

processsharing.py (Page 1 of 1)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # processsharing.py
5: # Author: Bob Dondero
6: #-----
7:
8: import multiprocessing
9:
10: #-----
11:
12: def my_function(greeting):
13:
14:     greeting[1] = 'COS333'
15:
16: #-----
17:
18: def main():
19:
20:     greeting = ['hello', 'world']
21:
22:     child_process = multiprocessing.Process(
23:         target=my_function, args=[greeting])
24:     child_process.start()
25:     child_process.join()
26:
27:     print(greeting)
28:
29: #-----
30:
31: if __name__ == '__main__':
32:     main()

```

threadsharing.py (Page 1 of 1)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # threadsharing.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class MyThread (threading.Thread):
13:
14:     def __init__(self, greeting):
15:         threading.Thread.__init__(self)
16:         self._greeting = greeting
17:
18:     def run(self):
19:         self._greeting[1] = 'COS333'
20:
21: #-----
22:
23: def main():
24:
25:     greeting = ['hello', 'world']
26:
27:     child_thread = MyThread(greeting)
28:     child_thread.start()
29:     child_thread.join()
30:
31:     print(greeting)
32:
33: #-----
34:
35: if __name__ == '__main__':
36:     main()

```

race.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # race.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:
17:     def get_balance(self):
18:         return self._balance
19:
20:     def deposit(self, amount):
21:         temp = self._balance
22:         temp += amount
23:         print(temp)
24:         self._balance = temp
25:
26:     def withdraw(self, amount):
27:         temp = self._balance
28:         temp -= amount
29:         print(temp)
30:         self._balance = temp
31:
32: #-----
33:
34: class DepositThread (threading.Thread):
35:
36:     def __init__(self, bank_acct):
37:         threading.Thread.__init__(self)
38:         self._bank_acct = bank_acct
39:
40:     def run(self):
41:         for _ in range(10):
42:             self._bank_acct.deposit(1)
43:
44: #-----
45:
46: class WithdrawThread (threading.Thread):
47:
48:     def __init__(self, bank_acct):
49:         threading.Thread.__init__(self)
50:         self._bank_acct = bank_acct
51:
52:     def run(self):
53:         for _ in range(5):
54:             self._bank_acct.withdraw(2)
55:
56: #-----
57:
58: def main():
59:
60:     bank_acct = BankAcct()
61:
62:     deposit_thread = DepositThread(bank_acct)
63:     withdraw_thread = WithdrawThread(bank_acct)
64:
65:     deposit_thread.start()

```

race.py (Page 2 of 2)

```

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

```

locking.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # locking.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:
18:     def get_balance(self):
19:         self._lock.acquire()
20:         try:
21:             return self._balance
22:         finally:
23:             self._lock.release()
24:
25:     def deposit(self, amount):
26:         self._lock.acquire()
27:         temp = self._balance
28:         temp += amount
29:         print(temp)
30:         self._balance = temp
31:         self._lock.release()
32:
33:     def withdraw(self, amount):
34:         self._lock.acquire()
35:         temp = self._balance
36:         temp -= amount
37:         print(temp)
38:         self._balance = temp
39:         self._lock.release()
40:
41: #-----
42:
43: class DepositThread (threading.Thread):
44:
45:     def __init__(self, bank_acct):
46:         threading.Thread.__init__(self)
47:         self._bank_acct = bank_acct
48:
49:     def run(self):
50:         for _ in range(10):
51:             self._bank_acct.deposit(1)
52:
53: #-----
54:
55: class WithdrawThread (threading.Thread):
56:
57:     def __init__(self, bank_acct):
58:         threading.Thread.__init__(self)
59:         self._bank_acct = bank_acct
60:
61:     def run(self):
62:         for _ in range(5):
63:             self._bank_acct.withdraw(2)
64:
65: #-----

```

locking.py (Page 2 of 2)

```

66:
67: def main():
68:
69:     bank_acct = BankAcct()
70:
71:     deposit_thread = DepositThread(bank_acct)
72:     withdraw_thread = WithdrawThread(bank_acct)
73:
74:     deposit_thread.start()
75:     withdraw_thread.start()
76:
77:     deposit_thread.join()
78:     withdraw_thread.join()
79:
80:     print('Final balance:', bank_acct.get_balance())
81:
82: #-----
83:
84: if __name__ == '__main__':
85:     main()

```

lockingw.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # lockingw.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:
18:     def get_balance(self):
19:         with self._lock:
20:             return self._balance
21:
22:     def deposit(self, amount):
23:         with self._lock:
24:             temp = self._balance
25:             temp += amount
26:             print(temp)
27:             self._balance = temp
28:
29:     def withdraw(self, amount):
30:         with self._lock:
31:             temp = self._balance
32:             temp -= amount
33:             print(temp)
34:             self._balance = temp
35:
36: #-----
37:
38: class DepositThread (threading.Thread):
39:
40:     def __init__(self, bank_acct):
41:         threading.Thread.__init__(self)
42:         self._bank_acct = bank_acct
43:
44:     def run(self):
45:         for _ in range(10):
46:             self._bank_acct.deposit(1)
47:
48: #-----
49:
50: class WithdrawThread (threading.Thread):
51:
52:     def __init__(self, bank_acct):
53:         threading.Thread.__init__(self)
54:         self._bank_acct = bank_acct
55:
56:     def run(self):
57:         for _ in range(5):
58:             self._bank_acct.withdraw(2)
59:
60: #-----
61:
62: def main():
63:
64:     bank_acct = BankAcct()
65:

```

lockingw.py (Page 2 of 2)

```

66:     deposit_thread = DepositThread(bank_acct)
67:     withdraw_thread = WithdrawThread(bank_acct)
68:
69:     deposit_thread.start()
70:     withdraw_thread.start()
71:
72:     deposit_thread.join()
73:     withdraw_thread.join()
74:
75:     print('Final balance:', bank_acct.get_balance())
76:
77: #-----
78:
79: if __name__ == '__main__':
80:     main()

```

Spawning.java (Page 1 of 1)

```

1: //-----
2: // Spawning.java
3: // Author: Bob Dondero
4: //-----
5:
6: class PrinterThread extends Thread
7: {
8:     private String color;
9:
10:    public PrinterThread(String color)
11:    {
12:        this.color = color;
13:    }
14:
15:    public void run()
16:    {
17:        for (int i = 0; i < 5; i++)
18:            System.out.println(color);
19:    }
20: }
21:
22: //-----
23:
24: public class Spawning
25: {
26:     public static void main(String[] args)
27:     {
28:         PrinterThread blueThread = new PrinterThread("blue");
29:         PrinterThread redThread = new PrinterThread("red");
30:
31:         blueThread.start();
32:         redThread.start();
33:
34:         System.out.println("main thread terminated");
35:     }
36: }

```

Joining.java (Page 1 of 1)

```

1: //-----
2: // Joining.java
3: // Author: Bob Dondero
4: //-----
5:
6: class PrinterThread extends Thread
7: {
8:     private String color;
9:
10:    public PrinterThread(String color)
11:    {
12:        this.color = color;
13:    }
14:
15:    public void run()
16:    {
17:        for (int i = 0; i < 5; i++)
18:            System.out.println(color);
19:    }
20: }
21:
22: //-----
23:
24: public class Joining
25: {
26:     public static void main(String[] args)
27:     {
28:         PrinterThread blueThread = new PrinterThread("blue");
29:         PrinterThread redThread = new PrinterThread("red");
30:
31:         blueThread.start();
32:         redThread.start();
33:
34:         try
35:         {
36:             blueThread.join();
37:             redThread.join();
38:         }
39:         catch (InterruptedException e)
40:         {
41:             Thread.currentThread().interrupt();
42:         }
43:
44:         System.out.println("main thread terminated");
45:     }
46: }

```

Race.java (Page 1 of 2)

```

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

```

Race.java (Page 2 of 2)

```

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

```

Locking.java (Page 1 of 2)

```

1: //-----
2: // Locking.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:        int temp = balance;
18:        temp += amount;
19:        System.out.println(temp);
20:        balance = temp;
21:    }
22:
23:    public synchronized void withdraw(int amount)
24:    {
25:        int temp = balance;
26:        temp -= amount;
27:        System.out.println(temp);
28:        balance = temp;
29:    }
30: }
31:
32: //-----
33:
34: class DepositThread extends Thread
35: {
36:     private Bank bank;
37:
38:     public DepositThread(Bank bank)
39:     {
40:         this.bank = bank;
41:     }
42:
43:     public void run()
44:     {
45:         for (int i = 0; i < 10; i++)
46:             bank.deposit(1);
47:     }
48: }
49:
50: //-----
51:
52: class WithdrawThread extends Thread
53: {
54:     private Bank bank;
55:
56:     public WithdrawThread(Bank bank)
57:     {
58:         this.bank = bank;
59:     }
60:
61:     public void run()
62:     {
63:         for (int i = 0; i < 5; i++)
64:             bank.withdraw(2);
65:     }

```

Locking.java (Page 2 of 2)

```

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

```