#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 howManyGames function below.

int howManyGames(int p, int d, int m, int s) {

// Return the number of games you can buy

int loop\_count;

loop\_count = s/m;

int i;

int item\_count;

item\_count =0;

int left\_money;

left\_money = s;

for(i=0;i<loop\_count;i++)

{

int temp = p-d\*i;

printf("%d",temp);

if(temp > m)

{

if(left\_money >= temp && left\_money > 0)

{

left\_money = left\_money - temp;

item\_count++;

printf("\n\*\*\*\*lp1 %d",left\_money);

}

else if(left\_money < temp)

{

break;

}

}

else {

if(left\_money >= m && left\_money>0)

{

left\_money = left\_money - m;

item\_count++;

printf("\n\*\*\*\*lp2 %d",left\_money);

}

}

printf("\n %d",left\_money);

}

return item\_count;

}

int main()

{

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

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

char\* p\_endptr;

char\* p\_str = pdms[0];

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

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

char\* d\_endptr;

char\* d\_str = pdms[1];

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

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

char\* m\_endptr;

char\* m\_str = pdms[2];

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

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

char\* s\_endptr;

char\* s\_str = pdms[3];

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

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

int answer = howManyGames(p, d, m, s);

fprintf(fptr, "%d\n", answer);

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;

}