

















































@02721735

Prajjwal Dungal

Github link: <https://github.com/prajjwaldangal/SocketProgramming.git>

All commits:

Commits on Feb 2, 2017		
	added final touches prajjwaldangal committed 4 minutes ago	 e8a55bd 
	file size and return to client prajjwaldangal committed 3 hours ago	 3ac9d47 
	CAP parts fully working prajjwaldangal committed 3 hours ago	 6751097 
	added comments prajjwaldangal committed 6 hours ago	 1fce781 
	Added commandline arguments prajjwaldangal committed 6 hours ago	 9d993b7 
	added handleT prajjwaldangal committed 9 hours ago	 a375edf 
Commits on Feb 1, 2017		
	string formatting still prajjwaldangal committed 15 hours ago	 b1f0d82 
	managing string format prajjwaldangal committed 16 hours ago	 ec8b4eb 
	extracting CAP of FILE prajjwaldangal committed 22 hours ago	 fe83fe0 
	fixed handling s in client prajjwaldangal committed 22 hours ago	 d39e7c0 
	fgets not working prajjwaldangal committed 22 hours ago	 a154f19 
	connection established prajjwaldangal committed a day ago	 3dc17b3 
	setup readline and writeline in server prajjwaldangal committed a day ago	 83be754 
	setup readline function in server prajjwaldangal committed a day ago	 a12f649 
	Added files prajjwaldangal committed a day ago	 c08d5da 
	Initial commit prajjwaldangal committed a day ago	 c38eac2 

Added files

[Browse files](#) master

prajjwaldangal committed a day ago

1 parent c38eac2

commit c08d5da426a7276e849922ac927d447f850a052f

 Showing 2 changed files with 131 additions and 0 deletions.

Unified

Split

70 client.cc

[View](#)

```
... @@ -0,0 +1,70 @@
1  + #include <stdio.h>
2  +
3  + #include <sys/socket.h>
4  + #include <sys/types.h>
5  + #include <arpa/inet.h>
6  + #include <unistd.h>
7  + #include <netdb.h>
8  +
9  + // char * handleS()
10 + // {
11 + //     return "ap";
12 + // }
13 +
14 + // char * handleT()
15 + // {
16 + //
17 + //     return "ap";
18 + // }
19 +
20 + #define ECHO_PORT    (2002)
21 +
22 + int main ()
23 + {
24 +     char choice;
25 +     char *msg;
26 +
27 +     // printf("Enter s, t, or q (lowercase):\n");
28 +     // scanf("%s", choice);
29 +
30 +     // switch (choice)
31 +     // {
32 +     //     case 's':
33 +     //         msg = handleS();
34 +     //     case 't':
35 +     //         msg = handleT();
36 +     //     case 'q':
```

setup readline function in server

[Browse files](#)

master

prajwaldangal committed a day ago

1 parent c08d5da commit a12f649c32fc5d62540ac36c11cebfd5ad2e6e2d

Showing 1 changed file with 33 additions and 5 deletions.

Unified Split

38 socket.cc

View

```

@@ -45,17 +45,45 @@ int main()
45 45         break;
46 46     }
47 47
48 -     read(conn_s, buffer, MAX_LINE - 1);
49 -
48 +     //read(conn_s, buffer, MAX_LINE - 1);
49 +     Readline(conn_s, buffer, MAX_LINE-1);
50 50
51 51 +
52 52     close (conn_s);
53 53
54 54 }
55 -
56 -
57 55     // sa_family -> AF_INET
58 56     // sa_data -> port numbers
57 +
58 +}
59 +
60 +ssize_t Readline(int sockd, void *vptr, size_t maxlen) {
61 +    ssize_t n, rc;
62 +    char c, *buffer;
63 +
64 +    buffer = vptr;
59 65
66 +    for ( n = 1; n < maxlen; n++ ) {
60 67
61 -    }
68 +        if ( (rc = read(sockd, &c, 1)) == 1 ) {
69 +            *buffer++ = c;
70 +            if ( c == '\n' )
71 +                break;
72 +        }
73 +        else if ( rc == 0 ) {
74 +            if ( n == 1 )
75 +                return 0;
76 +            else
77 +                break;
78 +        }
79 +        else {
80 +            if ( errno == EINTR )
81 +                continue;

```

```

48 +     switch (choice)
49 +     {
50 +         case 's':
51 +             msg = handleS();
52 +             break;
53 +         // case 't':
54 +         //     msg = handleT();
55 +         case 'q':
56 +             return 0;
57 +             break;
58 +         default:
59 +             break;
60 +     }
61 +
62 +     printf("Message is: %c, %c \n", *msg, *(msg + 5));
70 63
71 64     char buffer[MAX_LINE-1];
72 65     struct sockaddr_in servaddr;
* @@ -95,8 +88,9 @@ int main ()
95 88         printf("Error connecting \n");
96 89     }
97 90
98 -     buffer[0] = 'a';
99 -     buffer[1] = 'b';
100 94
101 95     Writeline(conn_s, buffer, MAX_LINE-1);
102 96     Readline(conn_s, buffer, MAX_LINE-1);

```

setup readline and writeline in server

[Browse files](#)

master

prajjwaldangal committed a day ago

1 parent [a12f649](#)commit [83be754d35056ae49ae59d117e50bcf30baa2c17](#)

Showing 4 changed files with 47 additions and 6 deletions.

Unified

Split

BIN a.out

[View](#)

Binary file not shown.

18 client.cc

[View](#)

```
...    ...    @@ -1,11 +1,13 @@
1      1      #include <stdio.h>
2      2      +#include <stdlib.h>
3      3
4      4      #include <sys/socket.h>
5      5      #include <sys/types.h>
6      6      #include <arpa/inet.h>
7      7      #include <unistd.h>
8      8      #include <netdb.h>
9      9
10     10     +#include <string.h>
11     11     // char * handleS()
12     12     // {
13     13     //     return "ap";
14
15     ✱     @@ -17,7 +19,9 @@
17     17     //     return "ap";
18     18     // }
19     19
20     20     -#define ECHO_PORT      (2002)
21     21     +
22     22     +#define ECHO_PORT      (2002)
23     23     +#define MAX_LINE      (1000)
24     24     +
25
26     26     int main ()
27     27     {
28
29     ✱     @@ -39,11 +43,15 @@ int main ()
39     39     //         break;
40     40     // }
41     41
42     42     + char buffer[MAX_LINE-1];
43     43     struct sockaddr_in servaddr;
44     44     + short int port;          /* port number          */
45     45     + char    *szAddress;      /* Holds remote IP address */
46     46     + char    *szPort;         /* Holds remote port      */
47
48     48     // struct hostent *h;
49
50     50     - // memset(&servaddr, 0, sizeof(servaddr));
51     51     + memset(&servaddr, 0, sizeof(servaddr));
```

```

88 +ssize_t Readline(int sockd, char *vptr, size_t maxlen) {
89 +     ssize_t n, rc;
90 +     char c, *buffer;
91 +
92 +     buffer = vptr;
93 +
94 +     for ( n = 1; n < maxlen; n++ ) {
95 +
96 +         if ( (rc = read(sockd, &c, 1)) == 1 ) {
97 +             *buffer++ = c;
98 +             if ( c == '\n' )
99 +                 break;
100 +         }
101 +         else if ( rc == 0 ) {
102 +             if ( n == 1 )
103 +                 return 0;
104 +             else
105 +                 break;
106 +         }
107 +         else {
108 +             if ( errno == EINTR )
109 +                 continue;
110 +             return -1;
111 +         }
112 +     }
113 +
114 +     *buffer = 0;
115 +     return n;
116 +}
117 +
118 +ssize_t Writeline(int sockd, char *vptr, size_t n) {
119 +     size_t nleft;
120 +     ssize_t nwritten;
121 +     char *buffer;
122 +
123 +     buffer = vptr;
124 +     nleft = n;
125 +
126 +     while ( nleft > 0 ) {
127 +         if ( (nwritten = write(sockd, buffer, nleft)) <= 0 ) {
128 +             if ( errno == EINTR )
129 +                 nwritten = 0;
130 +             else
131 +                 return -1;
132 +         }
133 +         nleft -= nwritten;
134 +         buffer += nwritten;
135 +     }
136 +
137 +     return n;
138 +}

```

58	50	char * file_str = "FILE";
59	-	
60	51	char * ret_str = (char *) malloc(sizeof(char) * MAX_LINE);
61	-	char semi_buf[MAX_LINE-5];
52	+	// char ret_str[MAX_LINE-1];
53	+	char * semi_buf = (char *) malloc(sizeof(char) * MAX_LINE-5);
54	+	// char semi_buf[MAX_LINE-1];
62	55	char conv[4]; // convert cap_count to char
63	56	
57	+	printf("Received buffer: %s\n", buffer);
64	58	
65	59	FILE * fp;
66	-	char file_path[MAX_LINE-6];
60	+	char file_path[MAX_LINE-7];
67	61	long bytes;
68	-	char n_bytes[MAX_LINE-2]; // this means we will have limitation as to the size of file
69	-	char file_buf[MAX_LINE];
62	+	char n_bytes[MAX_LINE-3]; // this means we will have limitation as to the size of file
63	+	char file_buf[MAX_LINE-1];
70	64	
71	65	int n_cap = 0, n_file = 0;
72	66	
✱	@@ -83,28 +77,33 @@	int main(int argc, char *argv[])
83	77	
84	78	}
85	79	printf("n_cap: %d, n_file: %d\n", n_cap, n_file);
86	-	int cap_count = 0;
80	+	int cap_count;
87	81	if (n_cap > n_file)
88	82	{
89	83	printf("if block\n");
90	-	int i = 3;
84	+	int i = 4;
91	85	int n_c = 1;
92	-	while (n_c < 3)
86	+	while (n_c < 2)
93	87	{
94	88	if (buffer[i] == '\n')
95	89	{
96	90	n_c++;
97	91	} else {
98	-	semi_buf[i-3] = toupper(buffer[i]);
92	+	* (semi_buf+ i-4) = toupper(buffer[i]);
99	93	cap_count++;
100	94	}
101	95	i++;
102	96	}
103	-	sprintf(conv, "%d", cap_count); /* converting int to string */
104	-	strcat(ret_str, conv);
97	+	printf("str len %lu\n", strlen(buffer));
98	+	// for (int i=0; i < str)
99	+	
100	+	sprintf(ret_str, "%i", cap_count); /* converting int to string */

fgets not working

[Browse files](#)

master

prajwaldangal committed 22 hours ago

1 parent 3dc17b3

commit a154f19f564583f950c1c3a237dc2076c59055f1

Showing 4 changed files with 42 additions and 13 deletions.

[Unified](#) [Split](#)

BIN a.out

[View](#)

Binary file not shown.

53 client.cc

[View](#)

```
@@ -9,9 +9,26 @@
9      9
10     10      #include <string.h>
11     11      #include <errno.h>
12     12      +
13     13      +#include "client_helper.c"
14     14      +
15     15      +#define ECHO_PORT      (2002)
16     16      +#define MAX_LINE        (1000)
17     17      +
18     18      +ssize_t Readline(int sockd, char *vptr, size_t maxlen);
19     19      +ssize_t Writeline(int sockd, char *vptr, size_t n);
20     20      +
12    21      // char * handleS()
13    22      // {
14    23      -//      return "ap";
23    24      +//      // char * msg = (char *) malloc(sizeof(char) * MAX_LINE);
24    25      +
25    26      +//      char msg[MAX_LINE];
26    27      +
27    28      +//      fgets(msg, MAX_LINE, stdin);
28    29      +//      char * ptr;
29    30      +//      ptr = msg;
30    31      +
31    32      +//      return ptr;
15    32      // }
16    33
17    34      // char * handleT()
```

12 client_helper.c

[View](#)

```
@@ -25,9 +25,17 @@ char * handleS()
25    25      char * handleT()
26    26      {
27    27          char * file_str = "FILE";
28    28      +      char * msg_comp = (char *) malloc(sizeof(char) * MAX_LINE);
29    29      +      strcat(msg_comp, file_str);
30    30      +      strcat(msg_comp, &newlineChar);
28    31
32    32      +      char * msg = (char *) malloc(sizeof(char) * MAX_LINE-6);
33    33      +      printf("Enter the message: \n");
34    34      +      fgets(msg, MAX_LINE, stdin);
35    35      +      scanf("%s", msg);
29    36
30    37      -          // printf("client helper, msg_comp: %c, ptr: %c\n", msg_comp[0], *ptr);
37    38      +      strcat(msg_comp, msg);
38    39      +      strcat(msg_comp, &newlineChar);
31    39
32    40      -      return "ap";
40    41      +      return msg_comp;
33    41      }
```


✱	@@ -52,7 +52,7 @@	int main(int argc, char *argv[])
52	52	// char ret_str[MAX_LINE-1];
53	53	char * semi_buf = (char *) malloc(sizeof(char) * MAX_LINE-5);
54	54	// char semi_buf[MAX_LINE-1];
55	-	char conv[4]; // convert cap_count to char
55	+	//char conv[4]; // convert cap_count to char
56	56	
57	57	printf("Received buffer: %s\n", buffer);
58	58	
✱	@@ -107,7 +107,7 @@	int main(int argc, char *argv[])
107	107	for (int i=5; i < strlen(buffer)-1; i++) { // MAX_LINE - 1 to escape reading the last \n
108	108	file_path[i-5] = buffer[i];
109	109	}
110	-	printf("file path: %s", file_path);
110	+	printf("file path: %s\n", file_path);
111	111	fp = fopen(file_path, "r");
112	112	if (fp == NULL) {
113	113	strcat(ret_str, "9");
✱	@@ -122,22 +122,29 @@	int main(int argc, char *argv[])
122	122	sprintf(n_bytes, "%ld", bytes);
123	123	strcat(ret_str, n_bytes);
124	124	strcat(ret_str, "\n");
125	-	int c;
126	-	char conv2;
125	+	fclose(fp);
126	+	char * conv2 = malloc(sizeof(char));
127	+	fp = fopen(file_path, "r");
128	+	int c = fgetc(fp);
129	+	sprintf(conv2, "%i", c);
130	+	strcat(ret_str, conv2);
131	+	Writeline(conn_s, ret_str, MAX_LINE-1);
132	+	printf("c: %c\n", c);
133	+	
127	134	c = fgetc(fp);

```

54 56      Readline(conn_s, buffer, MAX_LINE-1);
55 57 +      printf("MSG at server line 57: %c %c %c\n", *(buffer), *(buffer+1), *(buffer+8));
56 58
57 59 -      char C_or_F[3];
58 60 +      // CAPthisisthefuture
59 61 +      // corf is temporary, cf contains whether CAP or FILE
60 62 +      char C_or_F[3], cf[3];
61 63
62 64 -      C_or_F[0] = *buffer[0];
63 65 -      C_or_F[1] = *buffer[1];
64 66 -      C_or_F[2] = *buffer[2];
65 67 +      C_or_F[0] = *buffer;
66 68 +      C_or_F[1] = *(buffer+1);
67 69 +      C_or_F[2] = *(buffer+2);
68 70
69 71 -      if
70 72 +      // client -> server: "CAP\nxxx\n"
71 73
72 74 -      buffer[0] = 'j';
73 75 -      buffer[1] = 'k';
74 76 +      // server -> client: "FILE\nxxx\n"
75 77
76 78
77 79 +      strcat(cf, C_or_F);
78 80 +
79 81 +      printf("isequal %d, string: %d %d %d %c %c %c\n", (strcmp(cf, "CAP")), (strcmp(C_or_F, "CAP")), (cf == "
80 82 +
81 83 +      int cap_count;
82 84 +      if ((strcmp(C_or_F, "CAP") < 1))
83 85 +      {
84 86 +          int i = 3;
85 87 +          int n_c = 1;
86 88 +          while (n_c < 2)
87 89 +          {
88 90 +              if (*(buffer+i) == '\n')
89 91 +              {
90 92 +                  n_c++;
91 93 +              } else {
92 94 +                  *(buffer+i) = toupper(*(buffer+i));
93 95 +                  cap_count++;
94 96 +              }
95 97 +              i++;
96 98 +              printf("Cap count, server 77: %d, isequal: %d", cap_count, strcmp(cf, "CAP"));
97 99 +          }
98 100 +      }
99 101
100 102
101 103
102 104
103 105
104 106
105 107
106 108
107 109
108 110
109 111
110 112
111 113
112 114
113 115
114 116
115 117
116 118
117 119
118 120
119 121
120 122
121 123
122 124
123 125
124 126
125 127
126 128
127 129
128 130
129 131
130 132
131 133
132 134
133 135
134 136
135 137
136 138
137 139
138 140
139 141
140 142
141 143
142 144
143 145
144 146
145 147
146 148
147 149
148 150
149 151
150 152
151 153
152 154
153 155
154 156
155 157
156 158
157 159
158 160
159 161
160 162
161 163
162 164
163 165
164 166
165 167
166 168
167 169
168 170
169 171
170 172
171 173
172 174
173 175
174 176
175 177
176 178
177 179
178 180
179 181
180 182
181 183
182 184
183 185
184 186
185 187
186 188
187 189
188 190
189 191
190 192
191 193
192 194
193 195
194 196
195 197
196 198
197 199
198 200
199 201
200 202
201 203
202 204
203 205
204 206
205 207
206 208
207 209
208 210
209 211
210 212
211 213
212 214
213 215
214 216
215 217
216 218
217 219
218 220
219 221
220 222
221 223
222 224
223 225
224 226
225 227
226 228
227 229
228 230
229 231
230 232
231 233
232 234
233 235
234 236
235 237
236 238
237 239
238 240
239 241
240 242
241 243
242 244
243 245
244 246
245 247
246 248
247 249
248 250
249 251
250 252
251 253
252 254
253 255
254 256
255 257
256 258
257 259
258 260
259 261
260 262
261 263
262 264
263 265
264 266
265 267
266 268
267 269
268 270
269 271
270 272
271 273
272 274
273 275
274 276
275 277
276 278
277 279
278 280
279 281
280 282
281 283
282 284
283 285
284 286
285 287
286 288
287 289
288 290
289 291
290 292
291 293
292 294
293 295
294 296
295 297
296 298
297 299
298 300
300 301
301 302
302 303
303 304
304 305
305 306
306 307
307 308
308 309
309 310
310 311
311 312
312 313
313 314
314 315
315 316
316 317
317 318
318 319
319 320
320 321
321 322
322 323
323 324
324 325
325 326
326 327
327 328
328 329
329 330
330 331
331 332
332 333
333 334
334 335
335 336
336 337
337 338
338 339
339 340
340 341
341 342
342 343
343 344
344 345
345 346
346 347
347 348
348 349
349 350
350 351
351 352
352 353
353 354
354 355
355 356
356 357
357 358
358 359
359 360
360 361
361 362
362 363
363 364
364 365
365 366
366 367
367 368
368 369
369 370
370 371
371 372
372 373
373 374
374 375
375 376
376 377
377 378
378 379
379 380
380 381
381 382
382 383
383 384
384 385
385 386
386 387
387 388
388 389
389 390
390 391
391 392
392 393
393 394
394 395
395 396
396 397
397 398
398 399
399 400
400 401
401 402
402 403
403 404
404 405
405 406
406 407
407 408
408 409
409 410
410 411
411 412
412 413
413 414
414 415
415 416
416 417
417 418
418 419
419 420
420 421
421 422
422 423
423 424
424 425
425 426
426 427
427 428
428 429
429 430
430 431
431 432
432 433
433 434
434 435
435 436
436 437
437 438
438 439
439 440
440 441
441 442
442 443
443 444
444 445
445 446
446 447
447 448
448 449
449 450
450 451
451 452
452 453
453 454
454 455
455 456
456 457
457 458
458 459
459 460
460 461
461 462
462 463
463 464
464 465
465 466
466 467
467 468
468 469
469 470
470 471
471 472
472 473
473 474
474 475
475 476
476 477
477 478
478 479
479 480
480 481
481 482
482 483
483 484
484 485
485 486
486 487
487 488
488 489
489 490
490 491
491 492
492 493
493 494
494 495
495 496
496 497
497 498
498 499
499 500
500 501
501 502
502 503
503 504
504 505
505 506
506 507
507 508
508 509
509 510
510 511
511 512
512 513
513 514
514 515
515 516
516 517
517 518
518 519
519 520
520 521
521 522
522 523
523 524
524 525
525 526
526 527
527 528
528 529
529 530
530 531
531 532
532 533
533 534
534 535
535 536
536 537
537 538
538 539
539 540
540 541
541 542
542 543
543 544
544 545
545 546
546 547
547 548
548 549
549 550
550 551
551 552
552 553
553 554
554 555
555 556
556 557
557 558
558 559
559 560
560 561
561 562
562 563
563 564
564 565
565 566
566 567
567 568
568 569
569 570
570 571
571 572
572 573
573 574
574 575
575 576
576 577
577 578
578 579
579 580
580 581
581 582
582 583
583 584
584 585
585 586
586 587
587 588
588 589
589 590
590 591
591 592
592 593
593 594
594 595
595 596
596 597
597 598
598 599
599 600
600 601
601 602
602 603
603 604
604 605
605 606
606 607
607 608
608 609
609 610
610 611
611 612
612 613
613 614
614 615
615 616
616 617
617 618
618 619
619 620
620 621
621 622
622 623
623 624
624 625
625 626
626 627
627 628
628 629
629 630
630 631
631 632
632 633
633 634
634 635
635 636
636 637
637 638
638 639
639 640
640 641
641 642
642 643
643 644
644 645
645 646
646 647
647 648
648 649
649 650
650 651
651 652
652 653
653 654
654 655
655 656
656 657
657 658
658 659
659 660
660 661
661 662
662 663
663 664
664 665
665 666
666 667
667 668
668 669
669 670
670 671
671 672
672 673
673 674
674 675
675 676
676 677
677 678
678 679
679 680
680 681
681 682
682 683
683 684
684 685
685 686
686 687
687 688
688 689
689 690
690 691
691 692
692 693
693 694
694 695
695 696
696 697
697 698
698 699
699 700
700 701
701 702
702 703
703 704
704 705
705 706
706 707
707 708
708 709
709 710
710 711
711 712
712 713
713 714
714 715
715 716
716 717
717 718
718 719
719 720
720 721
721 722
722 723
723 724
724 725
725 726
726 727
727 728
728 729
729 730
730 731
731 732
732 733
733 734
734 735
735 736
736 737
737 738
738 739
739 740
740 741
741 742
742 743
743 744
744 745
745 746
746 747
747 748
748 749
749 750
750 751
751 752
752 753
753 754
754 755
755 756
756 757
757 758
758 759
759 760
760 761
761 762
762 763
763 764
764 765
765 766
766 767
767 768
768 769
769 770
770 771
771 772
772 773
773 774
774 775
775 776
776 777
777 778
778 779
779 780
780 781
781 782
782 783
783 784
784 785
785 786
786 787
787 788
788 789
789 790
790 791
791 792
792 793
793 794
794 795
795 796
796 797
797 798
798 799
799 800
800 801
801 802
802 803
803 804
804 805
805 806
806 807
807 808
808 809
809 810
810 811
811 812
812 813
813 814
814 815
815 816
816 817
817 818
818 819
819 820
820 821
821 822
822 823
823 824
824 825
825 826
826 827
827 828
828 829
829 830
830 831
831 832
832 833
833 834
834 835
835 836
836 837
837 838
838 839
839 840
840 841
841 842
842 843
843 844
844 845
845 846
846 847
847 848
848 849
849 850
850 851
851 852
852 853
853 854
854 855
855 856
856 857
857 858
858 859
859 860
860 861
861 862
862 863
863 864
864 865
865 866
866 867
867 868
868 869
869 870
870 871
871 872
872 873
873 874
874 875
875 876
876 877
877 878
878 879
879 880
880 881
881 882
882 883
883 884
884 885
885 886
886 887
887 888
888 889
889 890
890 891
891 892
892 893
893 894
894 895
895 896
896 897
897 898
898 899
899 900
900 901
901 902
902 903
903 904
904 905
905 906
906 907
907 908
908 909
909 910
910 911
911 912
912 913
913 914
914 915
915 916
916 917
917 918
918 919
919 920
920 921
921 922
922 923
923 924
924 925
925 926
926 927
927 928
928 929
929 930
930 931
931 932
932 933
933 934
934 935
935 936
936 937
937 938
938 939
939 940
940 941
941 942
942 943
943 944
944 945
945 946
946 947
947 948
948 949
949 950
950 951
951 952
952 953
953 954
954 955
955 956
956 957
957 958
958 959
959 960
960 961
961 962
962 963
963 964
964 965
965 966
966 967
967 968
968 969
969 970
970 971
971 972
972 973
973 974
974 975
975 976
976 977
977 978
978 979
979 980
980 981
981 982
982 983
983 984
984 985
985 986
986 987
987 988
988 989
989 990
990 991
991 992
992 993
993 994
994 995
995 996
996 997
997 998
998 999
999 1000

```

```

123 123          strcat(ret_str, n_bytes);
124 124          strcat(ret_str, "\n");
125 -
126 -          int c;
127 -          char conv2;
128 +          fclose(fp);
129 +          char * conv2 = malloc(sizeof(char));
130 +          fp = fopen(file_path, "r");
131 +          int c = fgetc(fp);
132 +          sprintf(conv2, "%i", c);
133 +          strcat(ret_str, conv2);
134 +          Writeline(conn_s, ret_str, MAX_LINE-1);
135 +          printf("c: %c\n", c);
136 +
137 137          c = fgetc(fp);
138 -          // printf("c: %c\n", c);
139 -          sprintf(&conv2, "%i", c);
140 -          strcat(ret_str, &conv2);
141 -          // Writeline(conn_s, ret_str, MAX_LINE-1);
142 -          // char c = fgetc
143 -          // while ((char) fgetc(fp) != EOF) {
144 -          //     memset(ret_str, 0, sizeof(ret_str));
145 -          //     if (strlen(ret_str) == MAX_LINE-1) {
146 -          //         Writeline(conn_s, ret_str, MAX_LINE-1);
147 -          //     } else {
148 -          //         strcat(ret_str, getc(fp));
149 -          //     }
150 -          // }
151 +          while (c != EOF) {
152 +              if (strlen(ret_str) == MAX_LINE-8){
153 +                  Writeline(conn_s, ret_str, MAX_LINE-1);
154 +                  memset(ret_str, 0, MAX_LINE);
155 +              } else {
156 +                  c = fgetc(fp);
157 +                  printf("C: %c", c);
158 +                  sprintf(conv2, "%d", c);
159 +                  strcat(ret_str, conv2);
160 +              }
161 +          }
162 +          fclose(fp);
163 +          // break;
164 141 148      }
165 142 149  }

```