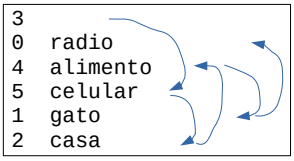
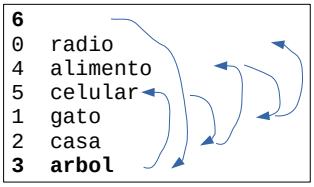
CC3301 Programación de Software de Sistemas – Control 2
Semestre Primavera 2025 – Profs. Mateu/Ibarra/Urrea

Pregunta 1 (40%)



El cuadro de la derecha muestra un ejemplo del contenido del archivo binario *lista.bin* que almacena una lista enlazada con nodos *celular*, *casa*, *alimento*, *gato* y *radio, en ese orden*. El archivo está compuesto por filas consecutivas, todas de 20 bytes enumeradas como 0, 1, 2, 3, 4 y 5. La fila 0 contiene *3*, la fila 1 contiene *0 radio*, la fila 2 contiene *4 alimento*, etc. Cada fila representa un nodo de la lista enlazada. Los primeros 4 bytes de una fila contienen en formato binario *little endian* un entero con el número de la fila que corresponde al próximo nodo en la lista enlazada (0 para el final de la lista), luego viene el texto almacenado en ese nodo en 16 caracteres, terminado con el carácter '\0' si ocupa menos de 16 caracteres. No hay caracter *newline* ('\n'). La fila 0 es especial porque solo contiene el número de la fila que contiene el primer nodo de la lista enlazada (celular en el ejemplo), **pero también ocupa 20 bytes**.

Programe la función *main* para el comando *./agregar.bin* que recibe como parámetros el nombre del archivo y un texto de hasta 16 caracteres. Este comando debe agregar el texto como primer nodo de la lista. Para ello, debe agregarlo como última fila en el archivo. Por ejemplo después de ejecutar el comando de abajo el archivo *lista.bin* queda como lo muestra el cuadro de la derecha.




```
$ ./agregar.bin lista.bin arbol
```

Observe que el texto *arbol* queda en la fila 6. La fila 0 cambia de 3 a 6, porque ahora el primer nodo de la lista está en la fila 6. Por simplicidad no necesita validar errores como por ejemplo que el archivo no existe.

Restricciones: No puede leer todo el archivo. Para modificar el archivo Ud. solo puede usar las funciones *fopen*, *fread*, *fwrite*, *ftell*, *fseek* y fclose.

Pregunta 2 (30%)

La función *f* de la derecha está programada en assembler Risc-V. Considere que se invoca *f* recibiendo en *a0* la dirección *d* de un arreglo de 6 enteros con los valores 3 1 1 5 5 0. La tabla de abajo muestra la ejecución parcial de la función *f* hasta el primer *addi*.

```
f:
        mv   t0,a0
L1:
        lw   t1,0(a0)
        sw   t1,0(t0)
        beq  t1,zero,L3
L2:
        addi a0,a0,4
        lw   t2,0(a0)
        beq  t1,t2,L2
        addi t0,t0,4
        j    L1
L3:
        mv a0,t0
        ret
```

Instrucción	*a0*	*t0*	*t1*	*t2*	*arreglo d*
	d				3 1 1 5 5 0
`mv t0,a0`		*d*			
`lw t1,0(a0)`			3		
`sw t1,0(t0)`					**3** 1 1 5 5 0
~~`beq t1,zero,L3`~~					
`addi a0,a0,4`	*d+4*				

Prosiga llenando la tabla con la ejecución de *lw t2,0(a0)* hasta que se ejecute la instrucción *ret*. No incluya lo que ya muestra este enunciado. Cada vez que se modifique *a0*, *t0*, *t1*, *t2* o el arreglo *d*, debe indicarlo en la columna correspondiente. En el caso de los saltos condicionales, tacharemos la instrucción para marcar cuando este no ocurre.

Pregunta 3 (30%)

Traduzca la función *g* del cuadro de la derecha a assembler Risc-V.

```
int g(int b, int e) {
  int r = 1;
  while (e != 0) {
    if ((e & 1) == 1)
      r = r * b;
    e = e >> 1;
    b = b * b;
  }
  return r;
}
```