1. **Memoria virtual.** Se dispone de una computadora de 32 bits, con páginas de 1KB y una tabla de páginas lineal y soporte para memoria compartida. En dicha computadora se ejecuta el siguiente programa

```
  uint8_t A[1048576]; /* 1MB array */
  uint8_t B[1048576]; /* 1MB array */
  uint8_t C[1048576]; /* 1MB array */
  uint8_t D[1048576]; /* 1MB array */

for(i = 0;i < 1048576; i ++){
    D[i] = A[i] + B[i] + C[i]
}
```

   Los arreglos `A`, `B` y `C` son escritos por otro proceso.

   a) Realizar un diagrama de la tabla de páginas, indicando el valor de los bits `N`, `V`, `G`, `W` para cada entrada.

   b) Si se utiliza un cache VI/VT, explicar dos alternativas para asegurarse de que un proceso no pueda acceder a los datos de otro sin permiso.

   c) Al ejecutar el programa, se detecta que los cambios realizados por un proceso en los arreglos compartidos no son detectados por el otro. Indicar una posible causa del problema, y una solución.

2. **Recursividad y desempeño.** Los siguientes programas son dos versiones de la función `factorial`, una recursiva y una iterativa.

```
unsigned int fact_rec(unsigned int n)
{
        if (n<2)
                return 1;
        else
        {
                return n*fact_rec(n-1);
        }
}
```

```
unsigned int fact_iter(unsigned int n)
{
        unsigned int acum = 1;
        unsigned int i;
        for (i=1;i<=n;i++)
                acum*=i;
        return acum;
}
```

   Estos programas se compilan y ejecutan en una computadora MIPS32 con una memoria cache split de mapeo directo de 16KB, WB/WA, política de reemplazo LRU y tamaño de bloque de 32 bytes. A su vez la memoria tiene una latencia de 100ns y un ancho de banda de 100MHz. El tiempo de ciclo de la CPU es 1ns. Sabemos que cada iteración de `fact_iter` son 14 instrucciones y ocupa 14 más para crear y destruir el stack, y para `fact_rec` son 18 para el caso base y 25 para el general.

   a) Dar un diagrama de stack de cada función.

   b) Si asumimos que el caché está vacío antes de empezar a ejecutar el programa, ¿cuántos misses tendrá cada caché para cada versión en función de `n`? Justificar. [1]

   c) Indicar el speedup entre `fact_iter` y `fact_rec` si tomamos en cuenta sólo la cantidad de misses en el cache. ¿Es realista hacer eso?

3. **Desempeño.** La técnica conocida como *branch delay slot* nos permite evitar la pérdida de un ciclo de reloj en instrucciones de salto condicional, haciendo que pase de tomar 3 ciclos a 2 ciclos. Si un programa que utiliza la técnica pasa el 10 % de su tiempo de ejecución en saltos condicionales, ¿cuánto pasaría en saltos condicionales si no la usara?

[1] Asumir que originalmente el stack está alineado a 32 bytes.