```
#include "filosofo.h"
#include <mqueue.h>
#include <signal.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

#define MAX_TIME_PENSAR 7
#define MAX_TIME_COMER 5

void filosofo(char *filosofo, char *buzon_mesa, char *buzon_palillo_izq, char
*buzon_palillo_der);
void controlador(int senhal);

int main(int argc, char *argv[]) {
   filosofo(argv[1], argv[2], argv[3], argv[4]);
   return 0;
}
```

Este fragmento de código incluye las cabeceras de los archivos de encabezado necesarios para el resto del programa y define dos constantes, MAX_TIME_PENSAR y MAX_TIME_COMER, que se utilizan más adelante en el programa. También se define una función filosofo y otra controlador, y el programa principal llama a la función filosofo con cuatro argumentos pasados desde la línea de comandos.

```
void filosofo(char *filosofo, char *buzon_mesa, char *buzon_palillo_izq, char
*buzon_palillo_der) {
   mqd_t qHandlerMesa, qHandlerIzq, qHandlerDer;
   int n_filosofo;
   char buffer[64];

   // Retrollamada de finalización.
   if (signal(SIGINT, controlador) == SIG_ERR) {
      fprintf(stderr, "Abrupt termination.\n");
      exit(EXIT_FAILURE);
   }
   n_filosofo = atoi(filosofo);

   // Recupera buzones.
   qHandlerMesa = mq_open(buzon_mesa, O_RDWR);
   qHandlerIzq = mq_open(buzon_palillo_izq, O_RDWR);
   qHandlerDer = mq_open(buzon_palillo_der, O_RDWR);

   srand((int) getpid());
   while (1) {
      printf("[Filosofo %d] pensando...\n", n_filosofo);
      sleep(rand() % MAX_TIME_PENSAR); // Pensar.
```

```
            mq_receive(qHandlerMesa, buffer, sizeof(buffer), NULL);

            // Hambriento (coger palillos)...
            mq_receive(qHandlerIzq, buffer, sizeof(buffer), NULL);
            mq_receive(qHandlerDer, buffer, sizeof(buffer), NULL);
            // Comiendo...
            printf("[Filosofo %d] comiendo...\n", n_filosofo);
            sleep(rand() % MAX_TIME_COMER); // Comer.
            // Dejar palillos...
            mq_send(qHandlerIzq, buffer, sizeof(buffer), 0);
            mq_send(qHandlerDer, buffer, sizeof(buffer), 0);

            mq_send(qHandlerMesa, buffer, sizeof(buffer), 0);
        }
    }
```

Este fragmento de código define la función filosofo, que implementa el comportamiento de cada filósofo en el problema del banquete de los filósofos. La función recibe cuatro argumentos, que son los identificadores de los buzones de mensajes que utiliza cada filósofo para comunicarse con los otros procesos del sistema.

La función utiliza la función signal para establecer una devolución de llamada de finalización (controlador) en caso de que se reciba una señal SIGINT. A continuación, la función recupera los identificadores de los buzones de mensajes utilizando la función mq_open, y entra en un bucle infinito en el que el filósofo piensa durante un tiempo aleatorio, coge los palillos correspondientes a su izquierda y a su derecha, come durante un tiempo aleatorio y finalmente deja los palillos en la mesa.

El código del manager Aquí están las principales funciones:

- main(): Inicializa los recursos compartidos, crea los procesos filósofos y espera a que terminen. Al final, libera los recursos compartidos y termina el programa.
- controlador(): Función de control de señales. Si recibe la señal SIGINT (Control + C), llama a las funciones finalizarprocesos() y liberarecursos() para finalizar los procesos y liberar los recursos compartidos, respectivamente.
- liberarecursos(): Libera los recursos compartidos (los buzones).
- finalizarprocesos(): Finaliza los procesos filósofos.

En general, este código sigue la misma estructura que el código de los filósofos: inicializa los recursos compartidos, crea los procesos y espera a que terminen. Sin embargo, también se encarga de liberar los recursos compartidos y finalizar los procesos en caso de recibir una señal SIGINT.