```
#1. Cube a Number Using return
def cube(num):
    return num ** 3

print("Task 1 Output:", cube(3))
```

```
Task 1 Output: 27
```

```
# 2. Using break in a Loop
while True:
    num = int(input("Task 2 - Enter a number: "))
    if num < 0:
        print("Negative number entered. Exiting.")
        break
    print("Cube:", num ** 3)
```

```
Task 2 - Enter a number: 100
Cube: 1000000
Task 2 - Enter a number: 200
Cube: 8000000
Task 2 - Enter a number: -300
Negative number entered. Exiting.
```

```
#3. Using continue in a Loop (Even numbers 1–20)
print("Task 3 Output:")
for i in range(1, 21):
    if i % 2 != 0:
        continue
    print(i)
```

```
Task 3 Output:
2
4
6
8
10
12
14
16
18
20
```

```
#4. break and continue Example
total = 0
while True:
    num = int(input("Task 4 - Enter a number (0 to stop): "))
    if num < 0:
        continue
    if num == 0:
        break
    total += num
```

```
print("Task 4 Output - Sum of positive numbers:", total)
```

```
Task 4 - Enter a number (0 to stop): 100
Task 4 - Enter a number (0 to stop): 0
Task 4 Output - Sum of positive numbers: 100
```

```
#5. Local Scope in a Function
def square():
    local_var = 5
    result = local_var ** 2
    print("Task 5 Output - Square is:", result)

square()
```

```
Task 5 Output - Square is: 25
```

```
#6. Modify a Global Variable
counter = 0

def increment():
    global counter
    counter += 1
    print("Task 6 Output - Counter:", counter)

increment()
increment()
increment()
```

```
Task 6 Output - Counter: 1
Task 6 Output - Counter: 2
Task 6 Output - Counter: 3
```

```
#7. Local vs Global Scope
x = 10

def modify_global():
    global x
    x = 20
    print("Task 7 Output - Inside function, global x:", x)

def local_scope():
    y = 30
    print("Task 7 Output - Inside function, local y:", y)

modify_global()
print("Task 7 Output - Outside function, global x:", x)

local_scope()
```

```
Task 7 Output - Inside function, global x: 20
Task 7 Output - Outside function, global x: 20
Task 7 Output - Inside function, local y: 30
```

```python
#8. Working with Tuples
fruits = ("apple", "banana", "cherry")
print("Task 8 Output - Second element:", fruits[1])

# Convert tuple to list to modify
fruits_list = list(fruits)
fruits_list[1] = "orange"
fruits = tuple(fruits_list)
print("Task 8 Output - Modified tuple:", fruits)
```

```
Task 8 Output - Second element: banana
Task 8 Output - Modified tuple: ('apple', 'orange', 'cherry')
```

```python
#9. Working with Sets
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}

print("Task 9 Output - Union:", set1 | set2)
print("Task 9 Output - Intersection:", set1 & set2)
print("Task 9 Output - Difference:", set1 - set2)

set1.add(6)
set2.remove(4)

print("Task 9 Output - Updated set1:", set1)
print("Task 9 Output - Updated set2:", set2)
print("Task 9 Output - Is 3 in set1?", 3 in set1)
```

```
Task 9 Output - Union: {1, 2, 3, 4, 5, 6, 7, 8}
Task 9 Output - Intersection: {4, 5}
Task 9 Output - Difference: {1, 2, 3}
Task 9 Output - Updated set1: {1, 2, 3, 4, 5, 6}
Task 9 Output - Updated set2: {5, 6, 7, 8}
Task 9 Output - Is 3 in set1? True
```

```python
#10. Tuple and Set Operations
my_tuple = (10, 20, 30, 40, 50)
print("Task 10 Output - Third element:", my_tuple[2])

my_set = set(my_tuple)
my_set.add(60)
print("Task 10 Output - Updated set:", my_set)
```

```
Task 10 Output - Third element: 30
Task 10 Output - Updated set: {40, 10, 50, 20, 60, 30}
```

```python
#11. List to Set Conversion
num_list = [1, 2, 2, 3, 4, 4, 5]
unique_set = set(num_list)
print("Task 11 Output - Unique elements:", unique_set)

another_set = {3, 4, 5, 6, 7}
print("Task 11 Output - Intersection:", unique_set & another_set)
print("Task 11 Output - Union:", unique_set | another_set)
```

```
Task 11 Output - Unique elements: {1, 2, 3, 4, 5}
Task 11 Output - Intersection: {3, 4, 5}
Task 11 Output - Union: {1, 2, 3, 4, 5, 6, 7}
```

```python
#12. Tuples, Sets, and Lists Together
num_list = [10, 20, 30, 40, 50]
num_list.append(60)

num_tuple = tuple(num_list)
print("Task 12 Output - Tuple:", num_tuple)

num_set = set(num_list)
print("Task 12 Output - Unique values (set):", num_set)
```

```
Task 12 Output - Tuple: (10, 20, 30, 40, 50, 60)
Task 12 Output - Unique values (set): {40, 10, 50, 20, 60, 30}
```

Start coding or generate with AI.