En esta práctica se pueden usar los **elementos de C permitidos** en prácticas anteriores, y hay que seguir teniendo en cuenta las **restricciones** de funcionamiento de esas prácticas.

Esta práctica está diseñada para seguir ejercitando el uso de **arrays**.

**De momento, no está permitido** el uso de strings, u otros elementos del lenguaje C.

Se deben **usar todos y cada uno de los prototipos** indicados en esta práctica y los de prácticas anteriores que se requieran.

En la práctica actual, se debe generar aleatoriamente los DNIs, calculándose la letra, y a continuación printarlos. A continuación, se debe **ordenar los DNIs ascendentemente** y printar el resultado de la ordenación. Se usará el algoritmo “bubble sort”, ineficiente pero didáctico. Se **utilizará** **asignación dinámica de memoria** en la función *main* y en las funciones *swap*.

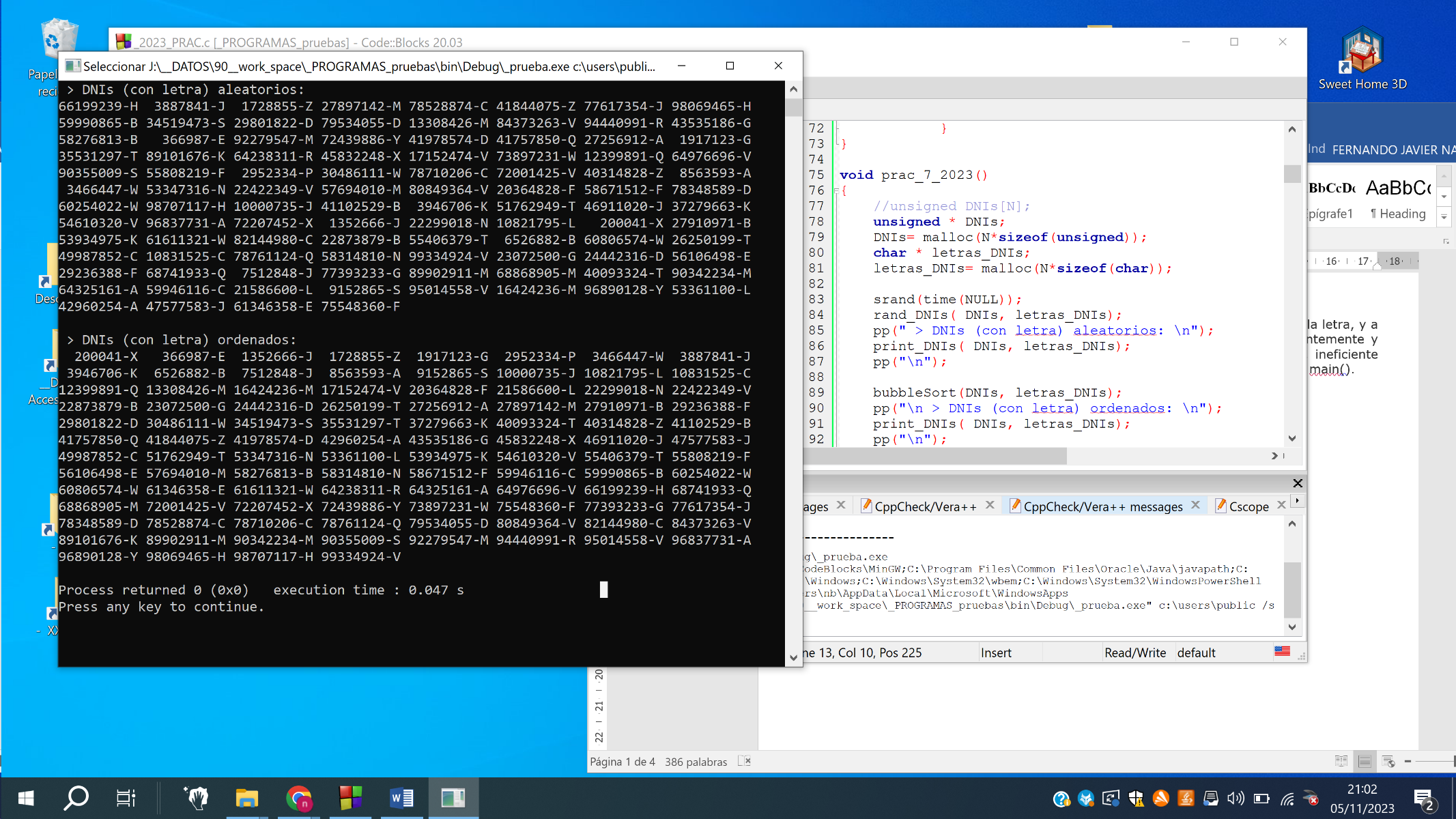


Figura 1. Ejemplo de ejecución del programa

El algoritmo **bubble sort** se puede visualizar en https://www.google.com/search?q=bubble+sort&sca\_esv=579651652&rlz=1C1RXQR\_esES974ES1028&tbm=vid&lr=lang\_es&sa=X&ved=2ahUKEwiF36veyq2CAxWxRqQEHeJoA8kQuAF6BAgaEAI&biw=919&bih=506&dpr=2.77#fpstate=ive&vld=cid:7f1bafe4,vid:pqZ04TT15PQ,st:0

//includes

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

//defines

#define N 100

//prototipos usados en practicas anteriores

unsigned resto\_DNI(unsigned );

char letra\_calculada(unsigned );

void rand\_dig (char \*);

void rand\_DNI (unsigned \*);

void rand\_DNIs (unsigned [N], char [N]);

void print\_DNIs (unsigned [N], char [N]);

//prototipos de esta práctica

void swap\_unsigned(unsigned\*, unsigned\*);

void swap\_char(char\*, char\*);

void bubbleSort(unsigned [N], char [N]);

//main

i int main()

{

unsigned \*ArrayDNIS;

ArrayDNIS = (unsigned \*)malloc(N \* sizeof(unsigned));

char \*LetrasDNIS;

LetrasDNIS= (char \*)malloc(N \* sizeof(char));

srand(time(NULL));

rand\_DNIs(ArrayDNIS, LetrasDNIS);

print\_DNIs(ArrayDNIS, LetrasDNIS);

printf("\nDNIs Ordenados: \n");

bubbleSort(ArrayDNIS, LetrasDNIS);

print\_DNIs(ArrayDNIS, LetrasDNIS);

free(ArrayDNIS);

free(LetrasDNIS);

return 0;

}

// definición de las funciones

void rand\_dig(char\* dig) {

\*dig = '0' + (rand() % 10);

}

void rand\_DNI(unsigned\* DNI) {

\*DNI = 0;

char digito;

for (int i = 1; i <= 8; i++) {

rand\_dig(&digito);

\*DNI = \*DNI \* 10 + (digito - '0');

}

}

void rand\_DNIs(unsigned ArrayDNIS[N], char LetrasDNIS[N]) {

unsigned DNI = 0;

int resto, letraCalculada;

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

rand\_DNI(&DNI);

ArrayDNIS[i] = DNI;

resto = resto\_DNI(DNI);

letraCalculada = letra\_calculada(resto);

LetrasDNIS[i] = letraCalculada;

}

}

unsigned resto\_DNI(unsigned DNI) {

return DNI % 23;

}

char letra\_calculada(unsigned resto) {

char letraCalculadaDNI[] = {'T', 'R', 'W', 'A', 'G', 'M', 'Y', 'F', 'P', 'D', 'X',

'B', 'N', 'J', 'Z', 'S', 'Q', 'V', 'H', 'L', 'C', 'K', 'E'};

return letraCalculadaDNI[resto];

}

void print\_DNIs(unsigned ArrayDNIS[N], char LetrasDNIS[N]) {

printf(" >DNIs Y Letras: \n");

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

printf("%08u - %c\t ", ArrayDNIS[i], LetrasDNIS[i]);

}

}

void swap\_unsigned(unsigned \*DNIAnt, unsigned \*DNISig){

int aux;

aux = \*DNIAnt;

\*DNIAnt = \*DNISig;

\*DNISig = aux;

}

void swap\_char(char \*LetraAnt, char \*LetraSig) {

char aux;

aux = \*LetraAnt;

\*LetraAnt = \*LetraSig;

\*LetraSig = aux;

}

void bubbleSort(unsigned ArrayDNIs[N], char LetrasDNIs[N]) {

int i,j;

for (i = N - 1; i >= 0; i--) {

for (j = 0; j < i; j++) {

if (ArrayDNIs[j] > ArrayDNIs[j + 1]) {

swap\_unsigned(&ArrayDNIs[j], &ArrayDNIs[j + 1]);

swap\_char(&LetrasDNIs[j], &LetrasDNIs[j + 1]);

}

}

}

}