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 los prototipos** indicados y sólo esos prototipos.

En C se pueden generar números aleatorios, incluyendo dos bibliotecas: “stdlib.h” y “time.h”. Lo primero es inicializar el proceso de generación, mediante srand(time(NULL)). Lo segundo, usar rand() para generar un número aleatorio concreto. Si queremos que ese valor aleatorio se genere en el intervalo [0,**rango**-1], usaremos rand()%**rango**.

En la práctica actual, se deben generar aleatoriamente los DNIs, calculándose la letra, para a continuación printarlos. Por último, para las letras de los DNIs, se deben calcular las frecuencias absolutas y relativas, printarlas y dibujar su diagrama de barras.

La función *letra\_calculada (unsigned)* se continuará definiendo con un array de chars (sin if ni switch). En esta ocasión, **no** se utilizará **asignación dinámica de memoria** (malloc, free).

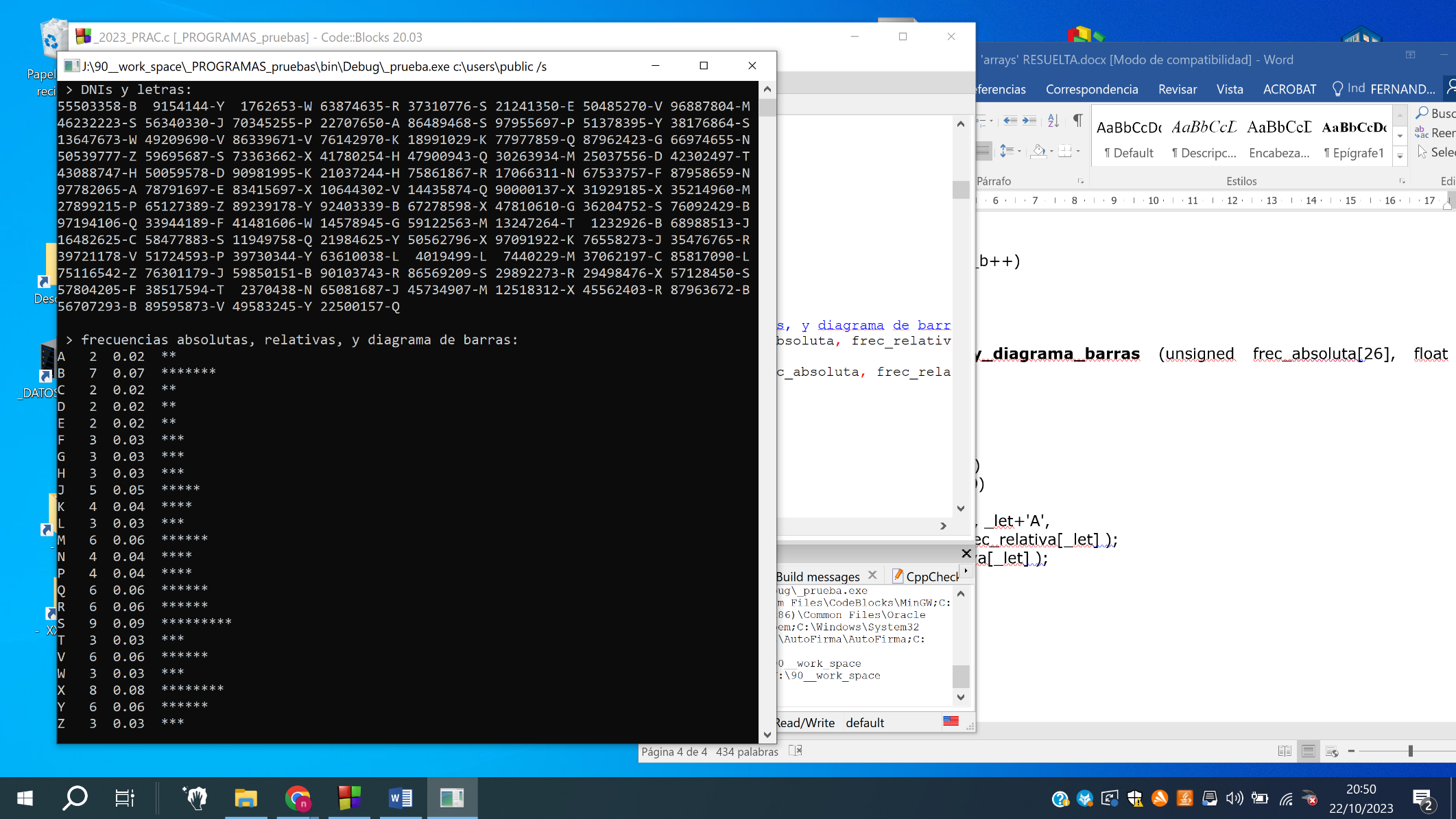


Figura 1. Ejemplo de ejecución del programa

//includes

#include ”stdio.h”

#include ”time.h” // time()

#include ”stdlib.h” // srand(), rand()

//defines

#define N 100

//prototipos usados en practicas anteriores

unsigned resto\_DNI(unsigned );

char letra\_calculada(unsigned );

//prototipos de esta práctica

void rand\_dig (char \*);

void rand\_DNI (unsigned \*);

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

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

void calcular\_frecuencias (char [N], unsigned [26], float [26]);

void print\_frecuencias (unsigned [26], float [26]);

void print\_barra (float);

//main

int main()

{

unsigned ArrayDNIS[N],Frecuencias\_Absolutas[23] = {0};

char LetrasDNIS[N];

float Frecuencias\_Relativas [23] = {0};

srand(time(NULL));

rand\_DNIs(ArrayDNIS, LetrasDNIS);

calcular\_frecuencias(LetrasDNIS,Frecuencias\_Absolutas,Frecuencias\_Relativas);

print\_frecuencias(Frecuencias\_Absolutas,Frecuencias\_Relativas);

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;

}

print\_DNIs(ArrayDNIS, LetrasDNIS);

}

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 - %-10c", ArrayDNIS[i], LetrasDNIS[i]);

}

}

void calcular\_frecuencias(char LetrasDNIS[N], unsigned Absolutas[23], float Relativas[23]) {

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

if (LetrasDNIS[i] >= 'A' && LetrasDNIS[i] <= 'W') {

Absolutas[LetrasDNIS[i] - 'A']++;

}

}

for (int i = 0; i < 23; i++) {

Relativas[i] = (float)Absolutas[i] / N \* 100;

}

}

void print\_barra(float barras) {

int veces = barras;

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

printf("\*");

}

}

void print\_frecuencias(unsigned Absoluto[23], float Relativo[23]) {

char letras[23] = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'J', 'K', 'L',

'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'V', 'W', 'X', 'Y', 'Z'};

printf("\n > frecuencias absolutas, relativa y diagramas de barra:\n");

for (int i = 0; i < 23; i++) {

printf("%c %2u %.2f ", letras[i], Absoluto[i], Relativo[i]/100);

print\_barra(Relativo[i]);

printf("\n");

}

}