```java
import java.io.IOException;
/**
 * InputDataHandler uses a linked list queue to hold the characters
 * input from System.in.  The handling of the input stream is done in
 * a separate thread to prevent blocking the program consuming the characters.
 * The queue is maintained in a thread-safe manner.
 * Usuage:
 *  InputDataHandler handler = new InputDataHandler();
 *  Thread(handler).start();
 *
 * @uthor Mr. Page
 *
 */
public final class InputDataHandler implements Runnable
{
    private final LinkedList<Character> queue;

    /**
     * constructor for InputDataHandler objects
     * constructs the queue
     */
    public InputDataHandler()
    {
        queue = new LinkedList<Character>();
    }
    /**
     * get and return the number of elements in the queue
     * @return the size of the queue
     */
    public synchronized int size()
    {
        return queue.size();
    }

    /**
     * get a character from the queue and return as a String object. White space
     * in the queue is ignored.
     * @return a String representation of the first character in the queue
     */
    public String getInput()
    {
        if(queue.isEmpty())
        {
            //System.out.println("waiting on buffer");
            try
            {
                while(queue.isEmpty()) Thread.sleep(10);
            }
            catch (InterruptedException e)
            {
                e.printStackTrace();
                System.exit(1);
            }
        }
        synchronized(this)
        {
            return "" + queue.remove();
        }
    }

    public void run()
    {
        while(true)
        {
            try
            {
                while(System.in.available()<= 0) Thread.sleep(10);
                Character key = (char) System.in.read();
                synchronized(this)
                {
                    queue.add(key);
                }

            }
            catch (IOException e)
            {
                e.printStackTrace();
                System.exit(1);
            }
            catch (InterruptedException e)
            {
                e.printStackTrace();
                System.exit(1);
            }
        }

    }
}

```