1. Sea un computador con memoria cache con las características siguientes:
- Memoria física de 32 KB
- Memoria cache de 512 bytes con bloques de 128 bytes y emplazamiento directo

a) Indique el formato de la dirección para Memoria Principal teniendo en cuenta las características de la MC
b) Si en un momento dado el directorio de la cache tiene los contenidos (en hexadecimal) indicados en la tabla, exprese en hexadecimal el rango de direcciones físicas ubicadas en cada bloque de la memoria cache.
c) Suponiendo que un programa realiza la siguiente cadena de referencias indique el número de aciertos que se producen: de 0x2080 a 0x209F, de 0x2880 a 0x289F y de 0x03F0 a 0x0410.

Etiqueta	Bloque
35	0
10	1
10	2
08	3

2. Se dispone de un computador con una memoria direccionable por bytes y dotado de una cache de 256 bytes. Cada bloque es de 16 bytes.
a) Si la cache es de emplazamiento directo, e inicialmente está vacía, indique para la siguiente secuencia de referencias a memoria si se trata de un acierto o un fallo: 0xA01, 0xB1F, 0x70A, 0x60F, 0xA70, 0xB11, 0xA7A, 0xA0B, 0x67A, 0xA7F, 0x071, 0x67F. Indique además qué fallos producen reemplazamiento.
b) Si la cache es asociativa por conjuntos con dos vías, con reemplazamiento LRU, indique para la misma secuencia de referencias del apartado anterior si se producen reemplazamientos.

3. Se va a ejecutar el código que se muestra a continuación en una máquina con direcciones de 16 bits que dispone de una memoria cache de datos de acceso directo con 4 bloques de 16 bytes cada uno y política de escritura write-back con asignación en escritura. Teniendo en cuenta sólo la cache de datos (es decir, ignorando la lectura de instrucciones):
a) Ignorando los accesos a la variable *tmp* (estará en registro), ¿cuántos fallos de cache se producirán? Justifique la respuesta, indicando además cuántos reemplazamientos se producen y en qué iteraciones.
b) Recalcule el número de fallos si la cache fuese asociativa con 2 vías y la política de reemplazamiento fuese LRU.

```
//Cada entero ocupa 4 bytes
int v[16];                      // Dirección de comienzo 0x0000
int mask[4];                    // Dirección de comienzo 0x0040

for(i=0; i<16; i++) {
      tmp = 0;
      for(j=0; j<4; j++){
         tmp = tmp + mask[j]*v[i];
      }
      v[i]=tmp;
}
```

4. Sea una memoria principal de 32M bytes direccionable por bytes a la que se dota de una memoria cache con las siguientes características:
- Tamaño de 2K bytes
- Líneas (bloques) de 256 bytes,
- Prebúsqueda bajo fallo (en caso de fallo se trae el bloque que lo provoca y el siguiente)
- Algoritmo de reemplazamiento FIFO.

Sea la secuencia de acceso a memoria principal dada por las siguientes direcciones: 0x00023FA, 0x00014A2, 0x0003F02, 0x00040B1, 0x0005572 y 0x00023AA. Muestre la evolución del directorio cache indicando los fallos (F), las prebúsquedas (P) y los aciertos (A) que se producen suponiendo:

a) emplazamiento directo
b) emplazamiento asociativo por conjuntos de 2 vías

5. Considere un computador con una memoria principal de 4MB, direccionable por bytes, al que se dota de una memoria cache de 2KB, con bloques de 512B. La memoria cache es de emplazamiento directo y con asignación de escritura. Se quiere ejecutar el siguiente código:

```
for(i=0;i<1024;i++) {
    C[i]=A[i]-B[1023-i];
}
```

Los datos son enteros y están almacenados desde la dirección 0x200000

a) ¿Cuántos fallos de cache se producen?
b) ¿Cuántos fallos se producen si aplicamos fusión de arrays? ¿Importa el orden de la fusión?
c) ¿Cuántos fallos se producen si aplicamos alargamiento de arrays? ¿Importa la cantidad "alargada"?

6. Considere un computador con direcciones de 32 bits, al que se dota de una memoria cache de 4KB, con líneas (bloques) de 16 bytes, de emplazamiento directo y con asignación de escritura.

Se quiere ejecutar el siguiente código:

```
int A[1024]; // A[0] están en la dirección 0x0C000000
int B[1024];
int C[1024];

for (i=0;i<10;i++) {
    for(j=0;j<1024;j++) {
        C[i] += A[j]*B[j];
    }
}
```

a) ¿Cuántos fallos de cache se producen?
b) ¿Qué asociatividad necesitamos (como mínimo) para evitar todos los fallos por conflicto con este código?
c) Siguiendo con emplazamiento directo, ¿qué modificaciones de código podríamos hacer para evitar todos los fallos por conflicto?

7. Sea un computador con una memoria principal de 64K bytes, direccionable en bytes, al que se dota de una memoria cache de 2K bytes, con líneas de 256 bytes, y prebúsqueda bajo fallo (en caso de fallo se trae el bloque que lo provoca y el siguiente).

Sea la secuencia de acceso a memoria principal dada por las siguientes direcciones: 0x3345, 0x1244, 0x3374, 0x5BAC, 0x132A, 0x5C41 y 0x13AF.

Muestre la evolución de las etiquetas de la cache indicando los fallos y las prebúsquedas que se producen en los siguientes casos:

a) Una organización directa
b) Una organización asociativa por conjuntos de 2 vías, con algoritmo de reemplazo LRU
c) Una organización directa y un buffer de 512 bytes donde se almacena inicialmente el bloque prebuscado hasta que se le referencia. El buffer actúa con emplazamiento asociativo y política FIFO.

8. Considere un computador con una memoria principal de 128MB, direccionable por bytes, al que se dota de una memoria cache de 1MB, con líneas de 256B, y cache de víctima de 1KB. La memoria cache tiene acceso directo, mientras que la cache víctima es completamente asociativa. Para ambas el algoritmo de reemplazamiento es FIFO. Sea la secuencia de acceso a memoria principal dada por las siguientes direcciones: 0x0334500, 0x014BF00, 0x4134501, 0x124BF00, 0x03345FE, 0x34345AC, 0x214BFAA y 0x014BF54.

Suponiendo la cache originalmente vacía muestre la evolución de las etiquetas de la cache indicando los fallos y las transferencias entre cache de víctima y cache.

9. En un computador con caches separadas de datos e instrucciones, de 32KB cada una y bloques de 256 bytes, se quiere ejecutar el siguiente código C, que realiza la operación matricial C = A+B:

```
#define N 128
int A[N][N];          /* Un entero ocupa 32 bits
int B[N][N];
int C[N][N];

for (i=0; i < N; i++)
    for (j=0; j < N; j++)
        C[i][j] = A[i][j] + B[i][j];
```

Asumiendo que la matriz **A se ubica en la dirección 0x0C000000**, que B y C se almacenan consecutivamente a continuación de A, y que en la traducción a ensamblador del código anterior la variable *i* se ubica en un registro y que las direcciones de comienzo de los *arrays* están también en registro, se pide:

a) Calcular el número de fallos de cache en lectura (load) y en escritura (store) si la cache es de emplazamiento directo sin asignación en escritura.

b) Repetir el cálculo para una cache asociativa por conjuntos, de 2 vías, con política de reemplazamiento LRU, en los siguientes casos:
   - sin asignación en escritura.
   - con asignación en escritura.

En cada caso indicar si el rendimiento mejoraría o empeoraría y en qué proporción.

c) Un programador sugiere la siguiente modificación de código para una cache asociativa por conjuntos de 2 vías con asignación en escritura:

```
#define N 128
struct {
 int A;
 int B;
} AB[N][N];
int C[N][N];

for (i=0; i < N; i++)
 for (j=0; j < N; j++)
     C[i][j] = AB[i][j].A + AB[i][j].B;
```

Asumiendo que el array AB se ubica en la dirección 0x0C00_0000 y que el array C se ubica a continuación, calcule el número de fallos en lectura y escritura que se producirían, y compárelo con el resultado del apartado b) para la misma configuración de cache.

d) ¿Puede obtenerse un resultado similar al del apartado c) con el código original y sin aumentar el tamaño de la cache? Razone la respuesta.

10. Considere un computador con una memoria principal de 4MB, direccionable por bytes, al que se dota de una memoria cache de 4KB, con líneas (bloques) de 512B, y cache de víctima de 1024B. La memoria cache tiene dos vías mientras que la cache víctima es completamente asociativa. Para ambas el algoritmo de reemplazamiento es FIFO.
Originalmente en las dos memorias se encuentran los bloques que aparecen en las siguientes tablas:

Conjunto	Etiqueta	FIFO
0	668	0
0	008	1
1	297	0
1	368	1
2	5FF	0
2	668	1
3	297	1
3	200	0

Nº bloque de MP	FIFO
0A80	0
3FFF	1

Sea la secuencia de acceso a memoria principal dada por las siguientes direcciones: 0x334500, 0x14BF00, 0x150084, 0x004021, 0x0540AB y 0x0041F1.
Muestre la evolución de las etiquetas de cache indicando los fallos y las transferencias entre cache de víctima y cache.

11. Considere un computador con direcciones de 32 bits, al que se dota de una memoria cache de 128 bytes, con bloques de 16 bytes y emplazamiento directo, sin asignación en escritura (No write-allocate). Suponga que la variable *a* está en el banco de registros y no en cache.
Se quiere ejecutar el siguiente código:

```
int nota[128];    // nota[0] está en la dirección 0x00000000
int media[128];   // media [0] está a continuación de nota[127]

for (i=0;i<128;i++) {
      if (i>7 && i<64) {
       nota[i] = media[i]/2;
       }
      else {
       a = nota[i]*media[i]
       }
      }
```

a) Indique cuántos fallos de cache se producen.
b) Proponga dos optimizaciones de código (independientes) que mejoren el resultado, indicando cuántos fallos se producen en esos casos.
c) Si el tiempo de acceso a una palabra en cache es 1 ns, el tiempo de lectura o escritura de una palabra en memoria principal es 50 ns, el tiempo de transferencia de un bloque de MP a cache es 200 ns, ¿cuál es el tiempo necesario para acceder a todas las referencias del programa?
d) Si se añade al computador una memoria cache de segundo nivel de 1MB y asociatividad 4, con bloques de 16 bytes, con asignación en escritura y postescritura y si el tiempo de transferencia de un bloque desde memoria cache de segundo nivel a la de primer nivel son 15ns, ¿cuál es el tiempo necesario para acceder a todas las referencias del programa?

12. Un microcontrolador basado en ARM dispone de una memoria principal de 1 MB con cachés separadas de datos e instrucciones de 512 B cada una, de emplazamiento directo y bloques de 64 B. La caché de datos tiene asignación en escritura (write-allocate). En dicho sistema se quiere evaluar el rendimiento del siguiente fragmento de un programa:

```
for (i=0; i < 510; i++) {
  for (j=1; j < 7; j++) {
    A[i,j] = (A[i,j-1]+ A[i,j+1])/4;
  }
}
```

siendo A una matriz de 512x8 de enteros (un entero son 4 bytes) ubicada en memoria a partir de la dirección 0x203C0 y que se almacena por filas. Responda razonadamente a las siguientes cuestiones:

a) Indíquese el bloque de MP y el marco de bloque asociado a A[0][0].
b) Indíquese el número de bloque de MP que contiene el elemento A[127][3] y cuál es su marco de cache asociado.
c) Si todos los bits V de la memoria tienen el valor 0, indíquese la tasa de fallo obtenida al ejecutar el programa anterior.

13. Considere un computador con una memoria principal de 1MB, direccionable por bytes, al que se dota de una memoria cache de 1KB, con líneas (bloques) de 256B, cache de víctima de 512B y buffer de prebúsqueda de 256B. La memoria cache es asociativa por conjuntos con dos vías mientras que la cache víctima es completamente asociativa. En todas ellas el algoritmo de reemplazamiento es LRU (el valor 0 indica el bloque que hay que reemplazar).

a) Originalmente en las dos memorias se encuentran los bloques que aparecen en las siguientes tablas y en el buffer de prebúsqueda se encuentra el bloque de memoria principal número 0x005. Todos los bits de validez están a 1.

Cache			
Conjunto	**Bloque**	**Etiqueta**	**LRU**
0	0	0x0AA	1
0	1	0x700	0
1	0	0x6B0	0
1	1	0x700	1

Cache víctima	
Nº bloque de MP	**LRU**
0x700	1
0x0BB	0

Indicar qué ocurre en estas estructuras cuando de modo consecutivo se referencian las siguientes dos direcciones: 0xE0000, 0x08000, mostrando la evolución de las etiquetas de cache, los números de bloque de memoria principal contenidos en la cache víctima y en el buffer de prebúsqueda.
Indicar claramente, especificando origen y destino, qué transferencias de bloques ocurrieron al hacer la referencia 0x08000

b) Si en las dos memorias se encuentran los bloques que aparecen en las siguientes tablas y en el buffer de prebúsqueda se encuentra el bloque de memoria principal número 0x3FF. Todos los bits de validez están a 1.

Cache			
Conjunto	**Bloque**	**Etiqueta**	**LRU**
0	0	0x03F	0
0	1	0x700	1
1	0	0x000	0
1	1	0x70D	1

Cache víctima	
Nº bloque de MP	**LRU**
0x700	0
0x03F	1

Indicar qué ocurre en estas estructuras cuando de modo consecutivo se referencian las siguientes dos direcciones: 0x700AA, 0x3FFFF, mostrando la evolución de las etiquetas de cache, los números de bloque de memoria principal contenidos en la cache víctima y en el buffer de prebúsqueda. Así como las transferencias de bloques que se producen.