NAME: AMOIT LYNN
REG NO: M23B38/007
ACCESS NO: B20813
COURSE: BACHELORS OF SCIENCE IN DATA SCIENCE AND ANALYTICS

NUMBER ONE

a) What type of data structure is being used to store the numbers in the code, and why is this data structure suitable for the task?

A list is the type of data structure being used to store the numbers in the code. It's suitable because it allows for efficient storage, retrieval, and manipulation of structured data.

b)The code initializes maxSum with the first element of the list. Can you explain
the significance of this initialization and how it affects the algorithm's correctness?

Initializing maxSum with the first element of the list ensures that there is a starting point for comparison. This initialization sets a baseline for the maximum subarray sum, allowing the algorithm to compare subsequent sums and update maxSum accordingly.

c) How does the for loop iterate through the list, and what is the purpose of the sumz variable? How is it being updated within the loop?

The for loop iterates through the list by starting from the second element. The sumz variable is used to keep track of the current sum of the subarray. Within the loop, sumz is updated by adding the current element to the previous sumz.

d) Explain how the code identifies the maximum subarray sum. What conditions trigger the update of the maxSum variable?

The code identifies the maximum subarray sum by comparing the current sumz with the maxSum. If the current sumz is greater than maxSum, maxSum is updated to the current sumz. This update occurs whenever a new subarray with a higher sum is found.

e)What is the time complexity of this code for finding the maximum subarray sum, and how does the choice of data structure and algorithm contribute to its efficiency or performance?

The time complexity of this code for finding the maximum subarray sum is O(n), where n is the size of the list. The choice of using a for loop to iterate through the list and a simple comparison to update maxSum contributes to the efficiency of the algorithm. By traversing the list once and updating maxSum only when necessary, the algorithm avoids unnecessary computations and achieves linear time complexity. Additionally, the use of a simple data structure like a list allows for efficient access and manipulation of elements, further contributing to the algorithm's performance.

QUESTION THREE

1. **C**
2. **B**
3. **B**
4. **A**
5. **D**
6. **A**
7. **A**
8. **D**

9. **B**
10. **A**