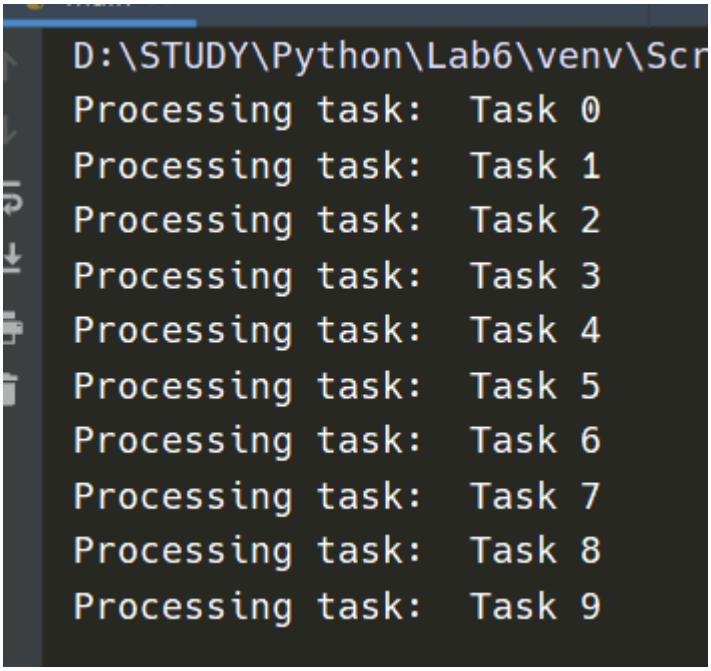
Задание 2 (вариант 5)

Самостоятельно изучите все основные способы синхронизации потоков.

Продемонстрируйте один из способом синхронизации потоков.

Вариант	Способ синхронизации
1	С помощью семафора (Semaphore)
2	Условием
3	Событием (Event Objects)
4	Барьером (Barrier Objects)
5	Очередью (Queue Objects)

```python
from threading import Thread
from queue import Queue


class Task:
    def __init__(self, num):
        self.description = "Task " + str(num)
        self.num = num


def process_tasks(q):
    while True:
        next_task = q.get()
        print("Processing task: ", next_task.description)
        q.task_done()


N = 10
q = Queue(N)
for i in range(N):
    q.put(Task(i))

workers = [Thread(target=process_tasks, args=(q,)),
           Thread(target=process_tasks, args=(q,)),
           ]
for w in workers:
    w.setDaemon(True)
    w.start()

q.join()
```


D:\STUDY\Python\Lab6\venv\Scr
Processing task: Task 0
Processing task: Task 1
Processing task: Task 2
Processing task: Task 3
Processing task: Task 4
Processing task: Task 5
Processing task: Task 6
Processing task: Task 7
Processing task: Task 8
Processing task: Task 9

Задание 3 (вариант 5, вариант 1)

Задание №3

Даны два набора по N векторов: $P = \{p_1, p_2, \dots, p_N)$ и $Q = \{q_1, q_2, \dots, q_N)$. Необходимо построить матрицу R размерностью $N \times N$, каждый элемент $r_{i,j}$ которой вычисляется по формуле:

Вариант	Формула
1	$r_{i,j} = \frac{1}{1 + \|q_j - p_i\|}, i = 1..N, j = 1..N, N = 5000$
2	$r_{i,j} = \frac{1}{1 + (q_j - p_i)^2}, i = 1..N, j = 1..N, N = 5000$
3	$r_{i,j} = \sqrt{(q_j - p_i)^2}, i = 1..N, j = 1..N, N = 5000$
4	$r_{i,j} = \sqrt{q_j^2 + p_i^2}, i = 1..N, j = 1..N, N = 5000$
5	$r_{i,j} = \sqrt{\|q_j - p_i\|}, i = 1..N, j = 1..N, N = 5000$

Сравните результат и время исполнения программы без использования модулей threading и multiprocessing, и с использованием модуля:

Вариант	Задание
1	threading
2	multiprocessing

```python
from threading import Thread
from math import sqrt
from random import randint
from time import time


N = 5000


def calculate_r(qi,pi):
    return sqrt(abs(qi-pi))

def fill_r(q,p):
    r = []
    for i in range(N):
        cur = []
        for j in range(N):
            r.append(calculate_r(q[j],p[i]))
        r.append(cur)


Q = []
P = []

for i in range(N):
   Q.append(randint(-5,5))
   P.append(randint(-5,5))

start = time()
fill_r(Q, P)
fill_r(Q, P)
print("Without threading: " + str(time()-start)+ " seconds")

start = time()
t1 = Thread(target=fill_r, args=(Q,P,))
t1.start()
t2 = Thread(target=fill_r, args=(Q,P,))
t2.start()
t1.join()
t2.join()
print("With threading: " + str(time()-start)+ " seconds")
```

```
D:\STUDY\Python\Lab6\venv\Scripts\python.exe D
Without threading: 22.161697387695312 seconds
With threading: 21.49915909767151 seconds
```

Задание 4 (вариант 1)

Задание №4

Выберите задание согласно варианту. Выполните задание в одном и в нескольких потоках. Сравните результаты и время выполнения программы.

Вариант	Задание
1	Получите содержимое 10 сайтов.

```python
import urllib.request
from time import time
from threading import Thread

sites = [
    "http://yandex.ru",
    "http://google.ru",
    "http://programmer.ucoz.com",
    "http://vk.com",
    "https://profi.ru",
    "https://docs.python.org/3/library/multiprocessing.html",
    "https://geekbrains.ru/posts/python_threading_part1",
    "https://orioks.miet.ru/",
    "https://www.youtube.com/",
    "https://www.crummy.com/software/BeautifulSoup/bs4/doc/"
]


def parse_sites():
    for site in sites:
        urllib.request.urlopen(site)


start = time()
parse_sites()
parse_sites()
print("With 1 thread: " + str(time() - start) + " seconds")

start = time()
t1 = Thread(target=parse_sites)
t1.start()
t2 = Thread(target=parse_sites)
t2.start()
t1.join()
t2.join()
print("With 2 threads: " + str(time() - start) + " seconds")
```

```
D:\STUDY\Python\Lab6\venv\Scripts\python.exe D
With 1 thread: 10.00900149345398 seconds
With 2 threads: 4.378299951553345 seconds
```