

```

void enQ(int n) {
    x = new Item(n);
    if (head==NULL) head=x;
    else tail->next=x;
    tail=x;
}

```

```

int deQ() {
    x=head;
    yield();
    if (x==NULL) return -1;
    if (x==tail) tail=NULL;
    head=x->next;
    v=x->value;
    delete x;
    return v;
}

```

1. [60 pts] The low-level mechanism for communicating between a certain pair of computers uses the `send(char* message)` and `recv(char* message)` primitives. These primitives are both blocking, and always send and receive 256 byte messages. The OS uses them to provide a “connection-oriented” messaging service. This service supplies the following primitives to applications:

`int connect():` Returns an integer which designates a new “connection ID”.

`Write(connectionID, char* message, int nchars):`

Sends a message of length `nchars` bytes, which consists of the bytes `message[0]...message[nchars-1]` on connection `connectionID`. `nchars` cannot be larger than 200.

`int Read(connectionID, char* message)::`

Reads a message from connection `connectionID` into `message[0]..message[n-1]`, where the message consists of `n` bytes. Returns `n`.

These primitives block until completion.

Even if many connections are active, the messages using each connection are kept distinct, and arrive at the receiving end of the connection in the same order in which they were sent.

Sketch an implementation of the connection-oriented primitives, and the daemon which monitors the low-level communication mechanism. Allow `Write` to complete as soon as the OS gets the message to be sent (don't force it to wait until the message is actually transmitted). `Read` must block if there are no messages waiting on this connection.

You may use the List class defined above, and you should assume it has been implemented correctly. You may use additional synchronization operations as needed. You may assume that when both machines execute a “connect()” operation, the same connection ID is returned by both operations. You need not show the implementation of “connect”. [Hint: The OS can embed the application’s message in a longer one for actual transmission.]