

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*.

```
> DNIs (con letra) aleatorios:
66199239-H 3887841-J 1728855-Z 27897142-M 78528874-C 41844075-Z 77617354-J 98069465-H
59990865-B 34519473-S 29801822-D 79534055-D 13308426-M 84373263-V 94440991-R 43535186-G
58276813-B 366987-E 92279547-M 72439886-Y 41978574-D 41757850-Q 27256912-A 1917123-G
35531297-T 89101676-K 64238311-R 45832248-X 17152474-V 73897231-W 12399891-Q 64976696-V
90355009-S 55808219-F 2952334-P 30486111-W 78710206-C 72001425-V 40314828-Z 8563593-A
3466447-W 53347316-N 22422349-V 57694010-M 80849364-V 20364828-F 58671512-F 78348589-D
60254022-W 98707117-H 10000735-J 41102529-B 3946706-K 51762949-T 46911020-J 37279663-K
54610320-V 96837731-A 72207452-X 1352666-J 22299018-N 10821795-L 200041-X 27910971-B
53934975-K 61611321-W 82144980-C 22873879-B 55406379-T 6526882-B 60806574-W 26250199-T
49987852-C 10831525-C 78761124-Q 58314810-N 99334924-V 23072500-G 24442316-D 56106498-E
29236388-F 68741933-Q 7512848-J 77393233-G 89902911-M 68868905-M 40093324-T 90342234-M
64325161-A 59946116-C 21586600-L 9152865-S 95014558-V 16424236-M 96890128-Y 53361100-L
42960254-A 47577583-J 61346358-E 75548360-F

> DNIs (con letra) ordenados:
200041-X 366987-E 1352666-J 1728855-Z 1917123-G 2952334-P 3466447-W 3887841-J
3946706-K 6526882-B 7512848-J 8563593-A 9152865-S 10000735-J 10821795-L 10831525-C
12399891-Q 13308426-M 16424236-M 17152474-V 20364828-F 21586600-L 22299018-N 22422349-V
22873879-B 23072500-G 24442316-D 26250199-T 27256912-A 27897142-M 27910971-B 29236388-F
29801822-D 30486111-W 34519473-S 35531297-T 37279663-K 40093324-T 40314828-Z 41102529-B
41757850-Q 41844075-Z 41978574-D 42960254-A 43535186-G 45832248-X 46911020-J 47577583-J
49987852-C 51762949-T 53347316-N 53361100-L 53934975-K 54610320-V 55406379-T 55808219-F
56106498-E 57694010-M 58276813-B 58314810-N 58671512-F 59946116-C 59990865-B 60254022-W
60806574-W 61346358-E 61611321-W 64238311-R 64325161-A 64976696-V 66199239-H 68741933-Q
68868905-M 72001425-V 72207452-X 72439886-Y 73897231-W 75548360-F 77393233-G 77617354-J
78348589-D 78528874-C 78710206-C 78761124-Q 79534055-D 80849364-V 82144980-C 84373263-V
89101676-K 89902911-M 90342234-M 90355009-S 92279547-M 94440991-R 95014558-V 96837731-A
96890128-Y 98069465-H 98707117-H 99334924-V
```

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_e sES974ES1028&tbm=vid&lr=lang_es&sa=X&ved=2ahUKEwiF36veyq2CAxWxRqQEHeJoA 8kQuAF6BAgaEAI&biw=919&bih=506&dpr=2.77#fpstate=ive&vld=cid:7f1baf4e4,vid:pqZ 04TT15PQ,st:0

```
//defines
```

```
//prototipos usados en practicas anteriores
```

```
void rand_dig (char *);  
void rand_DNI (unsigned *);  
void rand_DNIs (unsigned [N], char [N]);  
void print_DNIs (unsigned [N], char [N]);  
unsigned resto_DNI(unsigned);  
char letra_calculada(unsigned );
```

```
//prototipos de esta práctica
```

```
void swap_unsigned(unsigned*, unsigned*);  
void swap_char(char*, char*);  
void bubbleSort(unsigned [N], char [N]);
```

```
//main
```

```
int main()  
{ unsigned *DNIs = malloc(N*sizeof(unsigned));  
  char *letras = malloc(N*sizeof(char));  
  *DNIs;  
  
  srand(time(NULL));  
  rand_DNIs(DNIs,letras);  
  printf("> DNIs (con letra) aleatorios:\n");  
  print_DNIs(DNIs,letras);  
  bubbleSort(DNIs, letras);  
  printf("\n> DNIs (con letra) ordenados:\n");  
  print_DNIs(DNIs,letras);  
  
  free(DNIs);  
  free(letras);  
  return 0;  
}
```

```
// definición de las funciones
```

```
void rand_dig (char *digito){  
  *digito=rand()%10+'0';  
}  
  
void rand_DNI (unsigned *dni){  
  int i;  
  char digito;  
  for(i=1;i<=8;i++){  
    rand_dig(&digito);  
    *dni = (*dni*10) + (digito - '0');  
  }
```

```
}
```

```
}
```

```
void rand_DNIs (unsigned DNIs[N], char letras[N]){
    unsigned dni=0, i, resto;
    char letra;
    for(i=0;i<N;DNIs[i]=dni, letras[i]= letra, i++, dni=0){
        rand_DNI(&dni);
        resto = resto_DNI(dni);
        letra=letra_calculada(resto);
    }
}
```

```
unsigned resto_DNI(unsigned num){
    return (num%23);
}
```

```
char letra_calculada(unsigned resto){
    char letra[23]
    {'T','R','W','A','G','M','Y','F','P','D','X','B','N','J','Z','S','Q','V','H','L','C','K','E'};
    return letra[resto];
}
```

=

```
void print_DNIs (unsigned dnis[N], char letras[N]){
    int i;
    for(i=0;i<N;i++)
        printf("%08u-%c ", dnis[i],letras[i]); //El 08u muestra tambien el 0
}
```

```
void bubbleSort(unsigned DNIs[N], char letras[N]){
    int i,j, aux;

    for(i=0;i<N;i++){
        for(j=N-1;j>i;j--){
            if(DNIs[j]<DNIs[j-1]){
                swap_unsigned(&DNIs[j],&DNIs[j-1]);
                swap_char(&letras[j],&letras[j-1]);
            }
        }
    }
}
```

```

}

void swap_unsigned(unsigned *menor, unsigned *mayor){
    unsigned *aux = malloc(sizeof(unsigned));

```

```
*aux = *menor;
*menor = *mayor;
*mayor = *aux;
free(aux);
}
void swap_char(char *menor, char *mayor){
char *aux = malloc(sizeof(char));
*aux = *menor;
*menor = *mayor;
*mayor = *aux;
free(aux);
}
```