A.YUVAN CHARAN

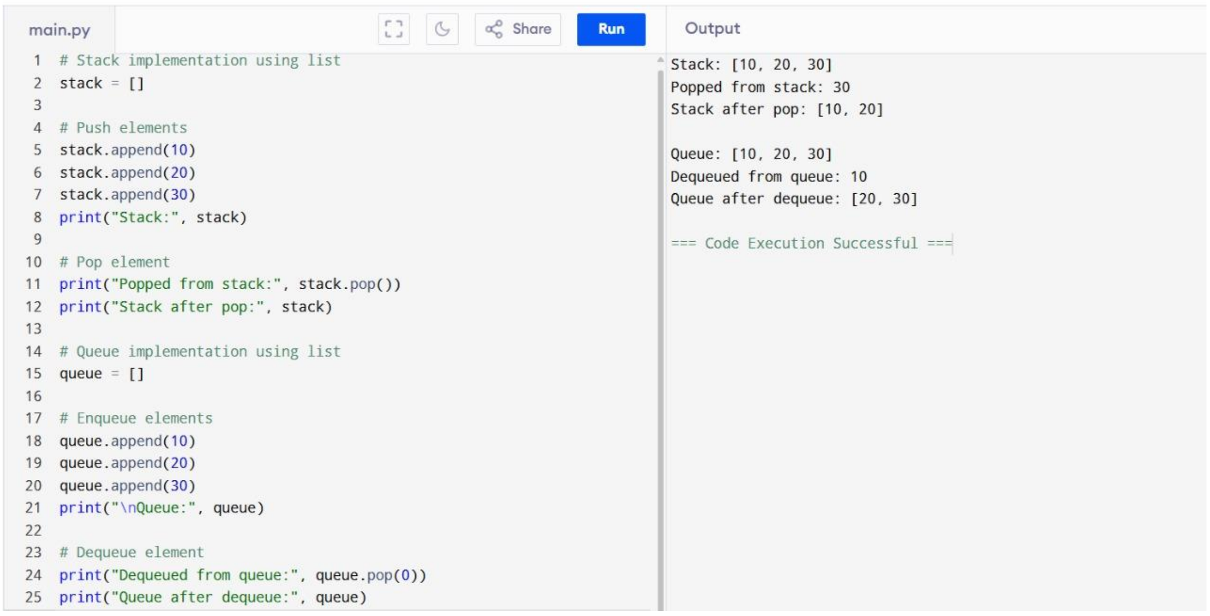
192424383

CSA0815

PYTHON PROGRAMMING

SLOT B

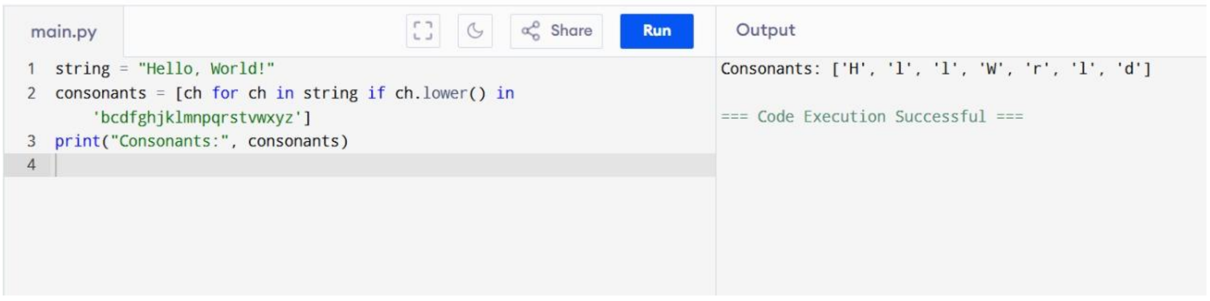
1.Write a program to implement the stack & queue data structure using list



```python
# Stack implementation using list
stack = []

# Push elements
stack.append(10)
stack.append(20)
stack.append(30)
print("Stack:", stack)

# Pop element
print("Popped from stack:", stack.pop())
print("Stack after pop:", stack)

# Queue implementation using list
queue = []

# Enqueue elements
queue.append(10)
queue.append(20)
queue.append(30)
print("\nQueue:", queue)

# Dequeue element
print("Dequeued from queue:", queue.pop(0))
print("Queue after dequeue:", queue)
```

```
Stack: [10, 20, 30]
Popped from stack: 30
Stack after pop: [10, 20]

Queue: [10, 20, 30]
Dequeued from queue: 10
Queue after dequeue: [20, 30]

=== Code Execution Successful ===
```

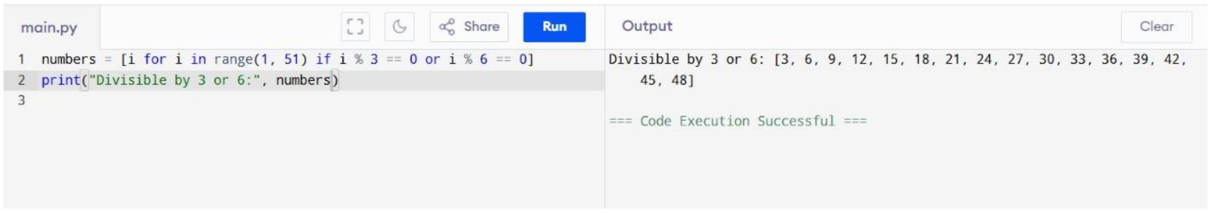
2.Write a program that prints all consonants in a string using list comprehension



```python
string = "Hello, World!"
consonants = [ch for ch in string if ch.lower() in
    'bcdfghjklmnpqrstvwxyz']
print("Consonants:", consonants)

```

```
Consonants: ['H', 'l', 'l', 'W', 'r', 'l', 'd']

=== Code Execution Successful ===
```

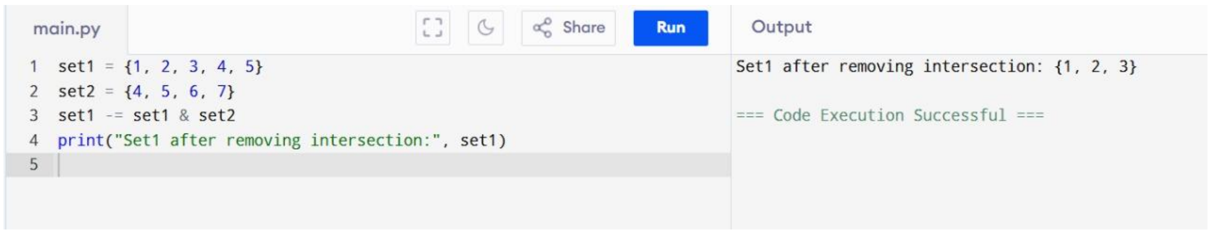
3.Write a program that creates a list of numbers from 1-50 that are either divisible by 3 or divisible by 6.



```python
numbers = [i for i in range(1, 51) if i % 3 == 0 or i % 6 == 0]
print("Divisible by 3 or 6:", numbers)
```

```
Divisible by 3 or 6: [3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42,
    45, 48]

=== Code Execution Successful ===
```

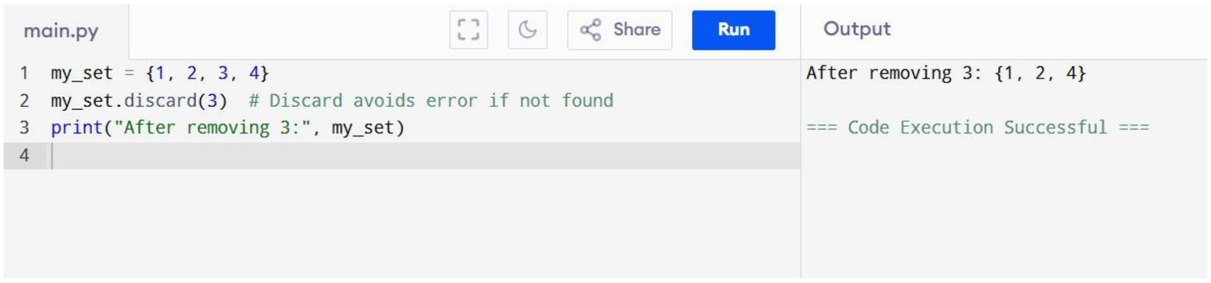
4.Write a Python program to remove the intersection of a 2nd set from the 1st set.



```python
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7}
set1 -= set1 & set2
print("Set1 after removing intersection:", set1)
```

```
Set1 after removing intersection: {1, 2, 3}

=== Code Execution Successful ===
```

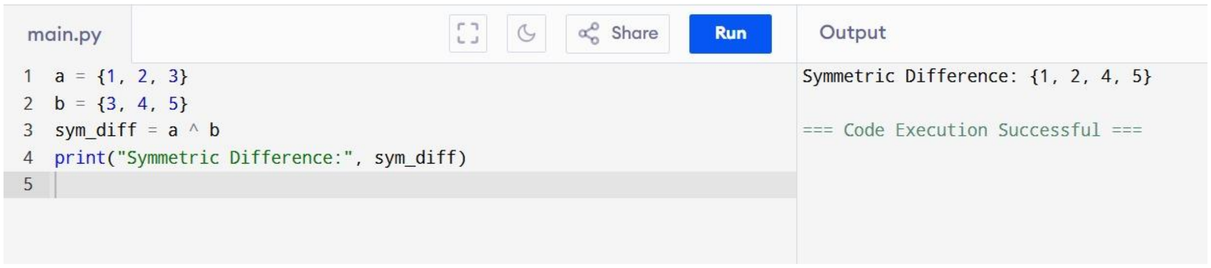
5.Write a Python program to remove an item from a set if it is present in the set.



```python
my_set = {1, 2, 3, 4}
my_set.discard(3)  # Discard avoids error if not found
print("After removing 3:", my_set)
```

```
After removing 3: {1, 2, 4}

=== Code Execution Successful ===
```

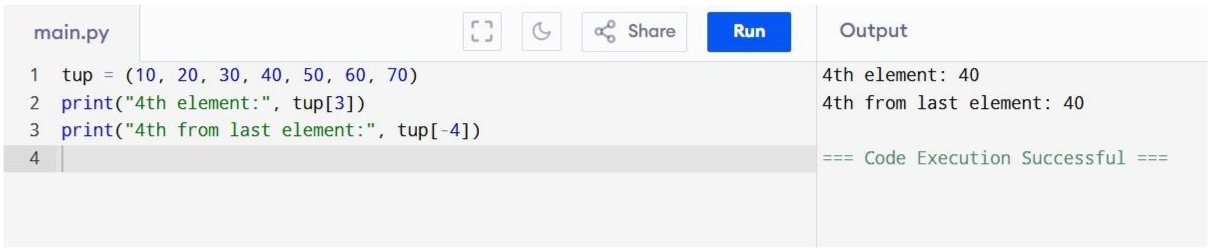
6.Write a Python program to create a symmetric difference.



```python
a = {1, 2, 3}
b = {3, 4, 5}
sym_diff = a ^ b
print("Symmetric Difference:", sym_diff)
```

```
Symmetric Difference: {1, 2, 4, 5}

=== Code Execution Successful ===
```

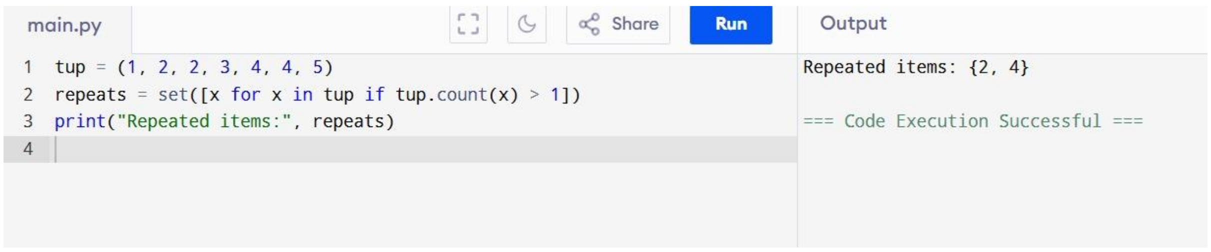
7.Write a Python program to get the 4th element and 4th element from last of a tuple.



```python
tup = (10, 20, 30, 40, 50, 60, 70)
print("4th element:", tup[3])
print("4th from last element:", tup[-4])
```

```
4th element: 40
4th from last element: 40

=== Code Execution Successful ===
```

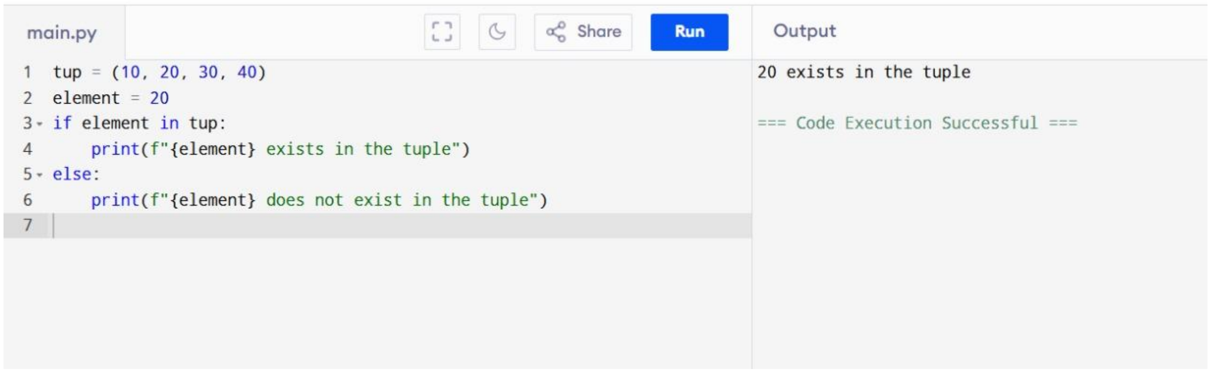
8.Write a Python program to find the repeated items of a tuple.



```python
tup = (1, 2, 2, 3, 4, 4, 5)
repeats = set([x for x in tup if tup.count(x) > 1])
print("Repeated items:", repeats)
```

```
Repeated items: {2, 4}

=== Code Execution Successful ===
```

9. Write a Python program to check whether an element exists within a tuple.



```python
tup = (10, 20, 30, 40)
element = 20
if element in tup:
    print(f"{element} exists in the tuple")
else:
    print(f"{element} does not exist in the tuple")
```

```
20 exists in the tuple

=== Code Execution Successful ===
```