

conditions.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # conditions.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class BankAcct:
13:
14:     def __init__(self):
15:         self._balance = 0
16:         self._lock = threading.RLock()
17:         self._condition = threading.Condition(self._lock)
18:
19:     def get_balance(self):
20:         self._lock.acquire()
21:         try:
22:             return self._balance
23:         finally:
24:             self._lock.release()
25:
26:     def deposit(self, amount):
27:         self._condition.acquire()
28:         self._balance += amount
29:         print(self._balance)
30:         self._condition.notify_all()
31:         self._condition.release()
32:
33:     def withdraw(self, amount):
34:         self._condition.acquire()
35:         while self._balance < amount:
36:             self._condition.wait()
37:         self._balance -= amount
38:         print(self._balance)
39:         self._condition.release()
40:
41: #-----
42:
43: def deposit_thread_target(bank_acct):
44:
45:     for _ in range(10):
46:         bank_acct.deposit(1)
47:
48: #-----
49:
50: def withdraw_thread_target(bank_acct):
51:
52:     for _ in range(5):
53:         bank_acct.withdraw(2)
54:
55: #-----
56:
57: def main():
58:
59:     bank_acct = BankAcct()
60:
61:     deposit_thread = threading.Thread(
62:         target=deposit_thread_target, args=[bank_acct])
63:     withdraw_thread = threading.Thread(
64:         target=withdraw_thread_target, args=[bank_acct])
65:

```

conditions.py (Page 2 of 2)

```

66:     deposit_thread.start()
67:     withdraw_thread.start()
68:
69:     deposit_thread.join()
70:     withdraw_thread.join()
71:
72:     print('Final balance:', bank_acct.get_balance())
73:
74: #-----
75:
76: if __name__ == '__main__':
77:     main()

```

conditionsw.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # conditionsw.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class BankAcct:
13:
14:     def __init__(self):
15:         self._balance = 0
16:         self._lock = threading.RLock()
17:         self._condition = threading.Condition(self._lock)
18:
19:     def get_balance(self):
20:         with self._lock:
21:             return self._balance
22:
23:     def deposit(self, amount):
24:         with self._condition:
25:             self._balance += amount
26:             print(self._balance)
27:             self._condition.notify_all()
28:
29:     def withdraw(self, amount):
30:         with self._condition:
31:             while self._balance < amount:
32:                 self._condition.wait()
33:             self._balance -= amount
34:             print(self._balance)
35:
36: #-----
37:
38: def deposit_thread_target(bank_acct):
39:
40:     for _ in range(10):
41:         bank_acct.deposit(1)
42:
43: #-----
44:
45: def withdraw_thread_target(bank_acct):
46:
47:     for _ in range(5):
48:         bank_acct.withdraw(2)
49:
50: #-----
51:
52: def main():
53:
54:     bank_acct = BankAcct()
55:
56:     deposit_thread = threading.Thread(
57:         target=deposit_thread_target, args=[bank_acct])
58:     withdraw_thread = threading.Thread(
59:         target=withdraw_thread_target, args=[bank_acct])
60:
61:     deposit_thread.start()
62:     withdraw_thread.start()
63:
64:     deposit_thread.join()
65:     withdraw_thread.join()

```

conditionsw.py (Page 2 of 2)

```

66:
67:     print('Final balance:', bank_acct.get_balance())
68:
69: #-----
70:
71: if __name__ == '__main__':
72:     main()

```

envvar1.py (Page 1 of 1)

```
1: #!/usr/bin/env python
2:
3: #-----
4: # envvar1.py
5: # Author: Bob Dondero
6: #-----
7:
8: import os
9:
10: GREET = os.environ.get('GREETING', 'hi')
11:
12: def main():
13:     print(GREET)
14:
15: if __name__ == '__main__':
16:     main()
```

envvar2.py (Page 1 of 1)

```
1: #!/usr/bin/env python
2:
3: #-----
4: # envvar2.py
5: # Author: Bob Dondero
6: #-----
7:
8: import os
9: import dotenv
10:
11: dotenv.load_dotenv()
12: GREET = os.environ.get('GREETING', 'hi')
13:
14: def main():
15:     print(GREET)
16:
17: if __name__ == '__main__':
18:     main()
```

prodconprocesses.py (Page 1 of 1)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # prodconprocesses.py
5: # Author: Bob Dondero
6: #-----
7:
8: import multiprocessing
9:
10: #-----
11:
12: def producer_process_target(producer_conn):
13:
14:     for i in range(100):
15:         producer_conn.send(i)
16:         print('Produced:', i)
17:
18: #-----
19:
20: def consumer_process_target(consumer_conn):
21:
22:     for _ in range(100):
23:         num = consumer_conn.recv()
24:         print('Consumed:', num)
25:
26: #-----
27:
28: def main():
29:
30:     producer_conn, consumer_conn = multiprocessing.Pipe()
31:
32:     producer_process = multiprocessing.Process(
33:         target=producer_process_target, args=[producer_conn])
34:     consumer_process = multiprocessing.Process(
35:         target=consumer_process_target, args=[consumer_conn])
36:
37:     producer_process.start()
38:     consumer_process.start()
39:
40:     producer_process.join()
41:     consumer_process.join()
42:
43:     print('Finished')
44:
45: #-----
46:
47: if __name__ == '__main__':
48:     main()

```

prodconthreads.py (Page 1 of 1)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # prodconthreads.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9: import queue
10:
11: QUEUE_SIZE = 8 # Arbitrary
12:
13: #-----
14:
15: def producer_thread_target(intqueue):
16:
17:     for i in range(100):
18:         intqueue.put(i)
19:         print('Produced:', i)
20:
21: #-----
22:
23: def consumer_thread_target(intqueue):
24:
25:     for _ in range(100):
26:         num = intqueue.get()
27:         print('Consumed:', num)
28:
29: #-----
30:
31: def main():
32:
33:     intqueue = queue.Queue(QUEUE_SIZE)
34:
35:     producer_thread = threading.Thread(
36:         target=producer_thread_target, args=[intqueue])
37:     consumer_thread = threading.Thread(
38:         target=consumer_thread_target, args=[intqueue])
39:
40:     producer_thread.start()
41:     consumer_thread.start()
42:
43:     producer_thread.join()
44:     consumer_thread.join()
45:
46:     print('Finished')
47:
48: #-----
49:
50: if __name__ == '__main__':
51:     main()

```

Conditions.java (Page 1 of 2)

```

1: //-----
2: // Conditions.java
3: // Author: Bob Dondero
4: //-----
5:
6: class Bank
7: {
8:     private int balance = 0;
9:
10:    public synchronized int getBalance()
11:    {
12:        return balance;
13:    }
14:
15:    public synchronized void deposit(int amount)
16:    {
17:        balance += amount;
18:        System.out.println(balance);
19:        notifyAll();
20:    }
21:
22:    public synchronized void withdraw(int amount)
23:    {
24:        try
25:        {
26:            while (balance < amount)
27:                wait();
28:            balance -= amount;
29:            System.out.println(balance);
30:        }
31:        catch (InterruptedException e)
32:        {
33:            Thread.currentThread().interrupt();
34:        }
35:    }
36: }
37:
38: //-----
39:
40: class DepositThread extends Thread
41: {
42:     private Bank bank;
43:
44:     public DepositThread(Bank bank)
45:     {
46:         this.bank = bank;
47:     }
48:
49:     public void run()
50:     {
51:         for (int i = 0; i < 10; i++)
52:             bank.deposit(1);
53:     }
54: }
55:
56: //-----
57:
58: class WithdrawThread extends Thread
59: {
60:     private Bank bank;
61:
62:     public WithdrawThread(Bank bank)
63:     {
64:         this.bank = bank;
65:     }

```

Conditions.java (Page 2 of 2)

```

66:
67:     public void run()
68:     {
69:         for (int i = 0; i < 5; i++)
70:             bank.withdraw(2);
71:     }
72: }
73:
74: //-----
75:
76: public class Conditions
77: {
78:     public static void main(String[] args)
79:     {
80:         Bank bank = new Bank();
81:
82:         Thread depositThread = new DepositThread(bank);
83:         Thread withdrawThread = new WithdrawThread(bank);
84:
85:         depositThread.start();
86:         withdrawThread.start();
87:
88:         try
89:         {
90:             depositThread.join();
91:             withdrawThread.join();
92:         }
93:         catch (InterruptedException e)
94:         {
95:             Thread.currentThread().interrupt();
96:         }
97:
98:         System.out.println("Final balance: " + bank.getBalance());
99:     }
100: }

```