

P.DILLI BABU

192472284

CSA0815

PYTHON PROGRAMMING

SLOT B




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

main.py	Output
<pre>1 # Stack implementation using list 2 stack = [] 3 4 # Push elements 5 stack.append(10) 6 stack.append(20) 7 stack.append(30) 8 print("Stack:", stack) 9 10 # Pop element 11 print("Popped from stack:", stack.pop()) 12 print("Stack after pop:", stack) 13 14 # Queue implementation using list 15 queue = [] 16 17 # Enqueue elements 18 queue.append(10) 19 queue.append(20) 20 queue.append(30) 21 print("\nQueue:", queue) 22 23 # Dequeue element 24 print("Dequeued from queue:", queue.pop(0)) 25 print("Queue after dequeue:", queue)</pre>	<pre>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 ===</pre>

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

main.py	Output
<pre>1 string = "Hello, World!" 2 consonants = [ch for ch in string if ch.lower() in 3               'bcdfghjklmnpqrstvwxyz'] 4 print("Consonants:", consonants)</pre>	<pre>Consonants: ['H', 'l', 'l', 'W', 'r', 'l', 'd']  === Code Execution Successful ===</pre>




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

main.py	   Share	Run	Output	Clear
<pre>1 numbers = [i for i in range(1, 51) if i % 3 == 0 or i % 6 == 0] 2 print("Divisible by 3 or 6:", numbers) 3</pre>			<pre>Divisible by 3 or 6: [3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48]  === Code Execution Successful ===</pre>	

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

main.py	   Share	Run	Output
<pre>1 set1 = {1, 2, 3, 4, 5} 2 set2 = {4, 5, 6, 7} 3 set1 -= set1 &amp; set2 4 print("Set1 after removing intersection:", set1) 5</pre>			<pre>Set1 after removing intersection: {1, 2, 3}  === Code Execution Successful ===</pre>

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

main.py	   Share	Run	Output
<pre>1 my_set = {1, 2, 3, 4} 2 my_set.discard(3) # Discard avoids error if not found 3 print("After removing 3:", my_set) 4</pre>			<pre>After removing 3: {1, 2, 4}  === Code Execution Successful ===</pre>

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

main.py	   Share	Run	Output
<pre>1 a = {1, 2, 3} 2 b = {3, 4, 5} 3 sym_diff = a ^ b 4 print("Symmetric Difference:", sym_diff) 5</pre>			<pre>Symmetric Difference: {1, 2, 4, 5}  === Code Execution Successful ===</pre>

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

main.py	   Share	Run	Output
<pre>1 tup = (10, 20, 30, 40, 50, 60, 70) 2 print("4th element:", tup[3]) 3 print("4th from last element:", tup[-4]) 4</pre>			<pre>4th element: 40 4th from last element: 40  === Code Execution Successful ===</pre>

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

main.py	Output
<pre>1 tup = (1, 2, 2, 3, 4, 4, 5) 2 repeats = set([x for x in tup if tup.count(x) &gt; 1]) 3 print("Repeated items:", repeats) 4</pre>	<pre>Repeated items: {2, 4}  === Code Execution Successful ===</pre>

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

main.py	Output
<pre>1 tup = (10, 20, 30, 40) 2 element = 20 3 if element in tup: 4     print(f"{element} exists in the tuple") 5 else: 6     print(f"{element} does not exist in the tuple") 7</pre>	<pre>20 exists in the tuple  === Code Execution Successful ===</pre>