IDENTIFICADOR ESTUDIANTE

FUNDAMENTOS DE PROGRAMACIÓN
EXAMEN FINAL SEGUNDO PARCIAL
24 ENERO 2023

APELLIDOS (en MAYÚSCULAS)

NOMBRE (en MAYÚSCULAS)

Los smartwatches y smartphones deben estar apagados. Cada hoja se entrega por separado y es obligatorio entregar todas las hojas. Se debe poner el ***nombre completo en todas las hojas y el identificador del estudiante****. No se podrá entregar hasta pasados* ***20 minutos****. No se pueden usar hojas de sucio, para ello se dispone del espacio sobrante en cada hoja. El examen tendrá una duración máxima de 1:30 h.* ***No se podrá salir de los bucles usando break o return****.*

1.- (1.5 ptos) Escribir un programa que lea números enteros sin signo contenidos en un fichero de nombre *numeros.dat* e imprima en pantalla el número leído y el resultante de poner a 1 los *dos bits* de más a la derecha de cada byte del entero. Por ejemplo, si el número es el 1238, cuya representación en binario es:

00000*00*	**00000*00***	**00001*00***	**11010*10***

tras la modificación debería quedar:

00000*11*	**00000*11***	**00001*11***	**11010*11***

```
int main()
{ unsigned i, num, masc=0;
  FILE *f_in;

  if (!(f_in = fopen("numeros.dat", "rb")))
    { printf("Error en la apertura del fichero.\n");
      return (1);
    }

  for (i=0; i<4; i++, masc<<=8)
     masc |= 3;

  while (fread(&num, sizeof(unsigned), 1, f_in))
  {  printf("%u\n", num);
     num |= masc;
     printf ("%u\n\n", num);
  }

  fclose(f_in);
  return (0);
}
```

2.- (1.5 ptos) Escribir un programa que, recibiendo como argumentos un fichero de texto y una palabra, grabe en otro fichero, de nombre *salida.txt*, todas las líneas del primero en las que no aparezca la palabra pasada como argumento. Las líneas en el fichero de salida deberán ir numeradas, empezando por la 1.

```
int main(int argc, char *argv[])
{ unsigned pos=1;
  char linea[256];
  FILE *f_in, *f_out;

  if (argc<3)
    { printf("Faltan parámetros.\n");
      return (1);
    }

  if (!(f_in = fopen(argv[1], "r")))
    { printf("Error en la apertura del fichero.\n");
      return (2);
    }
  if (!(f_out = fopen("salida.txt", "w")))
    { printf("Error en la apertura del fichero.\n");
      return (3);
    }

  while (fgets(linea, 256, f_in))
   if (! strstr(linea, argv[2]))
     fprintf (f_out, "%d %s", pos++, linea);

  fclose(f_in);
  fclose(f_out);
  return (0);
}
```

3.- (1.5 ptos) Escribir una función que, recibiendo como parámetros dos matrices de enteros, la primera de LxM y la segunda de MxN (siendo L, M y N constantes definidas previamente), imprima por pantalla el producto de las dos matrices. NO SE PERMITE EL USO DE MATRICES AUXILIARES.

```
void multip_matr (int matr1[L][M], int matr2[M][N])
{  int i, j, k, suma;

   for (i=0; i<L; i++)
   {  for (j=0; j<N; j++)
      {  for (k=0, suma=0; k<M; k++)
            suma += matr1[i][k] * matr2[k][j];
         printf ("%d   ", suma);
      }
      printf ("\n");
   }
}
```

IDENTIFICADOR ESTUDIANTE

FUNDAMENTOS DE PROGRAMACIÓN
EXAMEN FINAL SEGUNDO PARCIAL
24 ENERO 2023

APELLIDOS (en MAYÚSCULAS)

NOMBRE (en MAYÚSCULAS)

4.- (1.5 ptos) Dada la definición de una lista de enteros como se muestra a continuación:

```
typedef struct nodo
        { int n;
          struct nodo *sig;
        } NODO;
typedef NODO * LISTA;
```

Escribir una función que, pasándole como parámetros una lista como la definida y dos números enteros que representan un intervalo (pueden pasarse desordenados), elimine de la lista todos los nodos cuyo valor esté comprendido en el intervalo, incluyendo los límites.

```
void elim_interv (LISTA *l, int inf, int sup)
{  int n;
   LISTA act=*l, ant=act, aux;

   if (inf>sup)
     { n = inf;
       inf = sup;
       sup = n;
     }

   while (act)
    if (act->n >= inf && act->n <= sup)
     {  aux = act;
        if (aux == *l)
          { *l = act->sig;
            act = ant =*l;
          }
        else
          { ant-> sig = act->sig;
            act = act->sig;
          }
        free (aux);
     }
    else
      { ant = act;
        act = act->sig;
      }
}
```