**#include** **<**assert.h**>**

**#include** **<**limits.h**>**

**#include** **<**math.h**>**

**#include** **<**stdbool.h**>**

**#include** **<**stddef.h**>**

**#include** **<**stdint.h**>**

**#include** **<**stdio.h**>**

**#include** **<**stdlib.h**>**

**#include** **<**string.h**>**

**char**\* readline();

**char**\*\* split\_string(**char**\*);

*// Complete the aVeryBigSum function below.*

**long** aVeryBigSum(**int** ar\_count, **long**\* ar) {

**long** **int** i,s=0;

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

{s=s+ar[i];}

**return** s;

}

**int** main()

{

    FILE\* fptr = fopen(getenv("OUTPUT\_PATH"), "w");

**char**\* ar\_count\_endptr;

**char**\* ar\_count\_str = readline();

**int** ar\_count = strtol(ar\_count\_str, &ar\_count\_endptr, 10);

**if** (ar\_count\_endptr == ar\_count\_str || \*ar\_count\_endptr != '\0') { exit(EXIT\_FAILURE); }

**char**\*\* ar\_temp = split\_string(readline());

**long**\* ar = malloc(ar\_count \* **sizeof**(**long**));

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

**char**\* ar\_item\_endptr;

**char**\* ar\_item\_str = \*(ar\_temp + i);

**long** ar\_item = strtol(ar\_item\_str, &ar\_item\_endptr, 10);

**if** (ar\_item\_endptr == ar\_item\_str || \*ar\_item\_endptr != '\0') { exit(EXIT\_FAILURE); }

        \*(ar + i) = ar\_item;

    }

**long** result = aVeryBigSum(ar\_count, ar);

    fprintf(fptr, "%ld\n", result);

    fclose(fptr);

**return** 0;

}

**char**\* readline() {

    size\_t alloc\_length = 1024;

    size\_t data\_length = 0;

**char**\* data = malloc(alloc\_length);

**while** (**true**) {

**char**\* cursor = data + data\_length;

**char**\* line = fgets(cursor, alloc\_length - data\_length, stdin);

**if** (!line) { **break**; }

        data\_length += strlen(cursor);

**if** (data\_length < alloc\_length - 1 || data[data\_length - 1] == '\n') { **break**; }

        size\_t new\_length = alloc\_length << 1;

        data = realloc(data, new\_length);

**if** (!data) { **break**; }

        alloc\_length = new\_length;

    }

**if** (data[data\_length - 1] == '\n') {

        data[data\_length - 1] = '\0';

    }

    data = realloc(data, data\_length);

**return** data;

}

**char**\*\* split\_string(**char**\* str) {

**char**\*\* splits = NULL;

**char**\* token = strtok(str, " ");

**int** spaces = 0;

**while** (token) {

        splits = realloc(splits, **sizeof**(**char**\*) \* ++spaces);

**if** (!splits) {

**return** splits;

        }

        splits[spaces - 1] = token;

        token = strtok(NULL, " ");

    }

**return** splits;

}