```
// From James Peckol, Embedded Systems
// A simple OS kernel - step 2
#include <stdio.h>

// Prototypes for the tasks
void get (void* aNumber);               // input task
void increment (void* aNumber);         // computation task
void display (void* aNumber);    // display task

// Declare a TCB structure
typedef struct {
  void* taskDataPtr;
  void (*taskPtr)(void*);
}
TCB;

void main(void) {
  int i=0;                              // queue index
  int data;                             // declare a shared var, data
  int* aPtr = &data;                            // point to it
  TCB* queue[3];                        // declare queue as an array of pointers to TCBs

  // Declare some TCBs
  TCB inTask;
  TCB compTask;
  TCB outTask;
  TCB* aTCBPtr;

  // Initialize the TCBs
  inTask.taskDataPtr=(void*)&data;
  inTask.taskPtr=get;

  compTask.taskDataPtr=(void*)&data;
  compTask.taskPtr=increment;

  outTask.taskDataPtr=(void*)&data;
  outTask.taskPtr=display;

  // initialize the task queue
  queue[0]= &inTask;
  queue[1]= &compTask;
  queue[2]= &outTask;

  // schedule and dispatch the tasks
  while(1) {
    aTCBPtr=queue[i];
    aTCBPtr->taskPtr( (aTCBPtr->taskDataPtr) );
    i=(i+1)%3;
  }
  return;
}

void get(void* aNumber) {         // perform input operation
  printf("Enter a number, 0..9 ");
  *(int*) aNumber = getchar();
  getchar();                            // discard CR
  *(int*) aNumber -= '0';           // convert to decimal from ASCII
  return;
}

void increment(void* aNumber) {// perform computation
  int* aPtr = (int*) aNumber;
  (*aPtr)++;
  return;
}
```

```
void display (void* aNumber) {   // perform output operation
  printf("The result is: %d\n",*(int*)aNumber);
  return;
}
```