#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 kangaroo function below.

// Please either make the string static or allocate on the heap. For example,

// static char str[] = "hello world";

// return str;

//

// OR

//

// char\* str = "hello world";

// return str;

//

static char str1[] ="YES";

static char str2[] = "NO";

char\* kangaroo(int x1, int v1, int x2, int v2) {

int i;

for(i=0;i<10000;i++)

{

if((x1+v1\*i)==(x2+v2\*i))

{

return str1;

}

}

return str2;

}

int main()

{

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

char\*\* x1V1X2V2 = split\_string(readline());

char\* x1\_endptr;

char\* x1\_str = x1V1X2V2[0];

int x1 = strtol(x1\_str, &x1\_endptr, 10);

if (x1\_endptr == x1\_str || \*x1\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* v1\_endptr;

char\* v1\_str = x1V1X2V2[1];

int v1 = strtol(v1\_str, &v1\_endptr, 10);

if (v1\_endptr == v1\_str || \*v1\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* x2\_endptr;

char\* x2\_str = x1V1X2V2[2];

int x2 = strtol(x2\_str, &x2\_endptr, 10);

if (x2\_endptr == x2\_str || \*x2\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* v2\_endptr;

char\* v2\_str = x1V1X2V2[3];

int v2 = strtol(v2\_str, &v2\_endptr, 10);

if (v2\_endptr == v2\_str || \*v2\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* result = kangaroo(x1, v1, x2, v2);

fprintf(fptr, "%s\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;

}