#include <assert.h>

#include <ctype.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\* ltrim(char\*);

char\* rtrim(char\*);

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

int parse\_int(char\*);

/\*

\* Complete the 'largestRectangle' function below.

\*

\* The function is expected to return a LONG\_INTEGER.

\* The function accepts INTEGER\_ARRAY h as parameter.

\*/

long largestRectangle(int h\_count, int\* h) {

int\* stack = (int\*)malloc(h\_count \* sizeof(int));

int top = -1;

long max\_area = 0;

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

while (top != -1 && (i == h\_count || h[stack[top]] > h[i])) {

int height = h[stack[top--]];

int width = (top == -1) ? i : (i - stack[top] - 1);

long area = (long)height \* width;

if (area > max\_area) {

max\_area = area;

}

}

if (i < h\_count) {

stack[++top] = i;

}

}

free(stack);

return max\_area;

}

int main()

{

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

int n = parse\_int(ltrim(rtrim(readline())));

char\*\* h\_temp = split\_string(rtrim(readline()));

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

for (int i = 0; i < n; i++) {

int h\_item = parse\_int(\*(h\_temp + i));

\*(h + i) = h\_item;

}

long result = largestRectangle(n, h);

fprintf(fptr, "%ld\n", result);

fclose(fptr);

free(h);

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

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

}

return data;

}

char\* ltrim(char\* str) {

while (\*str != '\0' && isspace(\*str)) {

str++;

}

return str;

}

char\* rtrim(char\* str) {

char\* end = str + strlen(str) - 1;

while (end >= str && isspace(\*end)) {

end--;

}

\*(end + 1) = '\0';

return str;

}

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

char\*\* splits = NULL;

char\* token = strtok(str, " ");

int spaces = 0;

while (token) {

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

splits[spaces - 1] = token;

token = strtok(NULL, " ");

}

return splits;

}

int parse\_int(char\* str) {

return strtol(str, NULL, 10);

}