**Assignment – 6.2**

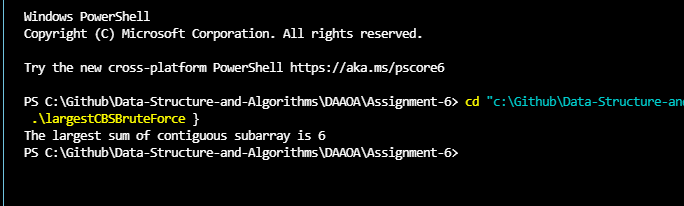
**Name:** Bhavin Patil

**Roll No.:** 66

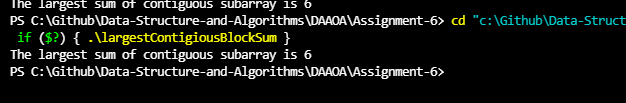
**Class:** TY CS-D

**Assignment based on Largest Contiguous Block Sum.**

1. **Brute Force Code**
   1. #include<bits/stdc++.h>
   2. using namespace std;
   3. int solve(vector<int>& nums) {
   4. int n = nums.size();
   5. int max\_sum = 0;
   6. // Iterate through all possible contiguous subarrays
   7. for (int i = 0; i < n; i++) {
   8. for (int j = i; j < n; j++) {
   9. int sum = 0;
   10. // Calculate the sum of the current subarray
   11. for (int k = i; k <= j; k++) {
   12. sum += nums[k];
   13. }
   14. // Update the maximum sum if the current sum is larger
   15. if (sum > max\_sum) {
   16. max\_sum = sum;
   17. }
   18. }
   19. }
   20. // Return the maximum sum found
   21. return max\_sum;
   22. }
   23. int main() {
   24. vector<int> nums = {-2, 1, -3, 4, -1, 2, 1, -5, 4};
   25. int largest\_sum = solve(nums);
   26. cout << "The largest sum of contiguous subarray is " << largest\_sum << endl;
   27. return 0;
   28. }
   29. // Time Complexxity - O(n^3)
2. **Output**



1. **Kadane’s Code**
   1. #include<bits/stdc++.h>
   2. using namespace std;
   3. int kadane(vector<int>& nums) {
   4. int n = nums.size();
   5. int max\_so\_far = nums[0];
   6. int max\_ending\_here = nums[0];
   7. for (int i = 1; i < n; i++) {
   8. max\_ending\_here = max(nums[i], max\_ending\_here + nums[i]);
   9. max\_so\_far = max(max\_so\_far, max\_ending\_here);
   10. }
   11. return max\_so\_far;
   12. }
   13. int main() {
   14. vector<int> nums = {-2, 1, -3, 4, -1, 2, 1, -5, 4};
   15. int largest\_sum = kadane(nums);
   16. cout << "The largest sum of contiguous subarray is " << largest\_sum << endl;
   17. return 0;
2. **Output**

****