**1. Назовите все состояния объекта Thread?**  
*NEW  
RUNNABLE  
BLOCKED  
WAITING  
TIMED\_WAITING  
TERMINATED*  
  
**2. В какие состояния может перейти нить, при входе в блок synchronized?**  
*RUNNABLE*  
*BLOCKED*  
  
В *RUNNABLE*, если блок кода, помеченный synchronized, не занят другой нитью. Иначе наша нить получит состояние *BLOCKED* и будет ждать освобождения объекта-мютекса.  
  
**3. В какое состояние перейдет нить, при вызове метода wait()?**  
Вызов этого метода переводит нить в состояние *WAITING*.  
Метод *wait()* можно вызвать только внутри блока synchronized у объекта-мютекса, который был «залочен (заблокирован)» текущей нитью, в противном случае метод выкинет исключение *IllegalMonitorStateException*. 

Object monitor = getMonitor();  
synchronized(monitor)  
{  
 …  
 monitor.wait();  
 …  
}

При вызове метода *wait()*, текущая нить снимает блокировку с объекта monitor, и переходит в состояние *WAITING*, ожидая вызова метода *monitor.notify()* или *monitor.notifyAll()* другой нитью. Как только это произойдет, нить проснется и если монитор не был занят, то захватит его и продолжит работу.  
Если монитор окажется занят другой нитью, текущая нить перейдет в состояние *BLOCKED*.  
  
**4. В какое состояние перейдет нить, при вызове метода wait(500)?**  
Вызов этого метода переводит нить в состояние *TIMED\_WAITING*.   
По аналогии с методом *wait()*, *wait(timeout)* можно вызвать только внутри блока synchronized у объекта-мютекса, который был «залочен (заблокирован)» текущей нитью. 

Object monitor = getMonitor();  
synchronized(monitor)  
{  
 …  
 monitor.wait(500);  
 …  
}

При вызове метода *wait()*, текущая нить снимает блокировку с объекта monitor, и засыпает на 500 миллисекунд. Объект monitor может быть захвачен другой нитью.  
Через 500 миллисекунд нить проснется и если monitor не был занят, то захватит его и продолжит работу.  
Если монитор окажется занят другой нитью, текущая нить перейдет в состояние *BLOCKED*.  
  
**5. В какое состояние перейдет нить, при вызове метода notify()?**

Object monitor = getMonitor();  
synchronized(monitor)  
{  
 …  
 monitor.wait();  
 …  
}

После *monitor.wait()*, нить перейдет в состояние *WAITING*. Метод *notify()*, вызванный другой нитью у объекта monitor переведет нить из состояния *WAITING* в состояние *RUNNABLE*, если объект monitor не будет захвачен другой нитью, иначе в состояние *BLOCKED*.  
  
**6. В какое состояние перейдет нить, при вызове метода notifyAll()?**  
*notifyAll()* «пробудет» все нити. Одна из всех «спящих» (*WAITING*) нитей перейдет в состояние *RUNNABLE*, захватит монитор используемого объекта и продолжит свою работу. Остальные окажутся в состоянии *BLOCKED*. Как только первая «проснувшаяся» нить отпустит монитор, который все остальные ожидают, её участь повторит следующая нить (произвольная нить из состояния *BLOCKED* перейдет в состояние *RUNNABLE*). Это будет продолжаться до тех пор, пока все «пробужденные» нити не покинут состояния *BLOCKED*.   
  
**7. Три нити в блоке synchronized вызвали wait() у объекта-мютекса. В какое состояние перейдут эти нити, если четвертая нить вызовет notifyAll()?**  
Две из них перейдут в состояние *BLOCKED*, одна в состояние *RUNNABLE*  
  
**8. Чем отличается join(500) от wait(500)?**  
Несмотря на то, что и *join(500)* и *wait(500)* переведут текущую нить в состояние *TIMED\_WAITING*, между ними существенные различия:  
  
*join(500)* вызывается у нити, *wait(500)* вызывается внутри синхронизированного блока у объекта, по которому данный блок синхронизирован.   
  
При вызове *join(500)* текущая нить будет ожидать 500 миллисекунд завершения нити, чей метод *join()* был вызван.   
При вызове *wait(500)* текущая нить снимет блокировку с синхронизированного объекта, и засыпает на 500 миллисекунд.   
  
Через 500 миллисекунд в обоих случаях нити продолжат работу.  
  
**9. Чем отличается wait(500) от sleep(500)?**  
*sleep(500)* вызывается у нити, *wait(500)* вызывается внутри синхронизированного блока у объекта, по которому данный блок синхронизирован.   
  
При вызове *sleep(500)* текущая нить будет ожидать 500 милисекунд, затем продолжит свою работу.  
При вызове *wait(500)* текущая нить снимет блокировку с синхронизированного объекта, и засыпает на 500 миллисекунд.   
  
**10. В какое состояние перейдет нить при вызове метода yield()?**  
При вызове метода *yield()* – текущая нить «пропускает свой ход» и java сразу переключается на выполнение следующей нити. Нить из состояния running переходит в состояние ready. Состояния running & ready – это подсостояния состояния *RUNNABLE*.