# Создание потока

Thr = threading.Thread(target=get\_data, name=”name\_tread”, args=(arg1,)

Аргумент daemon=True – делает то, что в случае завершения работы основного потока заваршаются все потоки daemon

# Запуск потока

1) threading.Thread(target=get\_data, name=”name\_tread”, args=(arg1,).start()

2) Thr = threading.Thread(target=get\_data, name=”name\_tread”, args=(arg1,)

Thr.start()

# Состояние потока

Thr.is\_alive()

# Ожидание завершения работы потока

Thr.join() – пока поток не будет завершён, дальше скрипт не пойдёт

# Синхронизация потоков

1) locker = treading.Lock() – создаёт локер

Locker.acquire() – блокирует область для других потоков, пока не будет разблокировано (ставится в начала кода, который надо заблокировать)

Locker.release() – Разблокирует область для других потоков (ставится в конец кода, который надо заблокировать)

Также можно контекстный менеджер использовать

With locker:

Код

2) locker = treading.RLock() – более сложный блокировщик, который нельзя разблокировать с основного потока, только с того, где был заблокирован, работает также, как и обычно Lock

# Общие функции библиотеки threading

threading.active\_count() – Количество активных потоков

threading.main\_thread() – Информация об основном потоке

# Создание отложенного начала потока, который в случае чего можно отменить

thread\_test = threading.Timer(5, test)

thread\_test.start()

thread\_test.cancel() – отменяет запланированную задачу