**Respuesta a los ejercicios:**

* **Solución al Problema 5.4**

1 /\* Asegurarse de que "dest" se actualice en cada iteración \*/

2 void combine3w(vec\_ptr v, data\_t \*dest)

3 {

4 long i; /\* Esta línea declara una variable llamada i de tipo long que se usará como índice del bucle.\*/

5 long length = vec\_length(v);

6 data\_t \*data = get\_vec\_start(v);

7 data\_t acc = IDENT;

8

9 /\* Inicializar si la longitud es <= 0 \*/

10 \*dest = acc;

11

12 for (i = 0; i < length; i++) {

13 acc = acc OP data[i]; /\*Esta línea actualiza el valor de acc en cada iteración del bucle sumando el elemento data[i]\*/

14 \*dest = acc;

15 }

16 }

La función `combine3w` toma un vector y un puntero a un valor `dest` como argumentos, y computa una operación de reducción en el vector, actualizando el valor de `dest` en cada iteración del bucle. La operación de reducción es definida por una operación binaria `OP` y un elemento de identidad `IDENT`. En cada iteración del bucle, la función combina el elemento actual del vector con un acumulador `acc` usando `OP`, actualiza el valor de `acc` con el resultado y actualiza el valor de `dest` con el valor actual de `acc`. Al final de la función, el valor de `dest` contiene el resultado de la operación de reducción en el vector.

* **Solución al Problema 5.5**

La función realiza 2n multiplicaciones y n adiciones.

Podemos ver que el cálculo que limita el rendimiento aquí es el cálculo repetido de la expresión "xpwr = x \* xpwr". Esto requiere una multiplicación en punto flotante (5 ciclos de reloj), y el cálculo para una iteración no puede comenzar hasta que el de la iteración anterior haya finalizado. La actualización del resultado solo requiere una suma en punto flotante (3 ciclos de reloj) entre iteraciones sucesivas.

* **Solución al Problema 7.1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Símbolo | ¿Entrada en .symtab? | Tipo de símbolo | Módulo donde se define | Sección |
| buf | sí | externo | m.o | .data |
| bufp0 | sí | global | swap.o | .data |
| bufp1 | sí | global | swap.o | COMMON |
| swap | sí | global | swap.o | .text |
| temp | no | - | - | - |

* **Solución al Problema 7.2**

El enlazador elige el símbolo fuerte definido en el módulo 1 sobre el símbolo débil definido en el módulo 2 (regla 2):

(a) REF(main.1)→DEF(main.1)

(b) REF(main.2)→DEF(main.1)

Esto es un error, porque cada módulo define un símbolo fuerte "main" (regla 1).

El enlazador elige el símbolo fuerte definido en el módulo 2 sobre el símbolo débil definido en el módulo 1 (regla 2):

(a) REF(x.1)→DEF(x.2)

(b) REF(x.2)→DEF(x.2)

* **Solución al Problema 7.3**

linux> gcc p.o libx.a

linux> gcc p.o libx.a liby.a

linux> gcc p.o libx.a liby.a libx.a

Estos comandos de línea de comandos son para compilar un programa ejecutable en Linux usando el compilador de C gcc. Los comandos toman un archivo objeto y uno o más archivos de biblioteca como entrada, y los enlazan juntos en un programa ejecutable. El primer comando enlaza el archivo objeto p.o y la biblioteca libx.a juntos en un programa ejecutable. El segundo comando hace lo mismo que el primero, pero también incluye la biblioteca liby.a en el enlace. El tercer comando hace lo mismo que el segundo, pero incluye la biblioteca libx.a dos veces en el enlace. Esto puede ser útil en situaciones donde la biblioteca libx.a tiene dependencias circulares y requiere ser incluida dos veces para resolverlas.