

READ ME

姓名：周宁

班级：111171

实习思路：

1. 采取回退 N 步的策略，即 GBN 协议，在一开始我会先缓存几个包（根据窗口的大小 `self.cwnd` 决定），然后我会发送几个包，但不是所有的缓存的包都会发送（根据窗口大小 `self.cwnd` 和预留空间 `self.reservezone` 决定），并且在 `base==nextseqno` 时，开启一个定时器，用来接收
2. 发送时的正常情况：我接受到一个应答包后，我会立即发送 `self.nextseqno` 位置所在的包，然后又缓存一个，于此同时会再次开启定时器（原来的定时器已经结束了），直到最后的 end 包
3. 发送端丢包或者损坏：这时接收端都不会有应答包，所以我这边会有超时的响应，我就会调用方法 `handle_timeout`，重发 `base` 到 `nextseqno` 之间的包
4. 当出现乱序：
 - a. 当响应包重复 3 次之前，我发包的条件是 `self.nextseqno < (self.base + self.cwnd - self.reservezone)`，而 `base = int(ack)`，是不会改变，所以此时并不会发送包，简单来说就是什么都没有做
 - b. 当响应包重复 3 次时，我就认为这里出现了丢包，我就重发了 `base` 到 `nextseqno` 之间的包
5. 当超时或者收到的响应包出现问题时（比如响应包损坏或者丢了），我单方面会认为是出现了丢包现象，此时类似 3，会调用 `handle_timeout`

实习遇到的问题：

1. 开始不知道如何下手

解决方法：看书和 PPT，理解滑动窗口协议，然后在草稿纸上自己画了一下基本的情况。我自己理解滑动窗口就是一个循环队列，通过不断的滑动进行发包。

2. 发送过去了但是总是“停不下来”，就是发送 end 包会发送好几次

解决方法：加一个控制条件 `isEnd` 判断 end 包是否发过，如果 end 包发过后就再也不发了

3. 在做测试时，总是 fails，但是却显示了 ack 包，并且收到了

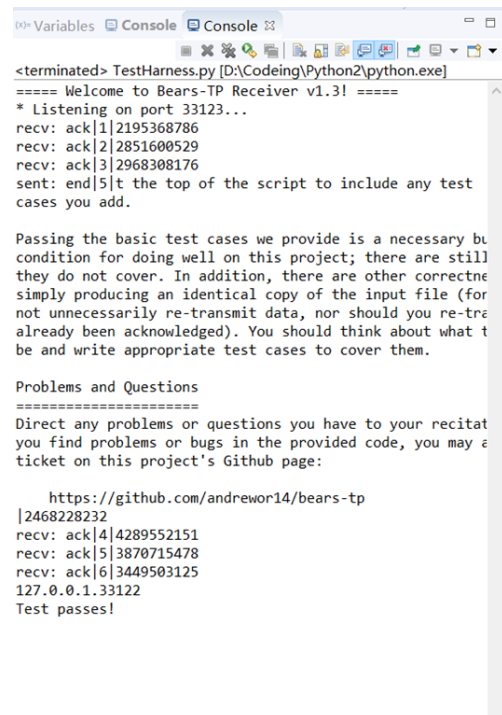
解决方法：查看了下生成的文件，发现是文件打开和保存的问题，打开保存都不是二进制导致了这个问题，所以我把发送端和接收端的打开、保存文件的方式改为了 2 进制，希望老师再测试时也会改

4. 原本 end 包是没有文件的，发现这个与协议不符合，并且多一个包的发送

解决方法：在 `msg=''` 时，我会将最后一个包给拆掉，重新改成 end 包，这个也算设计的一个失误，进行拆包可能会很慢，影响效率

结果展示:

正常测试: (测试: BasicTest, 文件: readme)



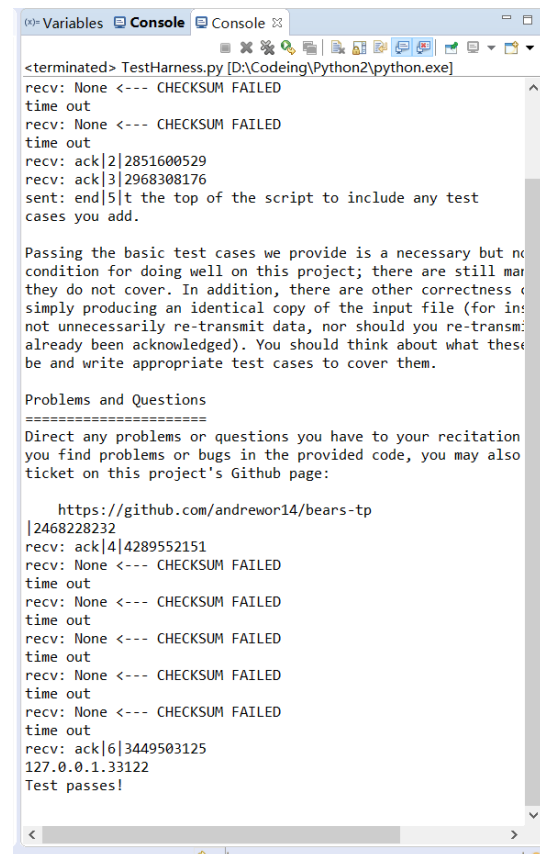
```
<terminated> TestHarness.py [D:\Coding\Python2\python.exe]
===== Welcome to Bears-TP Receiver v1.3! =====
* Listening on port 33123...
recv: ack|1|2195368786
recv: ack|2|2851600529
recv: ack|3|2968308176
sent: end|5|t the top of the script to include any test
cases you add.

Passing the basic test cases we provide is a necessary but
condition for doing well on this project; there are still
they do not cover. In addition, there are other correctness
simply producing an identical copy of the input file (for
not unnecessarily re-transmit data, nor should you re-trans
already been acknowledged). You should think about what t
be and write appropriate test cases to cover them.

Problems and Questions
=====
Direct any problems or questions you have to your recitat
you find problems or bugs in the provided code, you may a
ticket on this project's Github page:

    https://github.com/andrewor14/bears-tp
|2468228232
recv: ack|4|4289552151
recv: ack|5|3870715478
recv: ack|6|3449503125
127.0.0.1.33122
Test passes!
```

随机丢包: (测试: RandomDropTest, 文件: readme)



```
<terminated> TestHarness.py [D:\Coding\Python2\python.exe]
recv: None <--- CHECKSUM FAILED
time out
recv: None <--- CHECKSUM FAILED
time out
recv: ack|2|2851600529
recv: ack|3|2968308176
sent: end|5|t the top of the script to include any test
cases you add.

Passing the basic test cases we provide is a necessary but no
condition for doing well on this project; there are still man
they do not cover. In addition, there are other correctness o
simply producing an identical copy of the input file (for ins
not unnecessarily re-transmit data, nor should you re-transm
already been acknowledged). You should think about what these
be and write appropriate test cases to cover them.

Problems and Questions
=====
Direct any problems or questions you have to your recitation
you find problems or bugs in the provided code, you may also
ticket on this project's Github page:

    https://github.com/andrewor14/bears-tp
|2468228232
recv: ack|4|4289552151
recv: None <--- CHECKSUM FAILED
time out
recv: None <--- CHECKSUM FAILED
time out
recv: None <--- CHECKSUM FAILED
time out
recv: None <--- CHECKSUM FAILED
time out
recv: None <--- CHECKSUM FAILED
time out
recv: ack|6|3449503125
127.0.0.1.33122
Test passes!
```

将 2 号包丢三次的测试：（测试：Drop2Test，文件：readme）

```
<terminated> TestHarness.py [D:\Codeing\Python2\python.exe]
===== Welcome to Bears-TP Receiver v1.3! =====
* Listening on port 33123...
recv: ack|1|2195368786
recv: ack|2|2851600529
recv: ack|2|2851600529
recv: ack|2|2851600529
recv: None <--- CHECKSUM FAILED
time out
recv: ack|2|2851600529
is to self.handle_dup_ack
recv: ack|2|2851600529
recv: ack|2|2851600529
recv: ack|2|2851600529
is to self.handle_dup_ack
recv: ack|5|3870715478
sent: end|5|t the top of the script to include any test
cases you add.

Passing the basic test cases we provide is a necessary bu
condition for doing well on this project; there are still
they do not cover. In addition, there are other correctne
simply producing an identical copy of the input file (for
not unnecessarily re-transmit data, nor should you re-tra
already been acknowledged). You should think about what t
be and write appropriate test cases to cover them.

Problems and Questions
=====
Direct any problems or questions you have to your recitat
you find problems or bugs in the provided code, you may a
ticket on this project's Github page:

    https://github.com/andrewor14/bears-tp
|2468228232
recv: ack|5|3870715478
recv: ack|5|3870715478
recv: ack|6|3449503125
127.0.0.1.33122
Test passes!
```

乱序测试：（测试：unsoredTest，文件：readme）

```
<terminated> TestHarness.py [D:\Codeing\Python2\python.exe]
===== Welcome to Bears-TP Receiver v1.3! =====
* Listening on port 33123...
recv: ack|1|2195368786
recv: ack|3|2968308176
sent: end|5|t the top of the script to include any test
cases you add.

Passing the basic test cases we provide is a necessary bu
condition for doing well on this project; there are still
they do not cover. In addition, there are other correctne
simply producing an identical copy of the input file (for
not unnecessarily re-transmit data, nor should you re-tra
already been acknowledged). You should think about what t
be and write appropriate test cases to cover them.

Problems and Questions
=====
Direct any problems or questions you have to your recitat
you find problems or bugs in the provided code, you may a
ticket on this project's Github page:

    https://github.com/andrewor14/bears-tp
|2468228232
recv: ack|2|2851600529
recv: ack|4|4289552151
recv: ack|6|3449503125
127.0.0.1.33122
Test passes!
```

发了一张图片（测试：unsoredTest，文件：1，jpg，大小：870kb）

```

<terminated> TestHarness.py [D:\Coding\Python2\python.exe]
recv: ack|858|3292507578
recv: ack|859|3710164219
recv: ack|860|245406187
recv: ack|861|398174378
recv: ack|862|1016527721
recv: ack|863|630049320
recv: ack|864|1791778031
recv: ack|865|1943498158
recv: ack|866|1492794989
recv: ack|867|1105268524
recv: ack|868|3329823715
recv: ack|869|3747750562
recv: ack|870|258143196
recv: ack|871|377078429
recv: ack|872|1028952414
recv: ack|873|609198111
recv: ack|874|1796093656
recv: ack|875|1913980825
recv: ack|876|1496863834
recv: ack|877|1076061467
recv: ack|878|3350952404
recv: ack|879|3735046293
recv: ack|880|71218401
recv: ack|881|488998304
recv: ack|882|906548835
recv: ack|883|789833506
recv: ack|884|1616015845
recv: ack|885|2034843812
recv: ack|886|1382290279
recv: ack|887|1266623014
sent: end|889|n9000^00ay ?T000p00000d0000/d00V
gzl(000e)%d00000:(w000@v00|00ed0g0$0000rwsM200i>000Vv0Cc%
h00N00|000Rw[(000090bxb0D-)f0:0000Z30w0=00u }02~\0=0;a0
000,0S00J000000000|2097121578
recv: ack|888|3437706985
recv: ack|889|3590066088
recv: ack|890|100458198
127.0.0.1.33122
Test passes!

```

实习感想和改进:

本次实习让我们体会了一把可靠传输，我自己做的可靠传输是最简单的 GBN 的可靠传输，直接回退 N 步，并没有做部分选择的传输，如果时间充足我还是挺想试一试的，除此之外我发现我可以改进的地方还有许多，不如在发送端可以改写一下让发送端回的包告诉自己接收端的窗口大小以便自己这边对窗口大小做出改变等等，不过这次实习的收获步小，在学习操作系统之后我想去底层看一下，可以更改一下自己的可靠传输，体验一把。

