

daytimeiodelay/client.py (Page 1 of 1)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # client.py
5: # Author: Bob Dondero
6: #-----
7: # Try this:
8: # python client.py time-a.nist.gov 13
9: #-----
10:
11: import sys
12: import socket
13:
14: #-----
15:
16: def main():
17:
18:     if len(sys.argv) != 3:
19:         print(f'usage: python {sys.argv[0]} host port', file=sys.stderr)
20:         sys.exit(1)
21:
22:     try:
23:         host = sys.argv[1]
24:         port = int(sys.argv[2])
25:
26:         with socket.socket() as sock:
27:             sock.connect((host, port))
28:             with sock.makefile(mode='r', encoding='ascii') as flo:
29:                 for line in flo:
30:                     print(line, end='')
31:
32:     except Exception as ex:
33:         print(f'{sys.argv[0]}: {ex}', file=sys.stderr)
34:         sys.exit(1)
35:
36: #-----
37:
38: if __name__ == '__main__':
39:     main()

```

blank (Page 1 of 1)

```

1: This page is intentionally blank.

```

daytimeiodelay/server.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # server.py
5: # Author: Bob Dondero
6: #-----
7:
8: import os
9: import sys
10: import socket
11: import time
12:
13: #-----
14:
15: IODELAY = int(os.environ.get('IODELAY', '0'))
16:
17: #-----
18:
19: def handle_client(sock):
20:
21:     # Simulate an I/O-bound server.
22:     time.sleep(IODELAY)
23:
24:     datetime = time.asctime(time.localtime())
25:     with sock.makefile(mode='w', encoding='ascii') as flo:
26:         flo.write(datetime + '\n')
27:
28: #-----
29:
30: def main():
31:
32:     if len(sys.argv) != 2:
33:         print(f'usage: python {sys.argv[0]} port', file=sys.stderr)
34:         sys.exit(1)
35:
36:     try:
37:         port = int(sys.argv[1])
38:
39:         server_sock = socket.socket()
40:         print('Opened server socket')
41:         if os.name != 'nt':
42:             server_sock.setsockopt(
43:                 socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
44:         server_sock.bind(('', port))
45:         print('Bound server socket to port')
46:         server_sock.listen()
47:         print('Listening')
48:
49:         while True:
50:             try:
51:                 sock, _ = server_sock.accept()
52:                 with sock:
53:                     print('Accepted connection')
54:                     print('Opened socket')
55:                     handle_client(sock)
56:                     print('Closed socket')
57:             except Exception as ex:
58:                 print(f'{sys.argv[0]}: {ex}', file=sys.stderr)
59:
60:     except Exception as ex:
61:         print(f'{sys.argv[0]}: {ex}', file=sys.stderr)
62:         sys.exit(1)
63:
64: #-----
65:

```

daytimeiodelay/server.py (Page 2 of 2)

```

66: if __name__ == '__main__':
67:     main()

```

daytimeiodelayp/server.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # server.py
5: # Author: Bob Dondero
6: #-----
7:
8: import os
9: import sys
10: import socket
11: import time
12: import multiprocessing
13:
14: #-----
15:
16: IODELAY = int(os.environ.get('IODELAY', '0'))
17:
18: #-----
19:
20: def worker_process_target(sock):
21:
22:     print('Forked child process')
23:
24:     # Simulate an I/O-bound server.
25:     time.sleep(IODELAY)
26:
27:     datetime = time.asctime(time.localtime())
28:     with sock:
29:         with sock.makefile(mode='w', encoding='ascii') as flo:
30:             flo.write(datetime + '\n')
31:
32:     print('Closed socket in child process')
33:     print('Exiting child process')
34:
35: #-----
36:
37: def main():
38:
39:     if len(sys.argv) != 2:
40:         print(f'usage: python {sys.argv[0]} port', file=sys.stderr)
41:         sys.exit(1)
42:
43:     try:
44:         port = int(sys.argv[1])
45:
46:         server_sock = socket.socket()
47:         print('Opened server socket')
48:         if os.name != 'nt':
49:             server_sock.setsockopt(
50:                 socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
51:         server_sock.bind(('', port))
52:         print('Bound server socket to port')
53:         server_sock.listen()
54:         print('Listening')
55:
56:         while True:
57:             sock, _ = server_sock.accept()
58:             with sock:
59:                 print('Accepted connection')
60:                 print('Opened socket')
61:                 worker_process = multiprocessing.Process(
62:                     target=worker_process_target, args=[sock])
63:                 worker_process.start()
64:             print('Closed socket in parent process')
65:

```

daytimeiodelayp/server.py (Page 2 of 2)

```

66:     except Exception as ex:
67:         print(f'{sys.argv[0]}: {ex}', file=sys.stderr)
68:         sys.exit(1)
69:
70: #-----
71:
72: if __name__ == '__main__':
73:     main()

```

daytimeiodelayt/server.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # server.py
5: # Author: Bob Dondero
6: #-----
7:
8: import os
9: import sys
10: import socket
11: import time
12: import threading
13:
14: #-----
15:
16: IODELAY = int(os.environ.get('IODELAY', '0'))
17:
18: #-----
19:
20: def worker_thread_target(sock):
21:     print('Spawned child thread')
22:
23:     # Simulate an I/O-bound server.
24:     time.sleep(IODELAY)
25:
26:     datetime = time.asctime(time.localtime())
27:     with sock:
28:         with sock.makefile(mode='w', encoding='ascii') as flo:
29:             flo.write(datetime + '\n')
30:
31:     print('Closed socket in child thread')
32:     print('Exiting child thread')
33:
34: #-----
35:
36:
37: def main():
38:
39:     if len(sys.argv) != 2:
40:         print(f'usage: python {sys.argv[0]} port', file=sys.stderr)
41:         sys.exit(1)
42:
43:     try:
44:         port = int(sys.argv[1])
45:
46:         server_sock = socket.socket()
47:         print('Opened server socket')
48:         if os.name != 'nt':
49:             server_sock.setsockopt(
50:                 socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
51:         server_sock.bind(('', port))
52:         print('Bound server socket to port')
53:         server_sock.listen()
54:         print('Listening')
55:
56:         while True:
57:             try:
58:                 sock, _ = server_sock.accept()
59:                 print('Accepted connection')
60:                 print('Opened socket')
61:                 worker_thread = threading.Thread(
62:                     target=worker_thread_target, args=[sock])
63:                 worker_thread.start()
64:             except Exception as ex:
65:                 print(f'{sys.argv[0]}: {ex}', file=sys.stderr)

```

daytimeiodelayt/server.py (Page 2 of 2)

```

66:
67:     except Exception as ex:
68:         print(f'{sys.argv[0]}: {ex}', file=sys.stderr)
69:         sys.exit(1)
70:
71: #-----
72:
73: if __name__ == '__main__':
74:     main()

```

daytimecdelay/server.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # server.py
5: # Author: Bob Dondero
6: #-----
7:
8: import os
9: import sys
10: import socket
11: import time
12:
13: #-----
14:
15: CDELAY = int(os.environ.get('CDELAY', '0'))
16:
17: #-----
18:
19: def consume_cpu_time(delay):
20:
21:     initial_thread_time = time.thread_time()
22:     while (time.thread_time() - initial_thread_time) < delay:
23:         pass
24:
25: #-----
26:
27: def handle_client(sock):
28:
29:     # Simulate a compute-bound server.
30:     consume_cpu_time(CDELAY)
31:
32:     datetime = time.asctime(time.localtime())
33:     with sock.makefile(mode='w', encoding='ascii') as flo:
34:         flo.write(datetime + '\n')
35:
36: #-----
37:
38: def main():
39:
40:     if len(sys.argv) != 2:
41:         print(f'usage: python {sys.argv[0]} port', file=sys.stderr)
42:         sys.exit(1)
43:
44:     try:
45:         port = int(sys.argv[1])
46:
47:         server_sock = socket.socket()
48:         print('Opened server socket')
49:         if os.name != 'nt':
50:             server_sock.setsockopt(
51:                 socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
52:         server_sock.bind(('', port))
53:         print('Bound server socket to port')
54:         server_sock.listen()
55:         print('Listening')
56:
57:         while True:
58:             try:
59:                 sock, _ = server_sock.accept()
60:                 with sock:
61:                     print('Accepted connection')
62:                     print('Opened socket')
63:                     handle_client(sock)
64:                     print('Closed socket')
65:             except Exception as ex:

```

daytimecdelay/server.py (Page 2 of 2)

```

66:                 print(f'{sys.argv[0]}: {ex}', file=sys.stderr)
67:
68:     except Exception as ex:
69:         print(f'{sys.argv[0]}: {ex}', file=sys.stderr)
70:         sys.exit(1)
71:
72: #-----
73:
74: if __name__ == '__main__':
75:     main()

```

daytimecdelayp/server.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2: #-----
3: # server.py
4: # Author: Bob Dondero
5: #-----
6: #
7:
8: import os
9: import sys
10: import socket
11: import time
12: import multiprocessing
13:
14: #-----
15:
16: CDELAY = int(os.environ.get('CDELAY', '0'))
17:
18: #-----
19:
20: def consume_cpu_time(delay):
21:     initial_thread_time = time.thread_time()
22:     while (time.thread_time() - initial_thread_time) < delay:
23:         pass
24:
25:
26: #-----
27:
28: def worker_process_target(sock):
29:
30:     print('Forked child process')
31:
32:     # Simulate a compute-bound server.
33:     consume_cpu_time(CDELAY)
34:
35:     datetime = time.asctime(time.localtime())
36:     with sock:
37:         with sock.makefile(mode='w', encoding='ascii') as flo:
38:             flo.write(datetime + '\n')
39:
40:     print('Closed socket in child process')
41:     print('Exiting child process')
42:
43: #-----
44:
45: def main():
46:
47:     if len(sys.argv) != 2:
48:         print(f'usage: python {sys.argv[0]} port', file=sys.stderr)
49:         sys.exit(1)
50:
51:     try:
52:         port = int(sys.argv[1])
53:
54:         server_sock = socket.socket()
55:         print('Opened server socket')
56:         if os.name != 'nt':
57:             server_sock.setsockopt(
58:                 socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
59:         server_sock.bind(('', port))
60:         print('Bound server socket to port')
61:         server_sock.listen()
62:         print('Listening')
63:
64:         while True:
65:             sock, _ = server_sock.accept()

```

daytimecdelayp/server.py (Page 2 of 2)

```

66:             with sock:
67:                 print('Accepted connection')
68:                 print('Opened socket')
69:                 worker_process = multiprocessing.Process(
70:                     target=worker_process_target, args=[sock])
71:                 worker_process.start()
72:                 print('Closed socket in parent process')
73:
74:     except Exception as ex:
75:         print(f'{sys.argv[0]}: {ex}', file=sys.stderr)
76:         sys.exit(1)
77:
78: #-----
79:
80: if __name__ == '__main__':
81:     main()

```

daytimecdelayt/server.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # server.py
5: # Author: Bob Dondero
6: #-----
7:
8: import os
9: import sys
10: import socket
11: import time
12: import threading
13:
14: #-----
15:
16: CDELAY = int(os.environ.get('CDELAY', '0'))
17:
18: #-----
19:
20: def consume_cpu_time(delay):
21:
22:     initial_thread_time = time.thread_time()
23:     while (time.thread_time() - initial_thread_time) < delay:
24:         pass
25:
26: #-----
27:
28: def worker_thread_target(sock):
29:
30:     print('Spawned child thread')
31:
32:     # Simulate a compute-bound server.
33:     consume_cpu_time(CDELAY)
34:
35:     datetime = time.asctime(time.localtime())
36:     with sock:
37:         with sock.makefile(mode='w', encoding='ascii') as flo:
38:             flo.write(datetime + '\n')
39:
40:     print('Closed socket in child thread')
41:     print('Exiting child thread')
42:
43: #-----
44:
45: def main():
46:
47:     if len(sys.argv) != 2:
48:         print(f'usage: python {sys.argv[0]} port', file=sys.stderr)
49:         sys.exit(1)
50:
51:     try:
52:         port = int(sys.argv[1])
53:
54:         server_sock = socket.socket()
55:         print('Opened server socket')
56:         if os.name != 'nt':
57:             server_sock.setsockopt(
58:                 socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
59:         server_sock.bind(('', port))
60:         print('Bound server socket to port')
61:         server_sock.listen()
62:         print('Listening')
63:
64:         while True:
65:             try:

```

daytimecdelayt/server.py (Page 2 of 2)

```

66:                 sock, _ = server_sock.accept()
67:                 print('Accepted connection')
68:                 print('Opened socket')
69:                 worker_thread = threading.Thread(
70:                     target=worker_thread_target, args = [sock])
71:                 worker_thread.start()
72:             except Exception as ex:
73:                 print(f'{sys.argv[0]}: {ex}', file=sys.stderr)
74:
75:     except Exception as ex:
76:         print(f'{sys.argv[0]}: {ex}', file=sys.stderr)
77:         sys.exit(1)
78:
79: #-----
80:
81: if __name__ == '__main__':
82:     main()

```

deadlock.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # deadlock.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class BankAcct:
13:
14:     def __init__(self, title):
15:         self._title = title
16:         self._balance = 0
17:         self._lock = threading.RLock()
18:
19:     def transfer_to(self, other, amount):
20:         self._lock.acquire()
21:         other._lock.acquire()
22:         self._balance -= amount
23:         other._balance += amount
24:         print(self._title + ': ' + str(self._balance))
25:         print(other._title + ': ' + str(other._balance))
26:         other._lock.release()
27:         self._lock.release()
28:
29: #-----
30:
31: def alice_to_bob_thread_target(alice_acct, bob_acct):
32:
33:     for _ in range(1000):
34:         alice_acct.transfer_to(bob_acct, 1)
35:
36: #-----
37:
38: def bob_to_alice_thread_target(bob_acct, alice_acct):
39:
40:     for _ in range(1000):
41:         bob_acct.transfer_to(alice_acct, 1)
42:
43: #-----
44:
45: def main():
46:
47:     alice_acct = BankAcct('Alice')
48:     bob_acct = BankAcct('Bob')
49:
50:     alice_to_bob_thread = threading.Thread(
51:         target=alice_to_bob_thread_target,
52:         args=[alice_acct, bob_acct])
53:     bob_to_alice_thread = threading.Thread(
54:         target=bob_to_alice_thread_target,
55:         args=[bob_acct, alice_acct])
56:
57:     alice_to_bob_thread.start()
58:     bob_to_alice_thread.start()
59:
60:     alice_to_bob_thread.join()
61:     bob_to_alice_thread.join()
62:
63:     print('Finished')
64:
65: #-----

```

deadlock.py (Page 2 of 2)

```

66:
67: if __name__ == '__main__':
68:     main()

```

deadlockw.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # deadlockw.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class BankAcct:
13:
14:     def __init__(self, title):
15:         self._title = title
16:         self._balance = 0
17:         self._lock = threading.RLock()
18:
19:     def transfer_to(self, other, amount):
20:         with self._lock:
21:             with other._lock:
22:                 self._balance -= amount
23:                 other._balance += amount
24:                 print(self._title + ': ' + str(self._balance))
25:                 print(other._title + ': ' + str(other._balance))
26:
27: #-----
28:
29: def alice_to_bob_thread_target(alice_acct, bob_acct):
30:
31:     for _ in range(1000):
32:         alice_acct.transfer_to(bob_acct, 1)
33:
34: #-----
35:
36: def bob_to_alice_thread_target(bob_acct, alice_acct):
37:
38:     for _ in range(1000):
39:         bob_acct.transfer_to(alice_acct, 1)
40:
41: #-----
42:
43: def main():
44:
45:     alice_acct = BankAcct('Alice')
46:     bob_acct = BankAcct('Bob')
47:
48:     alice_to_bob_thread = threading.Thread(
49:         target=alice_to_bob_thread_target,
50:         args=[alice_acct, bob_acct])
51:     bob_to_alice_thread = threading.Thread(
52:         target=bob_to_alice_thread_target,
53:         args=[bob_acct, alice_acct])
54:
55:     alice_to_bob_thread.start()
56:     bob_to_alice_thread.start()
57:
58:     alice_to_bob_thread.join()
59:     bob_to_alice_thread.join()
60:
61:     print('Finished')
62:
63: #-----
64:
65: if __name__ == '__main__':

```

deadlockw.py (Page 2 of 2)

```

66:     main()

```

nodeadlock.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # nodeadlock.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class BankAcct:
13:
14:     def __init__(self, title, sequence_num):
15:         self._title = title
16:         self._balance = 0
17:         self._sequence_num = sequence_num
18:         self._lock = threading.RLock()
19:
20:     def transfer_to(self, other, amount):
21:         if self._sequence_num < other._sequence_num:
22:             self._lock.acquire()
23:             other._lock.acquire()
24:             self._balance -= amount
25:             other._balance += amount
26:             print(self._title + ': ' + str(self._balance))
27:             print(other._title + ': ' + str(other._balance))
28:             other._lock.release()
29:             self._lock.release()
30:         else:
31:             other._lock.acquire()
32:             self._lock.acquire()
33:             self._balance -= amount
34:             other._balance += amount
35:             print(self._title + ': ' + str(self._balance))
36:             print(other._title + ': ' + str(other._balance))
37:             self._lock.release()
38:             other._lock.release()
39:
40: #-----
41:
42: def alice_to_bob_thread_target(alice_acct, bob_acct):
43:
44:     for _ in range(1000):
45:         alice_acct.transfer_to(bob_acct, 1)
46:
47: #-----
48:
49: def bob_to_alice_thread_target(bob_acct, alice_acct):
50:
51:     for _ in range(1000):
52:         bob_acct.transfer_to(alice_acct, 1)
53:
54: #-----
55:
56: def main():
57:
58:     alice_acct = BankAcct('Alice', 1)
59:     bob_acct = BankAcct('Bob', 2)
60:
61:     alice_to_bob_thread = threading.Thread(
62:         target=alice_to_bob_thread_target,
63:         args=[alice_acct, bob_acct])
64:     bob_to_alice_thread = threading.Thread(
65:         target=bob_to_alice_thread_target,

```

nodeadlock.py (Page 2 of 2)

```

66:         args=[bob_acct, alice_acct])
67:
68:     alice_to_bob_thread.start()
69:     bob_to_alice_thread.start()
70:
71:     alice_to_bob_thread.join()
72:     bob_to_alice_thread.join()
73:
74:     print('Finished')
75:
76: #-----
77:
78: if __name__ == '__main__':
79:     main()

```

nodeadlockw.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # nodeadlockw.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class BankAcct:
13:
14:     def __init__(self, title, sequence_num):
15:         self._title = title
16:         self._balance = 0
17:         self._sequence_num = sequence_num
18:         self._lock = threading.RLock()
19:
20:     def transfer_to(self, other, amount):
21:         if self._sequence_num < other._sequence_num:
22:             with self._lock:
23:                 with other._lock:
24:                     self._balance -= amount
25:                     other._balance += amount
26:                     print(self._title + ': ' + str(self._balance))
27:                     print(other._title + ': ' + str(other._balance))
28:         else:
29:             with other._lock:
30:                 with self._lock:
31:                     self._balance -= amount
32:                     other._balance += amount
33:                     print(self._title + ': ' + str(self._balance))
34:                     print(other._title + ': ' + str(other._balance))
35:
36: #-----
37:
38: def alice_to_bob_thread_target(alice_acct, bob_acct):
39:
40:     for _ in range(1000):
41:         alice_acct.transfer_to(bob_acct, 1)
42:
43: #-----
44:
45: def bob_to_alice_thread_target(bob_acct, alice_acct):
46:
47:     for _ in range(1000):
48:         bob_acct.transfer_to(alice_acct, 1)
49:
50: #-----
51:
52: def main():
53:
54:     alice_acct = BankAcct('Alice', 1)
55:     bob_acct = BankAcct('Bob', 2)
56:
57:     alice_to_bob_thread = threading.Thread(
58:         target=alice_to_bob_thread_target,
59:         args=[alice_acct, bob_acct])
60:     bob_to_alice_thread = threading.Thread(
61:         target=bob_to_alice_thread_target,
62:         args=[bob_acct, alice_acct])
63:
64:     alice_to_bob_thread.start()
65:     bob_to_alice_thread.start()

```

nodeadlockw.py (Page 2 of 2)

```

66:
67:     alice_to_bob_thread.join()
68:     bob_to_alice_thread.join()
69:
70:     print('Finished')
71:
72: #-----
73:
74: if __name__ == '__main__':
75:     main()

```