**A Very Big Sum**

**Calculate and print the sum of the elements in an array, keeping in mind that some of those integers may be quite large.**

**Function Description**

**Complete the aVeryBigSum function in the editor below. It must return the sum of all array elements.**

**aVeryBigSum has the following parameter(s):**

**ar: an array of integers .**

**Input Format**

**The first line of the input consists of an integer .**

**The next line contains space-separated integers contained in the array.**

**Output Format**

**Print the integer sum of the elements in the array.**

**Sample Input**

**5**

**1000000001 1000000002 1000000003 1000000004 1000000005**

**Output**

**5000000015**

**Note:**

**When we add several integer values, the resulting sum might exceed the above range. You might need to use long long int in C/C++ or long data type in Java to store such sums.**

**Solution:**

**#include <bits/stdc++.h>**

**using namespace std;**

**vector<string> split\_string(string);**

**// Complete the aVeryBigSum function below.**

**long aVeryBigSum(vector<long> ar) {**

**long long sum=0;**

**for(int i=0; i < ar.size(); i++)**

**sum += ar[i];**

**return sum;**

**}**

**int main()**

**{**

**ofstream fout(getenv("OUTPUT\_PATH"));**

**int ar\_count;**

**cin >> ar\_count;**

**cin.ignore(numeric\_limits<streamsize>::max(), '\n');**

**string ar\_temp\_temp;**

**getline(cin, ar\_temp\_temp);**

**vector<string> ar\_temp = split\_string(ar\_temp\_temp);**

**vector<long> ar(ar\_count);**

**for (int i = 0; i < ar\_count; i++) {**

**long ar\_item = stol(ar\_temp[i]);**

**ar[i] = ar\_item;**

**}**

**long result = aVeryBigSum(ar);**

**fout << result << "\n";**

**fout.close();**

**return 0;**

**}**

**vector<string> split\_string(string input\_string) {**

**string::iterator new\_end = unique(input\_string.begin(), input\_string.end(), [] (const char &x, const char &y) {**

**return x == y and x == ' ';**

**});**

**input\_string.erase(new\_end, input\_string.end());**

**while (input\_string[input\_string.length() - 1] == ' ') {**

**input\_string.pop\_back();**

**}**

**vector<string> splits;**

**char delimiter = ' ';**

**size\_t i = 0;**

**size\_t pos = input\_string.find(delimiter);**

**while (pos != string::npos) {**

**splits.push\_back(input\_string.substr(i, pos - i));**

**i = pos + 1;**

**pos = input\_string.find(delimiter, i);**

**}**

**splits.push\_back(input\_string.substr(i, min(pos, input\_string.length()) - i + 1));**

**return splits;**

**}**