#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();

/\*

 \* Complete the 'timeConversion' function below.

 \*

 \* The function is expected to return a STRING.

 \* The function accepts STRING s as parameter.

 \*/

/\*

 \* To return the string from the function, you should either do static allocation or dynamic allocation

 \*

 \* For example,

 \* char\* return\_string\_using\_static\_allocation() {

 \*     static char s[] = "static allocation of string";

 \*

 \*     return s;

 \* }

 \*

 \* char\* return\_string\_using\_dynamic\_allocation() {

 \*     char\* s = malloc(100 \* sizeof(char));

 \*

 \*     s = "dynamic allocation of string";

 \*

 \*     return s;

 \* }

 \*

 \*/

char\* timeConversion(char\* s) {

        static char result[9];

        int hour;

        char period[3];

        sscanf(s,"%2d:%\*2d:%\*2d%2s",&hour,period);

        if(strcmp(period,"AM")==0){

            if(hour==12) hour=0;

        }

        else{

            if(hour !=12) hour+=12;

        }

        snprintf(result,sizeof(result),"%02d%s",hour,s+2);

        result[8]='\0';

        return result;

    }

int main()

{

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

    char\* s = readline();

    char\* result = timeConversion(s);

    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;

        }

        alloc\_length <<= 1;

        data = realloc(data, alloc\_length);

        if (!data) {

            data = '\0';

            break;

        }

    }

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

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

        data = realloc(data, data\_length);

        if (!data) {

            data = '\0';

        }

    } else {

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

        if (!data) {

            data = '\0';

        } else {

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

        }

    }

    return data;

}