Las acciones de la empresa ACME se transan en una bolsa de comercio cuyos operadores son representados mediante threads. Para comprar o vender una acción los operadores usan las funciones *compro* y *vendo*. Sus encabezados son:

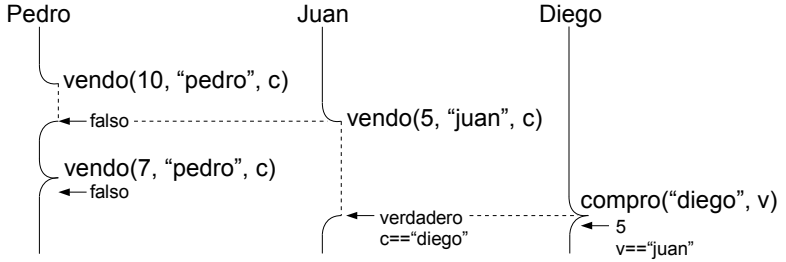
```
int vendo(int precio, char *vendedor, char *comprador);
int compro(char *comprador, char *vendedor);
```

En el cuadro siguiente se muestra a la izquierda el código usado por los múltiples threads vendedores de acciones y a la derecha el código de los múltiples compradores.

```
char *nom= miNombre();
int precio= miPrecio();
char comprador[100];
if (vendo(precio, nom,
          comprador)) {
  printf("vendi a %s\n",
         comprador);
}
```

```
char *nom= miNombre();
char vendedor[100];
sleep(tiempoAleatorio());
int precio= compro(nom, vendedor);
if (precio>0) {
  printf("compre a %s en %d\n",
         vendedor, precio);
}
```

La función *compro* transa una sola acción con el vendedor más barato de ese momento, retornando el precio pagado y copia el nombre del vendedor en el *2do parámetro*. Si en ese momento no hay ningún vendedor el precio será 0 y *compro* retorna de inmediato. La función *vendo* ofrece una acción al precio indicado y espera hasta que (i) aparezca un comprador, en cuyo caso retorna verdadero y copia el nombre del comprador en el 3er parámetro, o (ii) aparezca (o ya hay) un vendedor con un precio menor, retornando falso en tal caso. El siguiente diagrama muestra un ejemplo de ejecución.



Pedro llama a *vendo*, que retorna falso cuando Juan llama a *vendo* con un precio menor. La segunda llamada de Pedro fracasa de inmediato porque su precio todavía es mayor al de Juan. Juan sí tiene éxito cuando Diego llama a *compro* y por lo tanto la llamada a *vendo* de Juan retorna verdadero, copiando el nombre "diego" en el parámetro *comprador*. Por su parte *compro* retorna 5 (el precio) y copia "juan" en el parámetro *vendedor*.

Programe las funciones *vendo* y *compro*. Para la sincronización debe usar un monitor (es decir, un mutex y una condición). Observe que nunca habrá más de un vendedor en espera. Use variables globales. No olvide que en este curso Ud. no debe usar *busy-waiting*.

Recursos

Baje *t3.zip* de material docente en U-cursos y descomprímalo. El directorio *T3* contiene los archivos *test-bolsa.c* que prueba si su tarea funciona, *Makefile* que le servirá para compilar su tarea y *bolsa.h* que contiene los encabezados de las funciones pedidas. Ud. debe programar *vendo* y *compro* en el archivo *bolsa.c*. El programa de prueba lo felicitará si su tarea aprueba todos los tests o le indicará cuál test falla.

Lea los comentarios del inicio de *test-bolsa.c*. Le ayudarán a depurar su tarea.

Evaluación

Entregue su tarea solo si compila sin arrojar warnings en la máquina *anakena.dcc.uchile.cl*. Para optar a un 4.0 su tarea debe pasar exitosamente el primer test de *test-bolsa.c* en *anakena*. Esta máquina tiene un número restringido de threads y por eso es probable que no pase el segundo test. Para optar a un 7.0 su tarea debe pasar exitosamente ambos tests en su propia máquina. Si no pasa el segundo test, lo más probable es que se deba a un *datarace*.

Entrega

Ud. debe entregar el archivo *bolsa.c* por medio de U-cursos. Se descontará medio punto por día de atraso. No se consideran los días sábado, domingo, festivos o vacaciones.