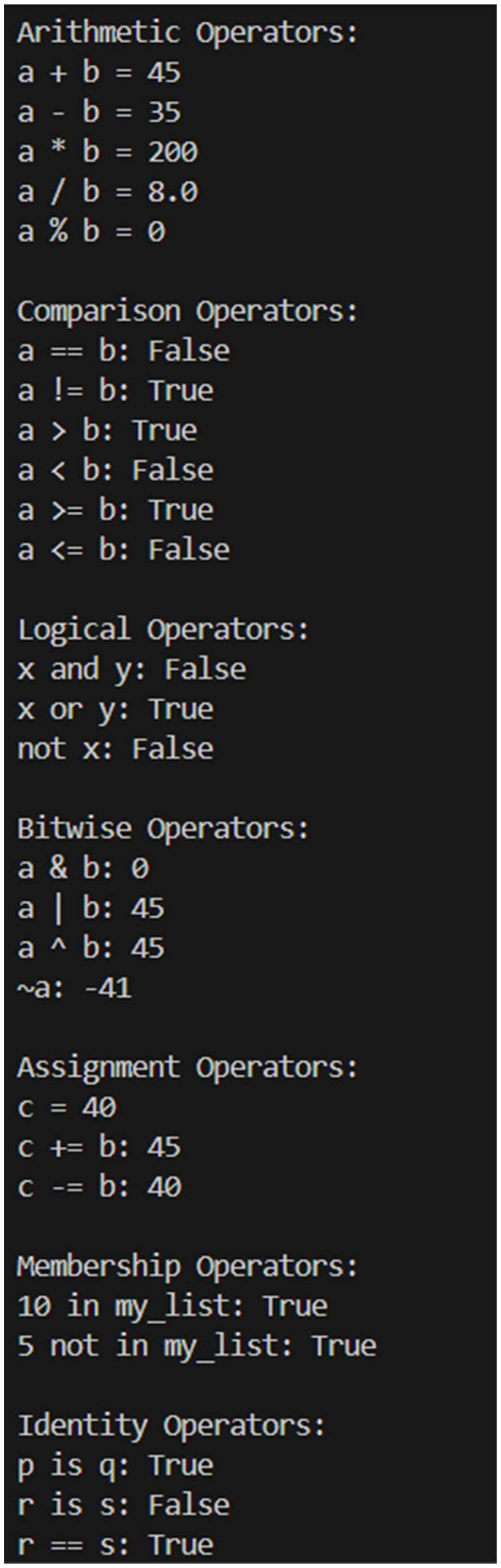
Output 1 :-



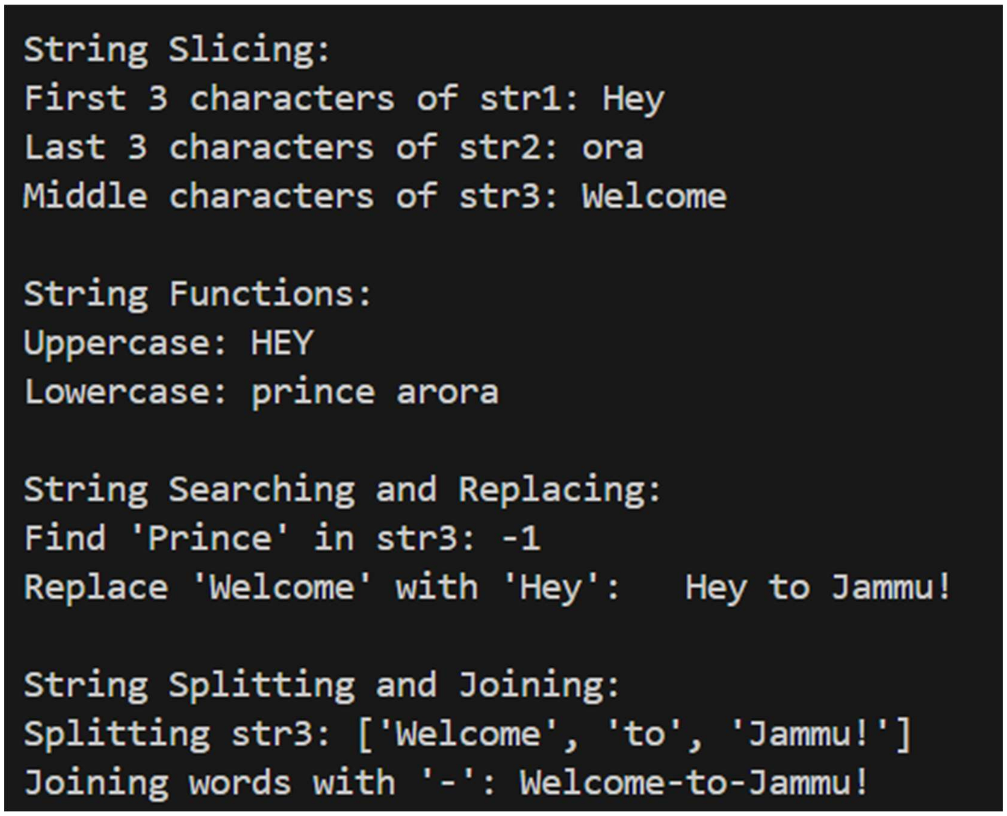
```
Arithmetic Operators:
a + b = 45
a - b = 35
a * b = 200
a / b = 8.0
a % b = 0

Comparison Operators:
a == b: False
a != b: True
a > b: True
a < b: False
a >= b: True
a <= b: False

Logical Operators:
x and y: False
x or y: True
not x: False

Bitwise Operators:
a & b: 0
a | b: 45
a ^ b: 45
~a: -41

Assignment Operators:
c = 40
c += b: 45
c -= b: 40

Membership Operators:
10 in my_list: True
5 not in my_list: True

Identity Operators:
p is q: True
r is s: False
r == s: True
```

1. **Write a program to demonstrate different types of operators.**

```
a = 20
b = 4
print("Arithmetic Operators:")
print("a + b =", a + b)
print("a - b =", a - b)
print("a * b =", a * b)
print("a / b =", a / b)
print("a % b =", a % b)

print("\nComparison Operators:")
print("a == b:", a == b)
print("a != b:", a != b)
print("a > b:", a > b)
print("a < b:", a < b)
print("a >= b:", a >= b)
print("a <= b:", a <= b)

print("\nLogical Operators:")
x = True
y = False
print("x and y:", x and y)
print("x or y:", x or y)
print("not x:", not x)

print("\nBitwise Operators:")
print("a & b:", a & b)
print("a | b:", a | b)
print("a ^ b:", a ^ b)
print("~a:", ~a)

print("\nAssignment Operators:")
```

```
c = a
print("c =", c)
c += b
print("c += b:", c)
c -= b
print("c -= b:", c)

print("\nMembership Operators:")
my_list = [1, 2, 3, 10]
print("10 in my_list:", 10 in my_list)
print("5 not in my_list:", 5 not in my_list)

print("\nIdentity Operators:")
p = 5
q = 5
r = [1, 2, 3]
s = [1, 2, 3]
print("p is q:", p is q)
print("r is s:", r is s)
print("r == s:", r == s)
```

Output 2 :-

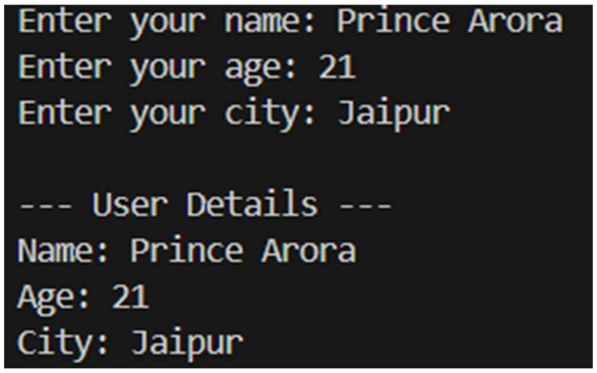


```
String Slicing:
First 3 characters of str1: Hey
Last 3 characters of str2: ora
Middle characters of str3: Welcome

String Functions:
Uppercase: HEY
Lowercase: prince arora

String Searching and Replacing:
Find 'Prince' in str3: -1
Replace 'Welcome' with 'Hey':   Hey to Jammu!

String Splitting and Joining:
Splitting str3: ['Welcome', 'to', 'Jammu!']
Joining words with '-': Welcome-to-Jammu!
```

2. **Write a program to demonstrate string operations and functions.**

```
str1 = "Hey"
str2 = "Prince Arora"
str3 = "Welcome to Jammu!"

print("\nString Slicing:")
print("First 3 characters of str1:", str1[:3])
print("Last 3 characters of str2:", str2[-3:])
print("Middle characters of str3:", str3[2:10])

print("\nString Functions:")
print("Uppercase:", str1.upper())
print("Lowercase:", str2.lower())

print("\nString Searching and Replacing:")
print("Find 'Prince' in str3:", str3.find("Prince"))
print("Replace 'Welcome' with 'Hey':", str3.replace("Welcome", "Hey"))

print("\nString Splitting and Joining:")
words = str3.split()
print("Splitting str3:", words)
joined_str = "-".join(words)
print("Joining words with '-':", joined_str)
```

Output 3 :-

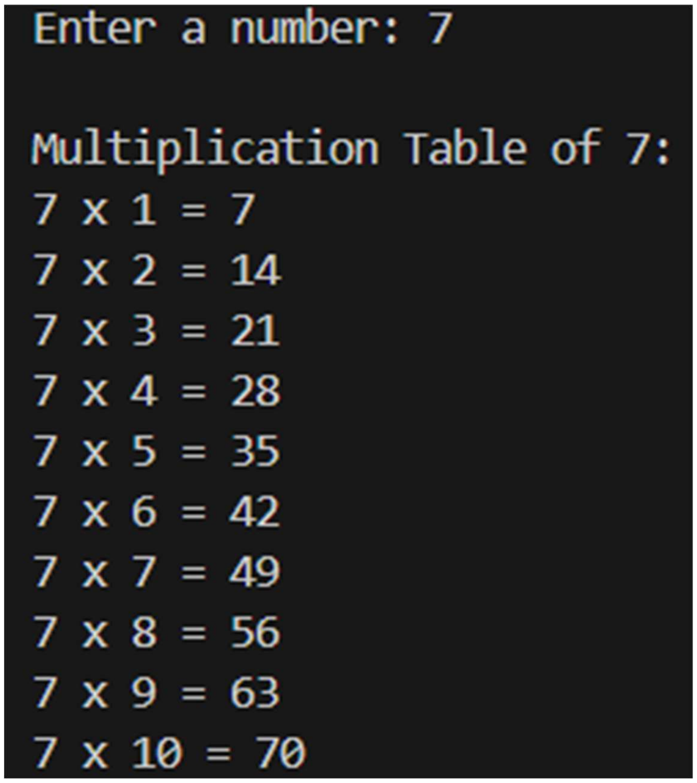


```
Enter your name: Prince Arora
Enter your age: 21
Enter your city: Jaipur

--- User Details ---
Name: Prince Arora
Age: 21
City: Jaipur
```

3. Write a program to take input from keyboard.

```
name = input("Enter your name: ")
age = int(input("Enter your age: "))
city = input("Enter your city: ")

print("\n--- User Details ---")
print("Name:", name)
print("Age:", age)
print("City:", city)
```

Output 4 :-

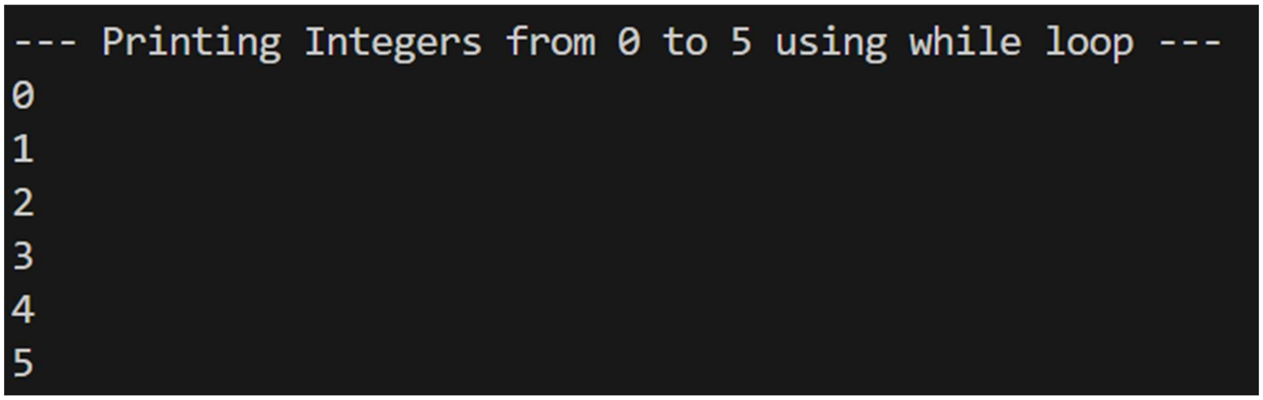


```
Enter a number: 7

Multiplication Table of 7:
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
```

4. Write a program to implement For loop.

```
num = int(input("Enter a number: "))
print(f"\nMultiplication Table of {num}:")
for i in range(1, 11):
    print(f"{num} x {i} = {num * i}")
```

Output 5 :-

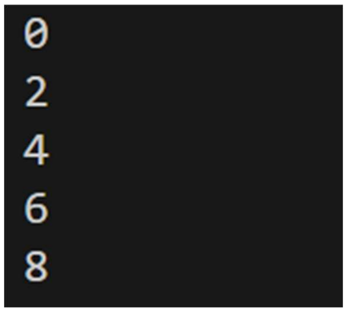


```
--- Printing Integers from 0 to 5 using while loop ---
0
1
2
3
4
5
```

5. Write a program to print integer from 0 to 5 using while loop.

```
print("\n--- Printing Integers from 0 to 5 using while loop ---")
i = 0
while i <= 5:
    print(i)
    i += 1
```

Output 6 :-

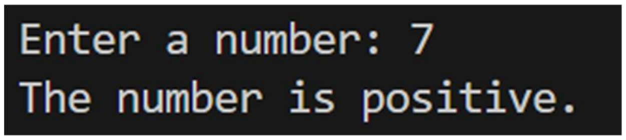


```
0
2
4
6
8
```

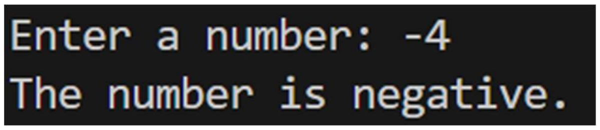
6. **Write a program to print first five even number using break and continue statements.**

```
count = 0
num = 0
while True:
    if num % 2 != 0:
        num += 1
        continue
    print(num)
    count += 1
    if count == 5:
        break
    num += 1
```

Output 7 :-



```
Enter a number: 7
The number is positive.
```

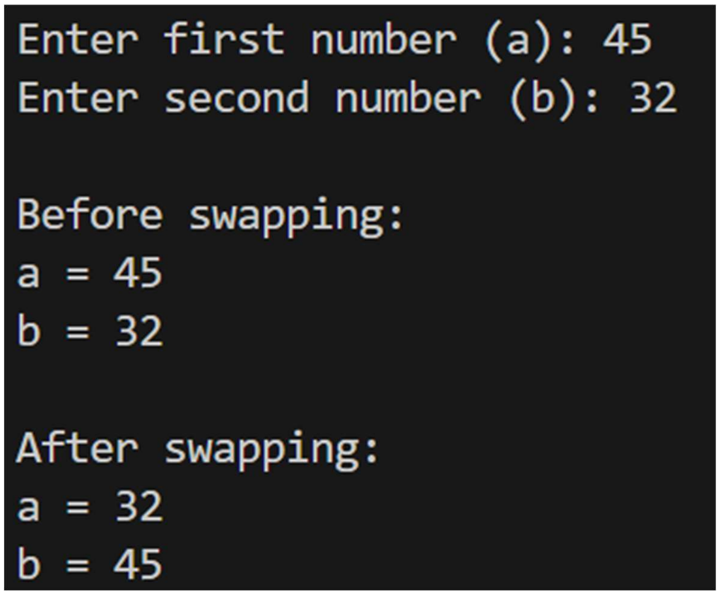


```
Enter a number: -4
The number is negative.
```

7. Write a program to implement if, elif and else statements.

```
num = float(input("Enter a number: "))
if num > 0:
    print("The number is positive.")
elif num < 0:
    print("The number is negative.")
else:
    print("The number is zero.")
```

Output 8 :-

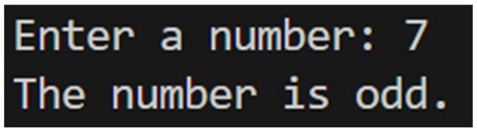


```
Enter first number (a): 45
Enter second number (b): 32

Before swapping:
a = 45
b = 32

After swapping:
a = 32
b = 45
```

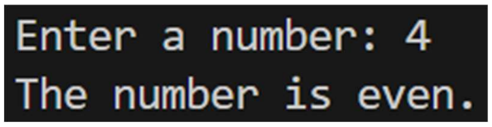
8. Write a program to swap the number of two variables.

```
a = int(input("Enter first number (a): "))
b = int(input("Enter second number (b): "))
print("\nBefore swapping:")
print("a =", a)
print("b =", b)
temp = a
a = b
b = temp
print("\nAfter swapping:")
print("a =", a)
print("b =", b)
```

Output 9 :-



```
Enter a number: 7
The number is odd.
```

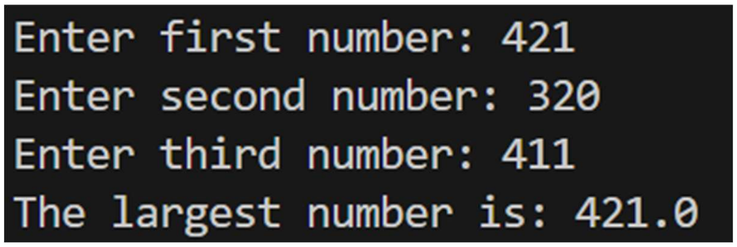


```
Enter a number: 4
The number is even.
```

9. Write a program to find whether a number is even or odd.

```
num = int(input("Enter a number: "))
if num % 2 == 0:
    print("The number is even.")
else:
    print("The number is odd.")
```

Output 10 :-

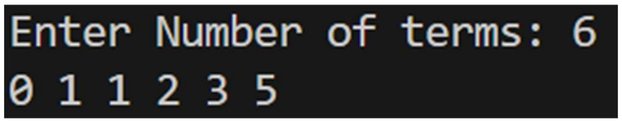


```
Enter first number: 421
Enter second number: 320
Enter third number: 411
The largest number is: 421.0
```

10. Write a program to check largest among the given three numbers.

```
a = float(input("Enter first number: "))
b = float(input("Enter second number: "))
c = float(input("Enter third number: "))
if a >= b and a >= c:
    print("The largest number is:", a)
elif b >= a and b >= c:
    print("The largest number is:", b)
else:
    print("The largest number is:", c)
```

Output 11 :-

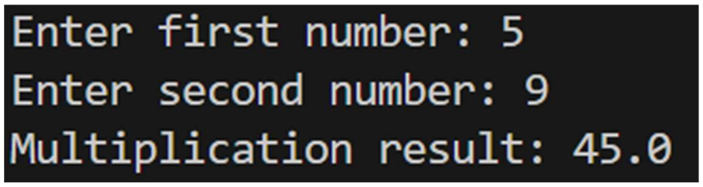


```
Enter Number of terms: 6
0 1 1 2 3 5
```

11. Write a program to display the Fibonacci sequence of n terms.

```
n = int(input("Enter Number of terms: "))
a,b=0,1
if n<=0:
    for i in range(n):
        print(a,end=" ")
        nth=a-b
        a=b
        b=nth
elif n==1:
    print(" ",a," ")
else:
    for i in range(n):
        print(a,end=" ")
        nth=a+b
        a=b
        b=nth
```

Output 12 :-

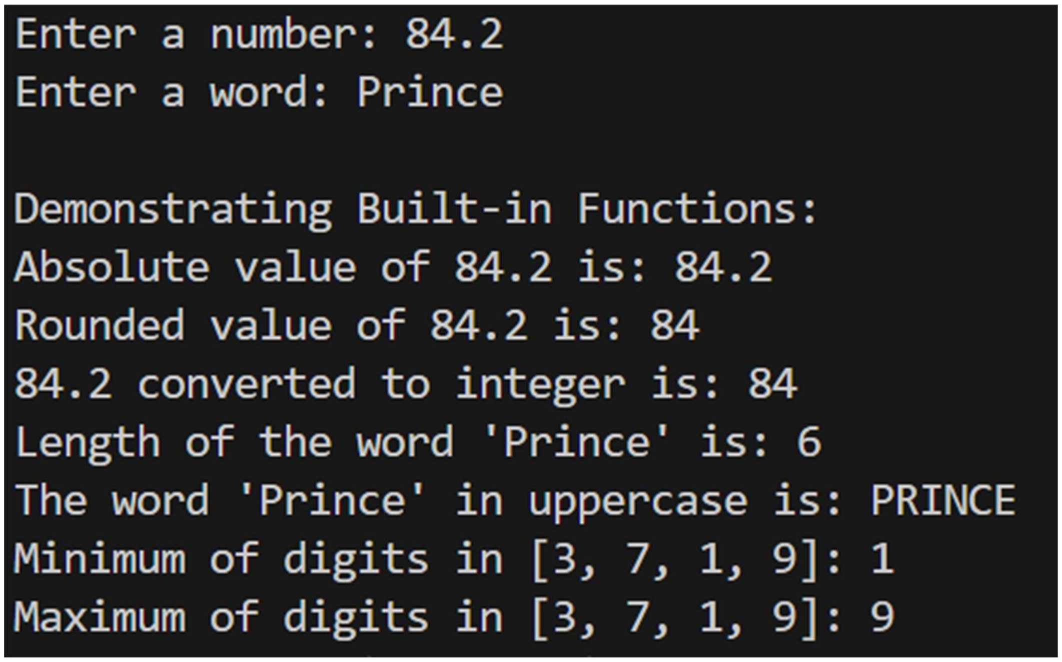


```
Enter first number: 5
Enter second number: 9
Multiplication result: 45.0
```

12. Write a program for multiplication of two numbers using function.

```
def multiply(x, y):
    return x * y
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
result = multiply(num1, num2)
print("Multiplication result:", result)
```

Output 13 :-

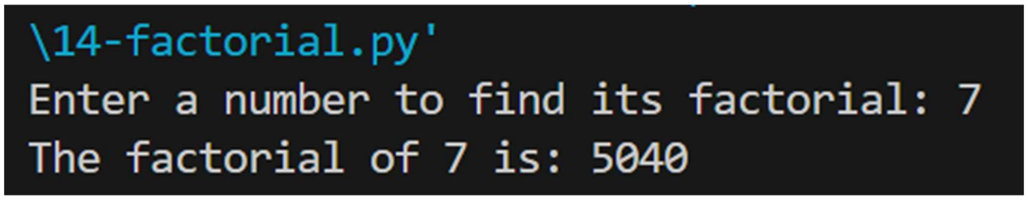


```
Enter a number: 84.2
Enter a word: Prince

Demonstrating Built-in Functions:
Absolute value of 84.2 is: 84.2
Rounded value of 84.2 is: 84
84.2 converted to integer is: 84
Length of the word 'Prince' is: 6
The word 'Prince' in uppercase is: PRINCE
Minimum of digits in [3, 7, 1, 9]: 1
Maximum of digits in [3, 7, 1, 9]: 9
```

13. Write a program that demonstrates the build in functions.

```
num = float(input("Enter a number: "))
word = input("Enter a word: ")
print("\nDemonstrating Built-in Functions:")
print(f"Absolute value of {num} is:", abs(num))
print(f"Rounded value of {num} is:", round(num))
print(f"{num} converted to integer is:", int(num))
print(f"Length of the word '{word}' is:", len(word))
print(f"The word '{word}' in uppercase is:", word.upper())
print(f"Minimum of digits in [3, 7, 1, 9]:", min(3, 7, 1, 9))
print(f"Maximum of digits in [3, 7, 1, 9]:", max(3, 7, 1, 9))
```

Output 14 :-

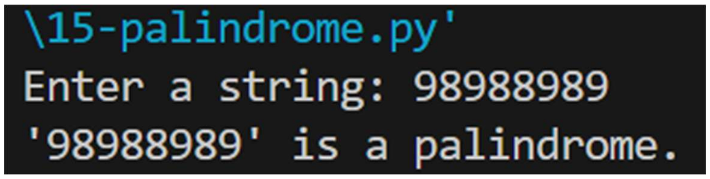


```
\14-factorial.py'
Enter a number to find its factorial: 7
The factorial of 7 is: 5040
```

14. Write a program to implement recursion for factorial of a number that demonstrate the user defined function and return statement.

```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)
num = int(input("Enter a number to find its factorial: "))
if num < 0:
    print("Factorial is not defined for negative numbers.")
else:
    result = factorial(num)
    print(f"The factorial of {num} is: {result}")
```

Output 15 :-

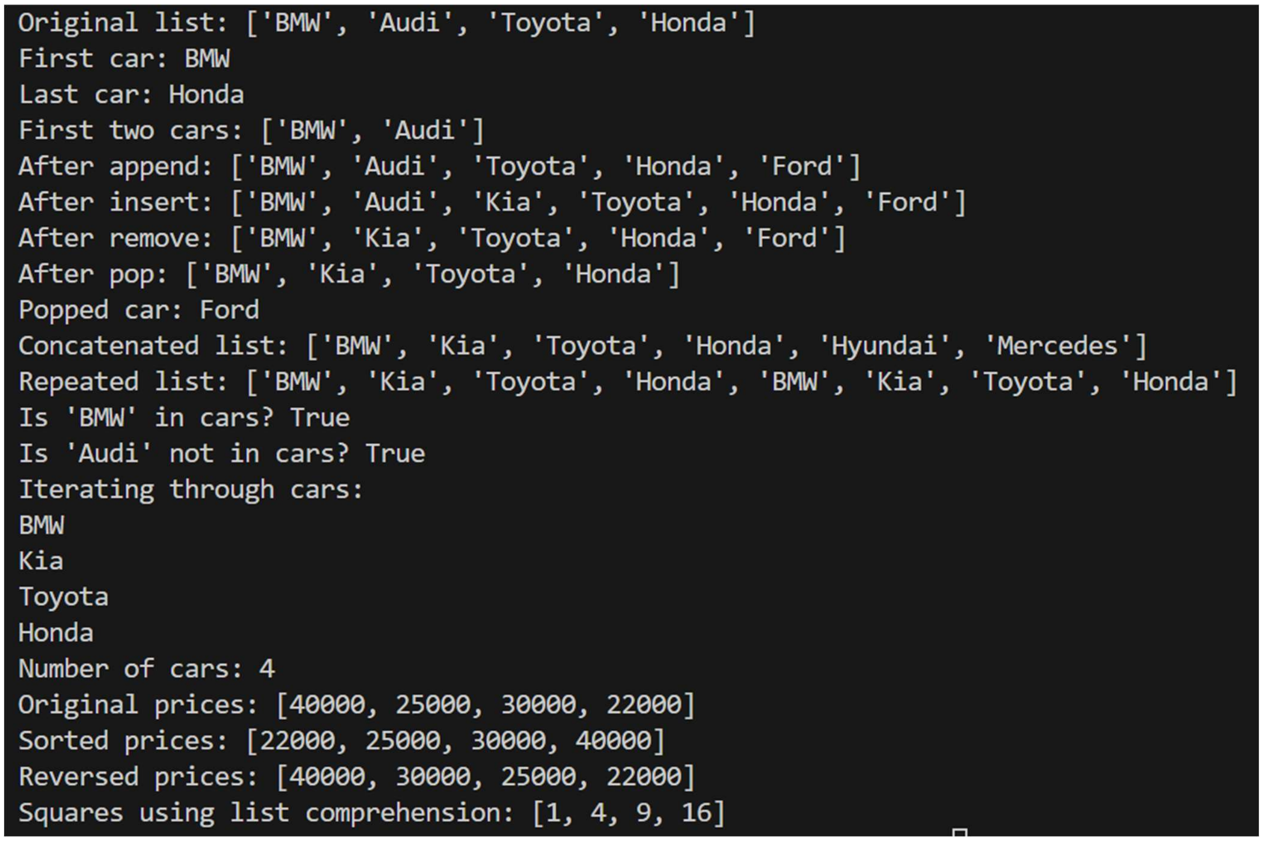


```
\15-palindrome.py'
Enter a string: 98988989
'98988989' is a palindrome.
```

15. Write program to check whether the string is palindrome or not.

```
def is_palindrome(s):
    return s == s[::-1]
string = input("Enter a string: ")
if is_palindrome(string):
    print(f"'{string}' is a palindrome.")
else:
    print(f"'{string}' is not a palindrome.")
```

Output 16 :-

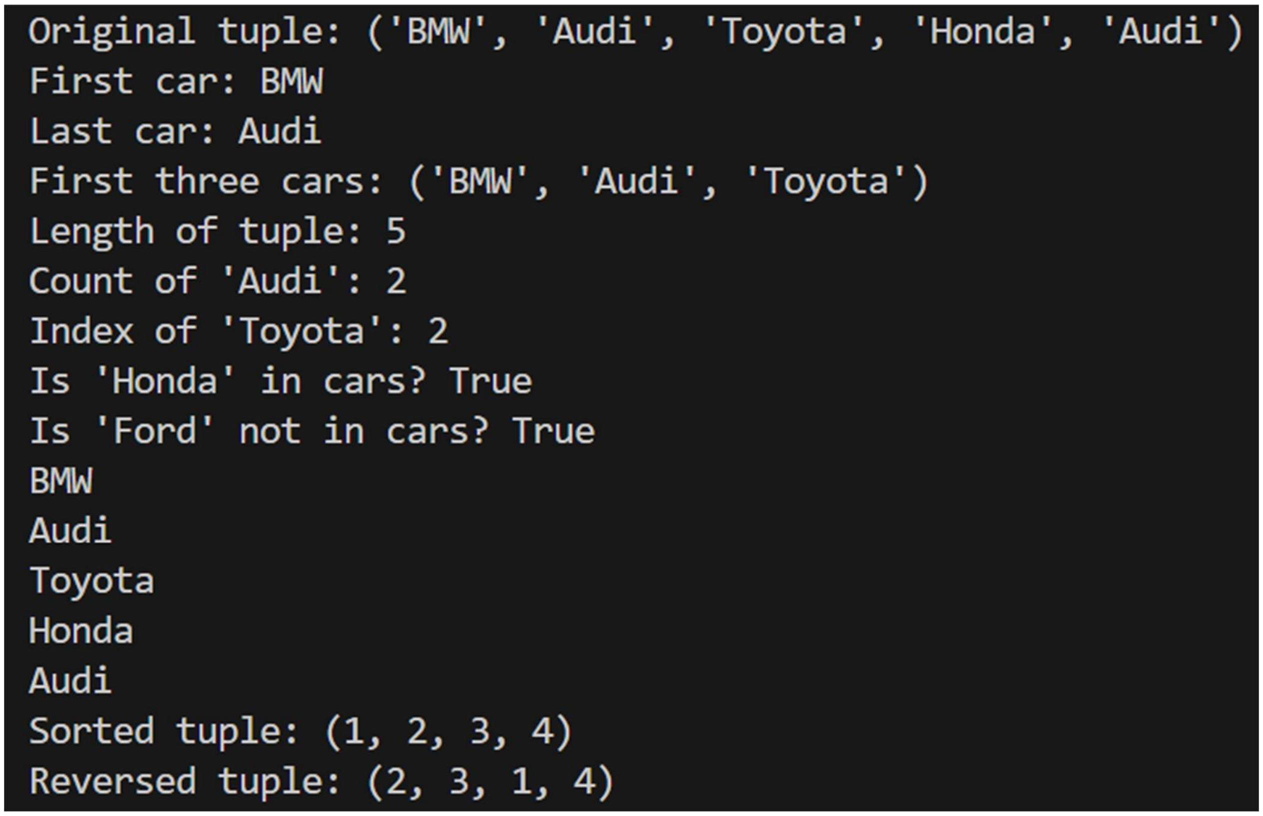

```
Original list: ['BMW', 'Audi', 'Toyota', 'Honda']
First car: BMW
Last car: Honda
First two cars: ['BMW', 'Audi']
After append: ['BMW', 'Audi', 'Toyota', 'Honda', 'Ford']
After insert: ['BMW', 'Audi', 'Kia', 'Toyota', 'Honda', 'Ford']
After remove: ['BMW', 'Kia', 'Toyota', 'Honda', 'Ford']
After pop: ['BMW', 'Kia', 'Toyota', 'Honda']
Popped car: Ford
Concatenated list: ['BMW', 'Kia', 'Toyota', 'Honda', 'Hyundai', 'Mercedes']
Repeated list: ['BMW', 'Kia', 'Toyota', 'Honda', 'BMW', 'Kia', 'Toyota', 'Honda']
Is 'BMW' in cars? True
Is 'Audi' not in cars? True
Iterating through cars:
BMW
Kia
Toyota
Honda
Number of cars: 4
Original prices: [40000, 25000, 30000, 22000]
Sorted prices: [22000, 25000, 30000, 40000]
Reversed prices: [40000, 30000, 25000, 22000]
Squares using list comprehension: [1, 4, 9, 16]
```

16. Write a program to demonstrate list functions and operations.

```
cars = ['BMW', 'Audi', 'Toyota', 'Honda']
print("Original list:", cars)
print("First car:", cars[0])
print("Last car:", cars[-1])
print("First two cars:", cars[0:2])
cars.append('Ford')
print("After append:", cars)
cars.insert(2, 'Kia')
print("After insert:", cars)
cars.remove('Audi')
print("After remove:", cars)
popped_car = cars.pop()
print("After pop:", cars)
print("Popped car:", popped_car)

more_cars = ['Hyundai', 'Mercedes']
all_cars = cars + more_cars
print("Concatenated list:", all_cars)

repeated_cars = cars * 2
print("Repeated list:", repeated_cars)

print("Is 'BMW' in cars?", 'BMW' in cars)
print("Is 'Audi' not in cars?", 'Audi' not in cars)
print("Iterating through cars:")
for car in cars:
    print(car)

print("Number of cars:", len(cars))
prices = [40000, 25000, 30000, 22000]
print("Original prices:", prices)
```

```
prices.sort()
print("Sorted prices:", prices)
prices.reverse()
print("Reversed prices:", prices)
squared_prices = [p**2 for p in range(1, 5)]
print("Squares using list comprehension:", squared_prices)
```

Output 17 :-

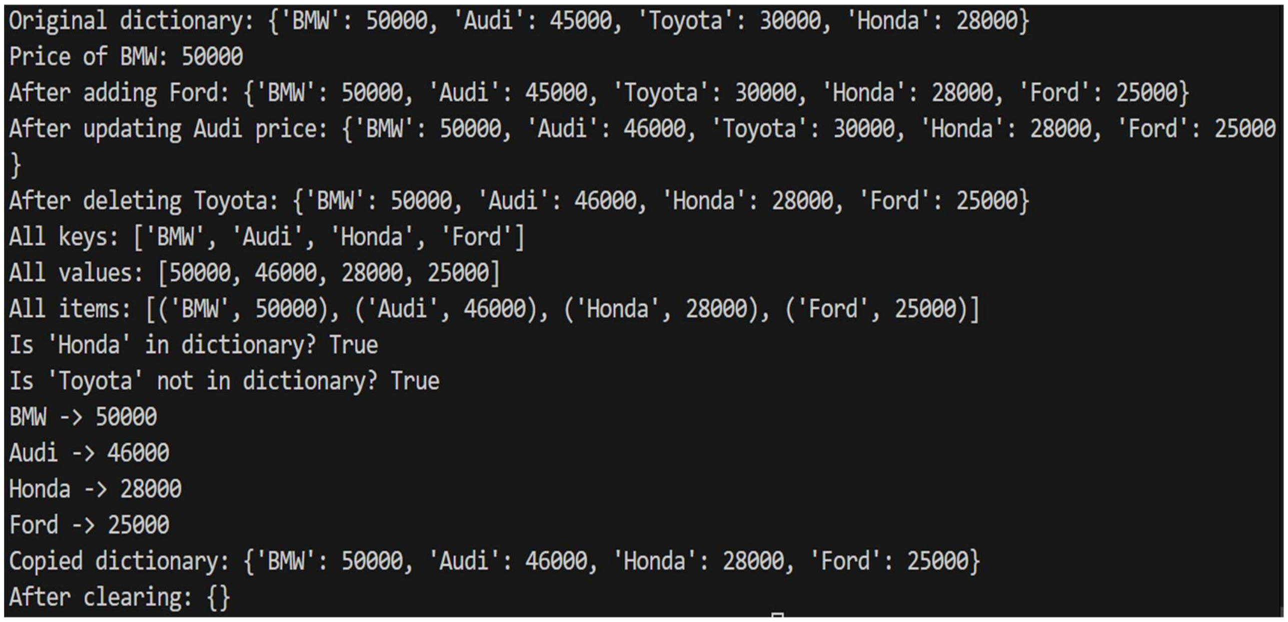


```
Original tuple: ('BMW', 'Audi', 'Toyota', 'Honda', 'Audi')
First car: BMW
Last car: Audi
First three cars: ('BMW', 'Audi', 'Toyota')
Length of tuple: 5
Count of 'Audi': 2
Index of 'Toyota': 2
Is 'Honda' in cars? True
Is 'Ford' not in cars? True
BMW
Audi
Toyota
Honda
Audi
Sorted tuple: (1, 2, 3, 4)
Reversed tuple: (2, 3, 1, 4)
```

17. Write a program to demonstrate tuple functions and operations.

```
cars = ('BMW', 'Audi', 'Toyota', 'Honda', 'Audi')
print("Original tuple:", cars)
print("First car:", cars[0])
print("Last car:", cars[-1])
print("First three cars:", cars[:3])
print("Length of tuple:", len(cars))
print("Count of 'Audi':", cars.count('Audi'))
print("Index of 'Toyota':", cars.index('Toyota'))
print("Is 'Honda' in cars?", 'Honda' in cars)
print("Is 'Ford' not in cars?", 'Ford' not in cars)
for car in cars:
    print(car)
numbers = (4, 1, 3, 2)
sorted_numbers = tuple(sorted(numbers))
print("Sorted tuple:", sorted_numbers)
reversed_numbers = tuple(reversed(numbers))
print("Reversed tuple:", reversed_numbers)
```

Output 18 :-

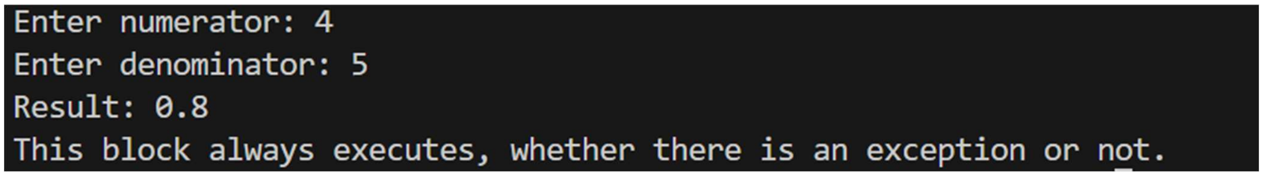


```
Original dictionary: {'BMW': 50000, 'Audi': 45000, 'Toyota': 30000, 'Honda': 28000}
Price of BMW: 50000
After adding Ford: {'BMW': 50000, 'Audi': 45000, 'Toyota': 30000, 'Honda': 28000, 'Ford': 25000}
After updating Audi price: {'BMW': 50000, 'Audi': 46000, 'Toyota': 30000, 'Honda': 28000, 'Ford': 25000
}
After deleting Toyota: {'BMW': 50000, 'Audi': 46000, 'Honda': 28000, 'Ford': 25000}
All keys: ['BMW', 'Audi', 'Honda', 'Ford']
All values: [50000, 46000, 28000, 25000]
All items: [('BMW', 50000), ('Audi', 46000), ('Honda', 28000), ('Ford', 25000)]
Is 'Honda' in dictionary? True
Is 'Toyota' not in dictionary? True
BMW -> 50000
Audi -> 46000
Honda -> 28000
Ford -> 25000
Copied dictionary: {'BMW': 50000, 'Audi': 46000, 'Honda': 28000, 'Ford': 25000}
After clearing: {}
```

18. Write a program to demonstrate dictionary functions and operations.

```
car_prices = {'BMW': 50000, 'Audi': 45000, 'Toyota': 30000, 'Honda': 28000}
print("Original dictionary:", car_prices)
print("Price of BMW:", car_prices['BMW'])

car_prices['Ford'] = 25000
print("After adding Ford:", car_prices)

car_prices['Audi'] = 46000
print("After updating Audi price:", car_prices)

del car_prices['Toyota']
print("After deleting Toyota:", car_prices)

print("All keys:", list(car_prices.keys()))
print("All values:", list(car_prices.values()))
print("All items:", list(car_prices.items()))

print("Is 'Honda' in dictionary?", 'Honda' in car_prices)
print("Is 'Toyota' not in dictionary?", 'Toyota' not in car_prices)

for car, price in car_prices.items():
    print(car, "->", price)

price_copy = car_prices.copy()
print("Copied dictionary:", price_copy)

car_prices.clear()
print("After clearing:", car_prices)
```

Output 19 :-

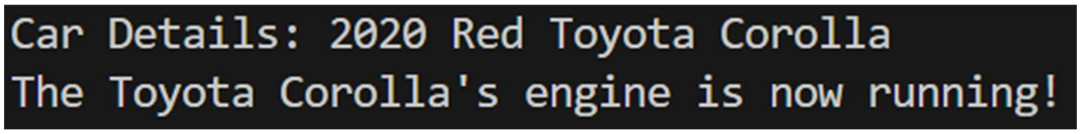


```
Enter numerator: 4
Enter denominator: 5
Result: 0.8
This block always executes, whether there is an exception or not.
```

19. Write a program to catch a divide by zero exception and add a finally block too.

```
def divide_numbers(a, b):
    try:
        result = a / b
        print(f"Result: {result}")
    except ZeroDivisionError:
        print("Error: Cannot divide by zero!")
    finally:
        print("This block always executes, whether there is an exception or not.")

num1 = int(input("Enter numerator: "))
num2 = int(input("Enter denominator: "))
divide_numbers(num1, num2)
```

Output 20 :-

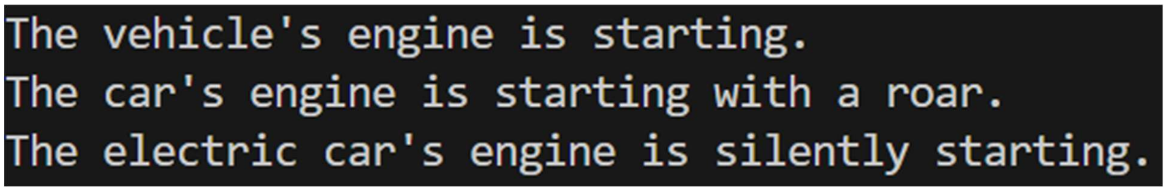


```
Car Details: 2020 Red Toyota Corolla
The Toyota Corolla's engine is now running!
```

20. Write a program to demonstrate classes and their attributes.

```
class Car:
    def __init__(self, make, model, year, color):
        self.make = make
        self.model = model
        self.year = year
        self.color = color

    def display_info(self):
        print(f"Car Details: {self.year} {self.color} {self.make} {self.model}")

    def start_engine(self):
        print(f"The {self.make} {self.model}'s engine is now running!")

car1 = Car("Toyota", "Corolla", 2020, "Red")
car1.display_info()
car1.start_engine()
```

Output 21 :-

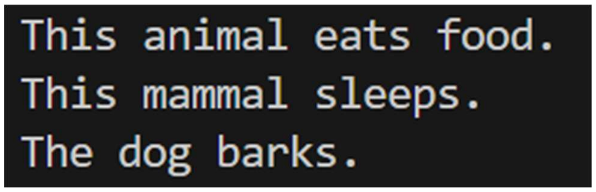


```
The vehicle's engine is starting.
The car's engine is starting with a roar.
The electric car's engine is silently starting.
```

21. Write a program to demonstrate inheritance and method overriding.

```
class Car(Vehicle):
    def start_engine(self):
        print("The car's engine is starting with a roar.")

class ElectricCar(Car):
    def start_engine(self):
        print("The electric car's engine is silently starting.")

vehicle = Vehicle()
car = Car()
electric_car = ElectricCar()

vehicle.start_engine()
car.start_engine()
electric_car.start_engine()
```

Output 22 :-

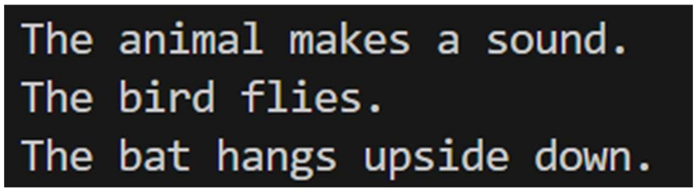


```
This animal eats food.
This mammal sleeps.
The dog barks.
```

22. Write a program to show the use of multilevel inheritance.

```
class Animal:
    def eat(self):
        print("This animal eats food.")

class Mammal(Animal):
    def sleep(self):
        print("This mammal sleeps.")

class Dog(Mammal):
    def bark(self):
        print("The dog barks.")

dog = Dog()
dog.eat()
dog.sleep()
dog.bark()
```

Output 23 :-

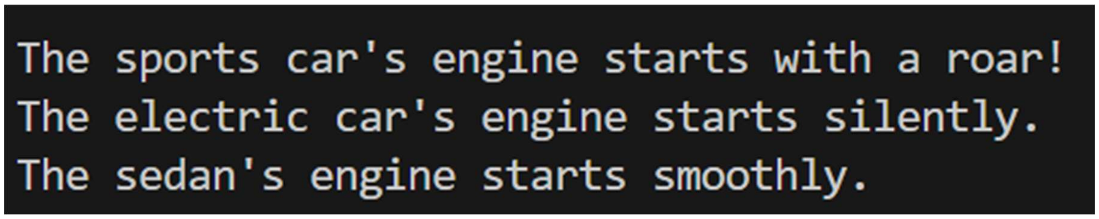


```
The animal makes a sound.
The bird flies.
The bat hangs upside down.
```

23. Write a program to show the use of multiple inheritance.

```
class Animal:
    def speak(self):
        print("The animal makes a sound.")


class Bird:
    def fly(self):
        print("The bird flies.")


class Bat(Animal, Bird):
    def hang(self):
        print("The bat hangs upside down.")


bat = Bat()
bat.speak()
bat.fly()
bat.hang()
```

Output 24 :-

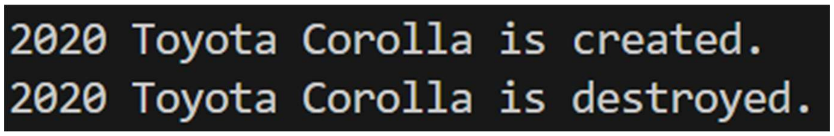


```
The sports car's engine starts with a roar!
The electric car's engine starts silently.
The sedan's engine starts smoothly.
```

24. Write a program to implement polymorphism

```
class Car:
    def start_engine(self):
        print("The car's engine is starting.")

class SportsCar(Car):
    def start_engine(self):
        print("The sports car's engine starts with a roar!")

class ElectricCar(Car):
    def start_engine(self):
        print("The electric car's engine starts silently.")

class Sedan(Car):
    def start_engine(self):
        print("The sedan's engine starts smoothly.")

def start_car_engine(car):
    car.start_engine()

sports_car = SportsCar()
electric_car = ElectricCar()
sedan = Sedan()

start_car_engine(sports_car)
start_car_engine(electric_car)
start_car_engine(sedan)
```

Output 25 :-

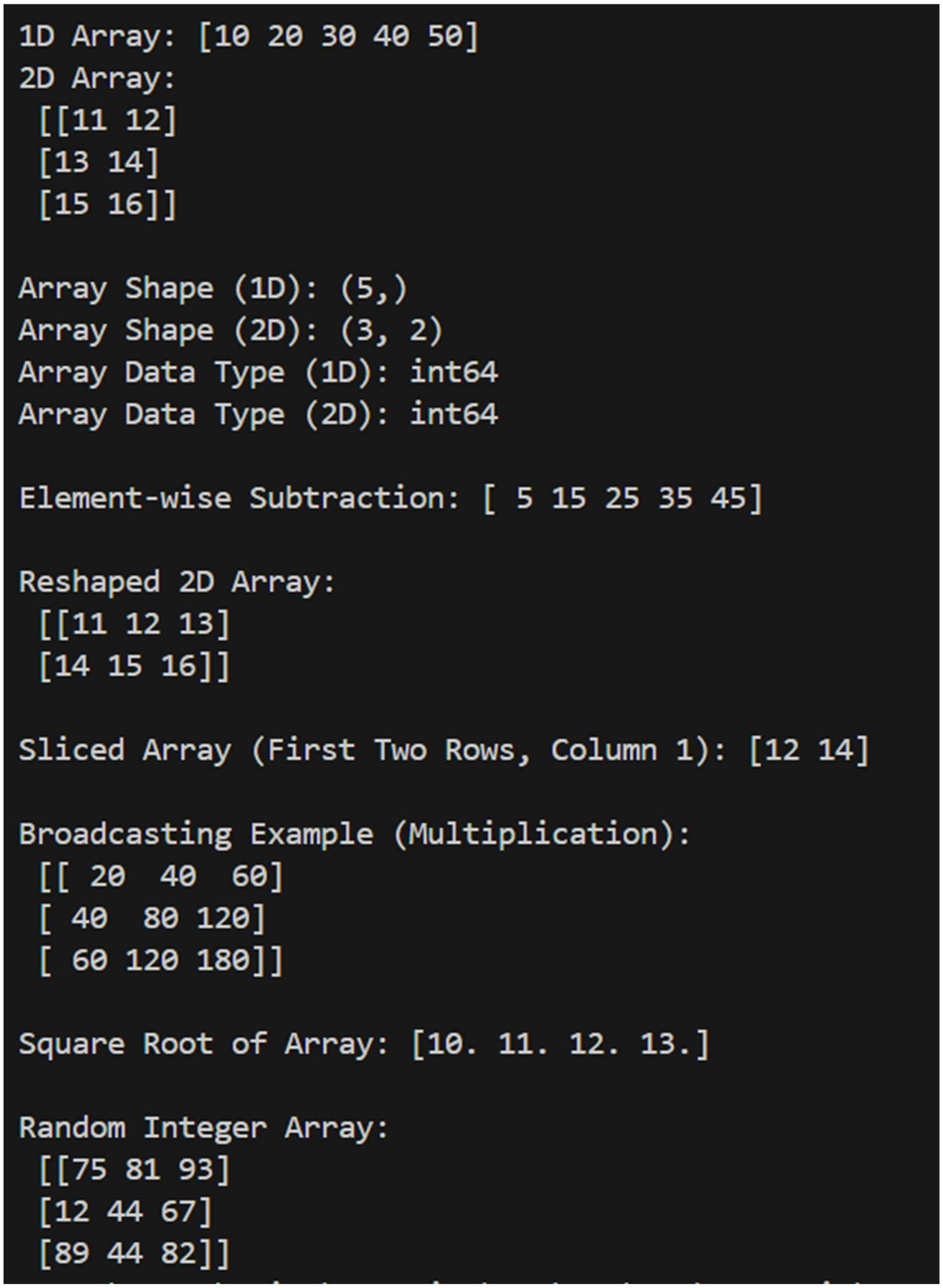


```
2020 Toyota Corolla is created.
2020 Toyota Corolla is destroyed.
```

25. Write a program to implement constructor and destructor.

```
class Car:
    def __init__(self, make, model, year):
        self.make = make
        self.model = model
        self.year = year
        print(f"{self.year} {self.make} {self.model} is created.")

    def __del__(self):
        print(f"{self.year} {self.make} {self.model} is destroyed.")

car1 = Car("Toyota", "Corolla", 2020)
del car1
```

Output 26 :-



```
1D Array: [10 20 30 40 50]
2D Array:
 [[11 12]
 [13 14]
 [15 16]]

Array Shape (1D): (5,)
Array Shape (2D): (3, 2)
Array Data Type (1D): int64
Array Data Type (2D): int64

Element-wise Subtraction: [ 5 15 25 35 45]

Reshaped 2D Array:
 [[11 12 13]
 [14 15 16]]

Sliced Array (First Two Rows, Column 1): [12 14]

Broadcasting Example (Multiplication):
 [[ 20  40  60]
 [ 40  80 120]
 [ 60 120 180]]

Square Root of Array: [10. 11. 12. 13.]

Random Integer Array:
 [[75 81 93]
 [12 44 67]
 [89 44 82]]
```

26. Write a program to demonstrate NumPy basics

```
import numpy as np

# 1. Creating NumPy arrays
arr1 = np.array([10, 20, 30, 40, 50])
arr2 = np.array([[11, 12], [13, 14], [15, 16]])

print("1D Array:", arr1)
print("2D Array:\n", arr2)

# 2. Array properties
print("\nArray Shape (1D):", arr1.shape)
print("Array Shape (2D):", arr2.shape)
print("Array Data Type (1D):", arr1.dtype)
print("Array Data Type (2D):", arr2.dtype)

# 3. Array Operations
arr3 = np.array([5, 5, 5, 5, 5])
difference_arr = arr1 - arr3
print("\nElement-wise Subtraction:", difference_arr)

# 4. Array Reshaping
reshaped_arr = arr2.reshape(2, 3)
print("\nReshaped 2D Array:\n", reshaped_arr)

# 5. Array Slicing
sliced_arr = arr2[0:2, 1]
print("\nSliced Array (First Two Rows, Column 1):", sliced_arr)

# 6. Array Broadcasting
arr4 = np.array([10, 20, 30])
arr5 = np.array([[2], [4], [6]])
broadcasted_arr = arr4 * arr5
print("\nBroadcasting Example (Multiplication):\n", broadcasted_arr)
```

```
# 7. Array Mathematical Functions
arr6 = np.array([100, 121, 144, 169])
sqrt_arr = np.sqrt(arr6)
print("\nSquare Root of Array:", sqrt_arr)

# 8. Random Array
random_arr = np.random.randint(1, 100, size=(3, 3))
print("\nRandom Integer Array:\n", random_arr)
```