Quiz 2

Test what you have learnt so far.

Question # 1

Consider the below setup:

```
from threading import Condition
from threading import Thread
from threading import current_thread

flag = False

cond_var = Condition()


def child_task():
    global flag
    name = current_thread().getName()

    cond_var.acquire()
    if not flag:
        cond_var.wait()
        print("\n{0} woken up \n".format(name))

    flag = False
    cond_var.release()

    print("\n{0} exiting\n".format(name))


thread1 = Thread(target=child_task, name="thread1")
thread2 = Thread(target=child_task, name="thread2")

thread1.start()
thread2.start()

cond_var.acquire()
cond_var.notify_all()

thread1.join()
thread2.join()

print("main thread exits")
```

Q When run, what would be the outcome of the program and why?

- A) All threads including main thread runs to completion
- B) Atleast one thread hangs up on `wait()` call
- C) None of the threads make progress

Submit Answer

Reset Quiz

```
from threading import Condition
from threading import Thread
from threading import current_thread

flag = False

cond_var = Condition()


def child_task():
    global flag
    name = current_thread().getName()

    cond_var.acquire()
    if not flag:
        cond_var.wait()
        print("\n{0} woken up \n".format(name))

    flag = False
    cond_var.release()
```

```
21
22      print("\n{0} exiting\n".format(name))
23
24
25  thread1 = Thread(target=child_task, name="thread1")
26  thread2 = Thread(target=child_task, name="thread2")
27
28  thread1.start()
```

Question # 2

Consider the following snippet:

```
def worker_1(cv, sem):
    with cv:
        sem.acquire()


def worker_2(cv, sem):
    with cv:
        sem.release()

    print("Released by worker_1 thread")


if __name__ == "__main__":
    cv = Condition()
    sem = Semaphore(0)

    Thread(target=worker_2, args=(cv, sem)).start()
    Thread(target=worker_1, args=(cv, sem)).start()
```

Q What will be the output of the above program?

A) It will always complete without errors

B) It will deadlock some of the times

C) It will always deadlock

Submit Answer

Reset Quiz

Question # 3

Consider the following snippet:

```
1. thread = Thread(target=None)
2. thread.start()
3.
4. timer = Timer(1, None)
5. timer.start()
```

Q What will be the outcome of executing the above snippet?

A) The snippet would exit without errors

B) Both start calls would throw errors

C) The timer object's start call throws error

Submit Answer

Reset Quiz

```
1  from threading import Thread
2  from threading import Timer
3
4  thread = Thread(target=None)
5  thread.start()
6
7  timer = Timer(1, None)
8  timer.start()
9
```

Question # 4

Consider the snippet below:

(/learn)

```
from threading import Timer


def timer_task():
    print("timer task")


timer = Timer(5, timer_task)
timer.start()

print("Main thread exiting")
```

Q In the above snippet, the main thread exits before the timer task has had a chance to run. Will the program exit too without running the timer task since the main thread has exited?

- A) Yes
- B) No

Submit Answer

Reset Quiz

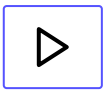
```
1  from threading import Timer
2
3
4  def timer_task():
```

```
    print("timer task")


timer = Timer(5, timer_task)
timer.start()

print("Main thread exiting")

```




← Back

Next →

Quiz 1

Introduction

Mark as Completed

Report an Issue

Ask a Question (https://discuss.educative.io/tag/quiz-2__threading-module__python-concurrency-for-senior-engineering-interviews)