#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 maxSubsetSum function below.**

**int maxSubsetSum(int n, int\* arr)**

**{**

**int incl = arr[0];**

**int excl = 0;**

**int excl\_new;**

**int i;**

**for (i = 1; i < n; i++)**

**{**

**/\* current max excluding i \*/**

**excl\_new = (incl > excl)? incl: excl;**

**/\* current max including i \*/**

**incl = excl + arr[i];**

**excl = excl\_new;**

**}**

**/\* return max of incl and excl \*/**

**return ((incl > excl)? incl : excl);**

**}**

int main()

{

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

    char\* n\_endptr;

    char\* n\_str = readline();

    int n = strtol(n\_str, &n\_endptr, 10);

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

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

    int\* arr = malloc(n \* sizeof(int));

    for (int i = 0; i < n; i++) {

        char\* arr\_item\_endptr;

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

        int arr\_item = strtol(arr\_item\_str, &arr\_item\_endptr, 10);

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

        \*(arr + i) = arr\_item;

    }

    int arr\_count = n;

    int res = maxSubsetSum(arr\_count, arr);

    fprintf(fptr, "%d\n", res);

    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;

        }

        alloc\_length <<= 1;

        data = realloc(data, alloc\_length);

        if (!line) {

            break;

        }

    }

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

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

        data = realloc(data, data\_length);

    } else {

        data = realloc(data, data\_length + 1);

        data[data\_length] = '\0';

    }

    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;

}