5. Go through the code and write your inference of the code, especially about the variable 'a'. [CO1]

```
def abc(a):
    print(a)
def xyz(a):
    print(a)
abc(10)
xyz(20)
```

Question

Given an array of integers nums and an integer target, return indices of the two numbers such that they add up to the target. Write the algorithm followed by the code to achieve the same.

python

Copy code

```python
def two_sum(nums, target):
    d = {}  # dictionary to store number and index

    for i, num in enumerate(nums):
        complement = target - num

        # check if complement already exists
        if complement in d:
            return [d[complement], i]

        # store current number with index
        d[num] = i

# Example usage:
nums = [2, 7, 11, 15]
target = 9
print(two_sum(nums, target))  # Output: [0, 1]
```

6. Go through the code and write your inference of the code, especially about the variable 'a'. [CO1]

```
def abc():
    a=10
    print(a)
def xyz():
    print(a)
abc()
xyz()
```

Ans: Write the answer

7. Find the output

```
def abc(x):
    x = 10
    return x

def xyz(x):
    x = 20
    return x

a = abc(5)
b = xyz(8)

print(a)
print(b)
```

Ans:

8. Find the output

```
def abc(x):
    x = 10

def xyz(x):
    x = 20

a = abc(5)
b = xyz(8)
```

9. Explain the concept of local variable. In the code given below, is the variable 'x' in the function 'abc', the same as the variable 'x' in the function, 'xyz'.

```
def abc(x):
    x = 10

def xyz(x):
    x = 20

a = abc(5)
b = xyz(8)

print(a)
print(b)
```

Ans:

10. Explain the concept of global variable. Differentiate a global variable from a local variable with respect to the code given below.

```
def abc(x):
    x = 10
    print(x)
def xyz(x):
    x = 20
    print(x)
a = abc(5)
b = xyz(8)
x = 100
print(x)
```

```
x = 100
def abc():
    print("Inside-function-abc,x=", x)
def xyz():
    print("Inside-function-xyz,x=", x)
a = abc()
b = xyz()
print("Outside-all-functions,x=",x)
```

(f) Another example to count the number of times a number occurs in an array

Listing 3: Counting Frequency of Elements

```
# Sample list of numbers
arr = [1, 2, 2, 3, 3, 3, 4]

# Initialize an empty dictionary to store frequencies
freq = {}

# Iterate through each number in the list
for num in arr:
    if num in freq:    # If the number is already in the
       dictionary
        freq[num] = freq[num] + 1        # Increment its count
        print("in_if=",freq)
    else:
        print("num_first_time=",num)
        # Otherwise, add it to the dictionary with count 1
        freq[num] = 1
        print(freq)

# Print the frequency dictionary
print(freq)
```

(h) Code to find the element having the highest frequency using Dictionary

Listing 5: Iterating Through a Dictionary with items()

```
def mostFreqNum(arr):
    freq={}
    n = len(arr)
    for num in arr:
        if num in freq:
            freq[num] = freq[num] + 1
        else:
            freq[num] = 1
    #How does the dictionary look like?
    print(freq)
    #max_count is used to find the maximum count
    max_count=0
    #to store the number with the maximum count
    res = 0
    for key, count in freq.items():
        # Update if frequency is higher
        print("Key=", key, "value=",count)
        if max_count < count:
            res = key
            max_count = count

    return res

if __name__ == "__main__":
    arr = [40, 50, 30, 40, 50, 30, 30]
    print(mostFreqNum(arr))
```

Question

Given an array of integers nums and an integer target, return true if there exists a pair whose sum equals the target, else return false. Write the algorithm followed by the code to achieve the same.

Example:
nums = [2, 7, 11, 15], target = 9
Output: **True**

(Hint: Use a **Set**, not a Dictionary, because we don't need indices.)

✅ Python Code (Using Set)

python

```python
def two_sum_exists(nums, target):
    s = set()  # set to store seen numbers

    for num in nums:
        complement = target - num

        if complement in s:
            return True

        s.add(num)

    return False

# Example
nums = [2, 7, 11, 15]
target = 9
print(two_sum_exists(nums, target))  # Output: True
```

Question

Problem: Maximum Sum with Non-Adjacent Elements
You are given an array of non-negative integers nums = [n1, n2, ..., nk]. You want to select a subset of values such that no two selected values are adjacent, and the sum is maximized. Return the maximum sum.

Python Code

python

```python
def maxNonAdjacentSum(nums):
    incl = 0  # include current element
    excl = 0  # exclude current element

    for num in nums:
        new_incl = excl + num
        new_excl = max(incl, excl)

        incl = new_incl
        excl = new_excl

    return max(incl, excl)

# Examples
print(maxNonAdjacentSum([1, 2, 3, 1]))  # 4
print(maxNonAdjacentSum([2, 7, 9, 3, 1]))  # 12
```