**Worksheet-1.1**

**Student Name:-** Pushpraj Roy **UID:-** 20BCS9866

**Branch:-** BE- CSE **Section/Group:-** WM\_617 “A”

**Subjetct Code:-** 20CSP-314 **Semester:-** 5th

**Subject Name:-** Competitive Coding Lab

# Question 1-

**Aim –** Given an Array of Integers , find the sum of its elements.

**Input Code- Approach 1-**

/\* C++ Program to find sum of elements

in a given array \*/ #include <bits/stdc++.h> using namespace std;

// function to return sum of elements

// in an array of size n int sum(int arr[], int n)

{ int sum = 0; // initialize sum

// Iterate through all elements // and add them to sum for (int i = 0; i < n; i++) sum += arr[i];

return sum;

} int main()

{ int arr[] = {1 , 3 , 6 , 18 , 2}; int n = sizeof(arr) / sizeof(arr[0]);

cout << "Sum of given array is " << sum(arr, n); return 0;

}

# Approach 2-

/\* C++ Program to find sum of elements in a given array using recursion \*/

#include<iostream>

using namespace std;

// function to return sum of elements

// in an array of size n int sum(int arr[],int n)

{

//base case if(n==0){ return 0;

} else{

//recursively calling the function

return arr[0]+sum(arr+1,n-1);

} }

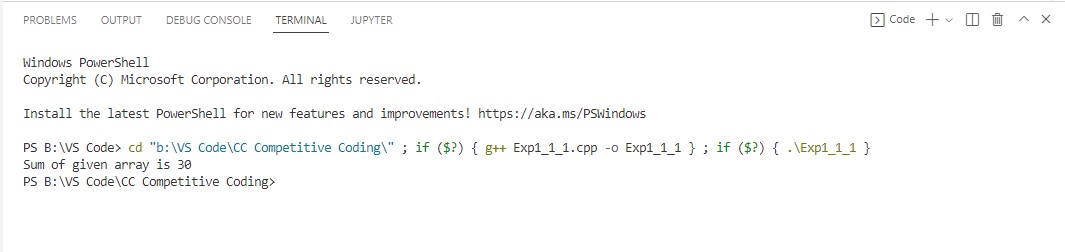
int main()

{ int arr[] = {1 , 3 , 6 , 18 , 2}; int n=sizeof(arr)/sizeof(arr[0]);

cout<<sum(arr,n);

return 0; }

**Output-**



**Time Complexity-** O(n) for Both Approaches but we will prefer Recursive Method.

# Question 2-

**Aim –**Program to Multiply two Matrices.

**Input Code- Approach 1-**

// C++ program to multiply two square matrices.

#include <iostream> using namespace std;

#define N 4

// This function multiplies

// mat1[][] and mat2[][], and // stores the result in res[][] void multiply(int mat1[][N], int mat2[][N],

int res[][N])

{

int i, j, k;

for (i = 0; i < N; i++) { for (j = 0; j < N; j++) { res[i][j] = 0; for (k = 0; k < N; k++)

res[i][j] += mat1[i][k] \* mat2[k][j];

}

}

}

int main()

{

int i, j;

int res[N][N]; // To store result int mat1[N][N] = { { 1, 2, 4, 6 },

{ 4, 3, 9, 7 },

{ 5, 3, 2, 6 },

{ 6, 1, 1, 3 } };

int mat2[N][N] = { { 1, 1, 1, 1 },

{ 2, 2, 2, 2 },

{ 3, 3, 3, 3 },

{ 4, 4, 4, 4 } };

multiply(mat1, mat2, res);

cout << "Result matrix is \n"; for (i = 0; i < N; i++) { for (j = 0; j < N; j++) cout << res[i][j] << " "; cout << "\n";

}

return 0;

}

**Output-**



**Time complexity:** O(n3). It can be optimized using Strassen’s Matrix Multiplication **Approach 2-**

#include<iostream> #include<vector> using namespace std;

int main(){ int r,c;

cin>>r>>c;

int arr1[r][c],arr2[r][c],ans[r][c];

for(int i=0;i<r;i++){ for(int j=0;j<c;j++){ cin>>arr1[i][j];

}

}

for(int i=0;i<r;i++){ for(int j=0;j<c;j++){ cin>>arr2[i][j];

}

}

for(int i=0;i<r;i++){ for(int j=0;j<c;j++){ ans[i][j]=0;

for(int k=0;k<c;k++){

ans[i][j]+=arr1[i][k]\*arr2[k][j];

}

}

}

for(int i=0;i<r;i++){ for(int j=0;j<c;j++){

cout<<ans[i][j]<<" ";

}

cout<<endl;

}

return 0;

}

**Output-**



**Learning outcomes (What I have learnt) -**

1.Using Iteration and Recursion Method to solve the problems.

2.Deeper Understanding of Matrix and Arrays and Use them to solve Problems.

3.To implement problems based on different algorithm design techniques.

4.To learn the importance of designing an algorithm in an effective way by considering space and time complexity.

**Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No. | Parameters | Marks Obtained | Maximum Marks |
| 1. |  |  |  |
| 2. |  |  |  |
| 3. |  |  |  |
|  |  |  |  |