text)
print("Tokens:", tokens)
```



```
C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>python -u "c:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON
-CODE-ASSIGNMENT\Regular-Expressions\Regex10.py"
Tokens: ['Hello', ',', 'world', '!', 'Let', "'", 's', 'code', 'in', ':', 'Python']

C:\Users\anshima\ANSHIMA-ASSIGNMENT\PYTHON-CODE-ASSIGNMENT>
```

Thankyou!