#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 libraryFine function below.

int libraryFine(int d1, int m1, int y1, int d2, int m2, int y2) {

if(y1==y2)

{

if(m1==m2 && d1>d2)

{

return (d1-d2)\*15;

}

else if(m1>m2)

{

return (m1-m2)\*500;

}

}

else if(y1>y2)

{

return 10000;

}

return 0;

}

int main()

{

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

char\*\* d1M1Y1 = split\_string(readline());

char\* d1\_endptr;

char\* d1\_str = d1M1Y1[0];

int d1 = strtol(d1\_str, &d1\_endptr, 10);

if (d1\_endptr == d1\_str || \*d1\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* m1\_endptr;

char\* m1\_str = d1M1Y1[1];

int m1 = strtol(m1\_str, &m1\_endptr, 10);

if (m1\_endptr == m1\_str || \*m1\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* y1\_endptr;

char\* y1\_str = d1M1Y1[2];

int y1 = strtol(y1\_str, &y1\_endptr, 10);

if (y1\_endptr == y1\_str || \*y1\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\*\* d2M2Y2 = split\_string(readline());

char\* d2\_endptr;

char\* d2\_str = d2M2Y2[0];

int d2 = strtol(d2\_str, &d2\_endptr, 10);

if (d2\_endptr == d2\_str || \*d2\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* m2\_endptr;

char\* m2\_str = d2M2Y2[1];

int m2 = strtol(m2\_str, &m2\_endptr, 10);

if (m2\_endptr == m2\_str || \*m2\_endptr != '\0') { exit(EXIT\_FAILURE); }

char\* y2\_endptr;

char\* y2\_str = d2M2Y2[2];

int y2 = strtol(y2\_str, &y2\_endptr, 10);

if (y2\_endptr == y2\_str || \*y2\_endptr != '\0') { exit(EXIT\_FAILURE); }

int result = libraryFine(d1, m1, y1, d2, m2, y2);

fprintf(fptr, "%d\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;

}