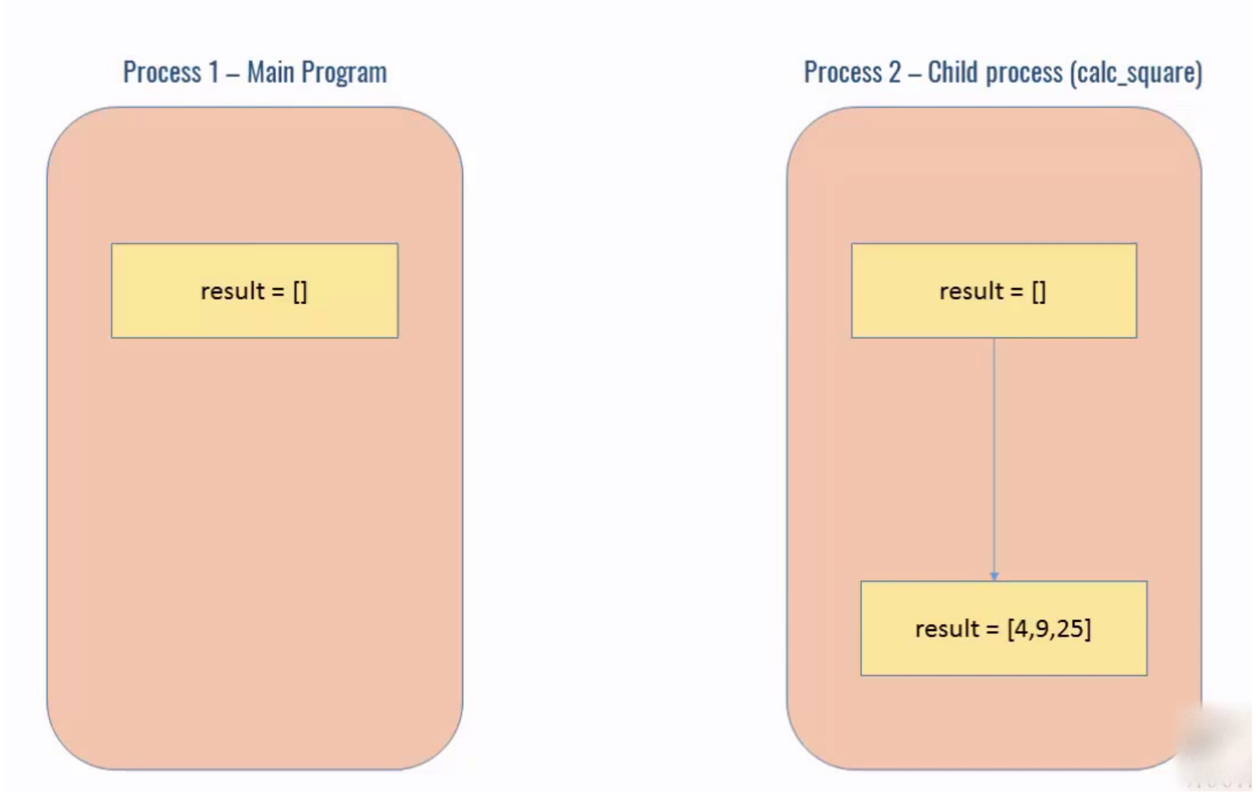
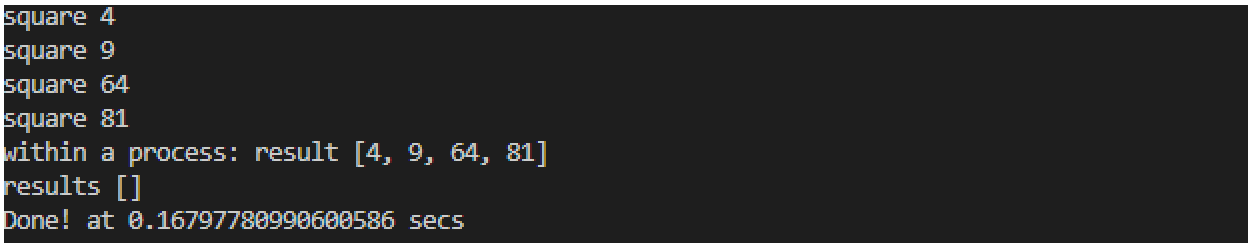
Video 4

Sharing Data Between Processes: Value and Array

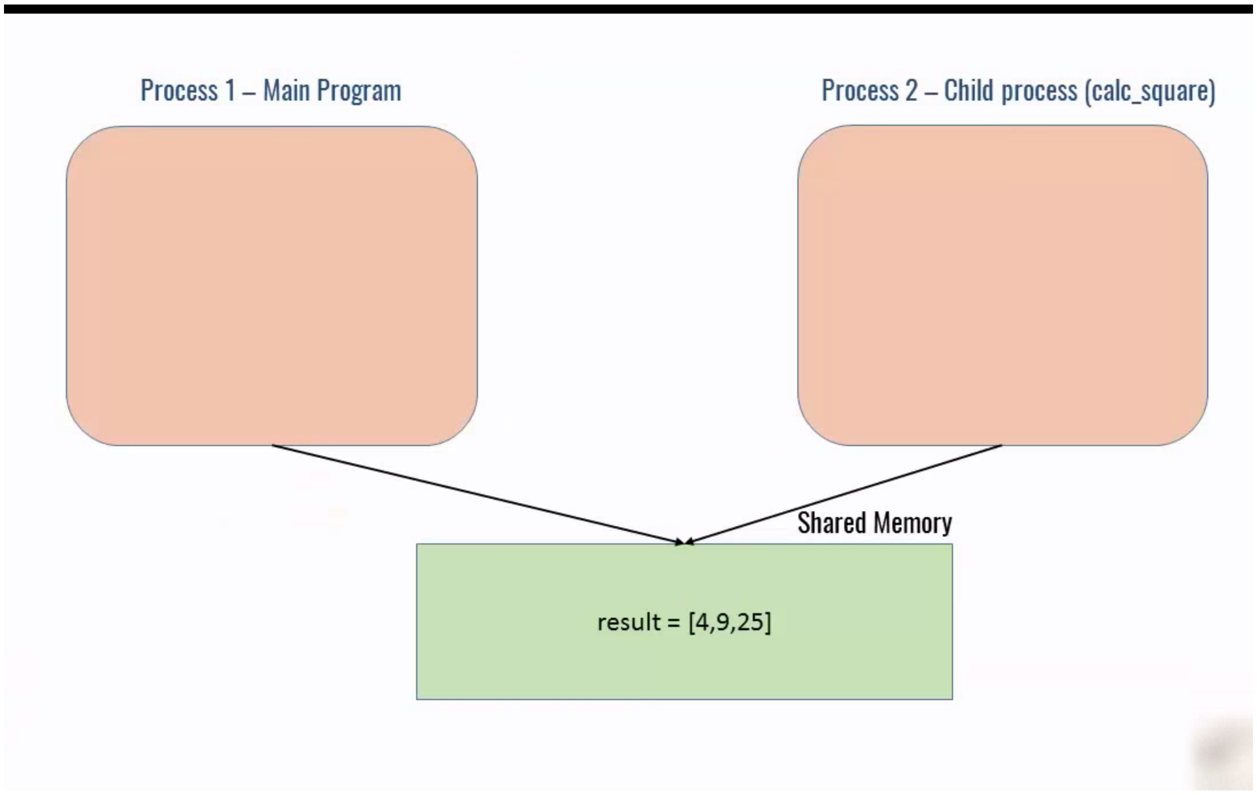


```
import time
import multiprocessing
square_results = []
def calc_square(numbers):
    global square_results
    for n in numbers:
        print(f"square {n*n}")
        square_results.append(n*n)
    print(f"within a process: result {square_results}")
if __name__ == "__main__":
    arr = [2,3,8,9]
    p1 = multiprocessing.Process(target=calc_square,args=(arr,))
    t = time.time()
    # start the processes
    p1.start()
    # wait until processes finish successfully!
    p1.join()
    print(f"results {square_results}")
    print(f"Done! at {time.time()-t} secs")
```

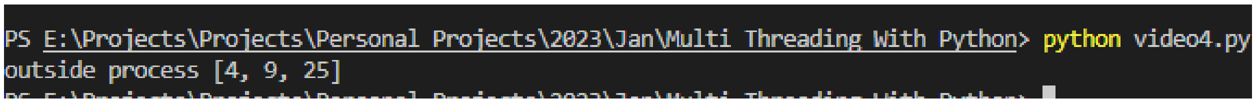


```
square 4
square 9
square 64
square 81
within a process: result [4, 9, 64, 81]
results []
Done! at 0.16797780990600586 secs
```

so as you can see in above result we can not access the results outside of the process and for fixing it there are several techniques that we can use, one those is shared memory

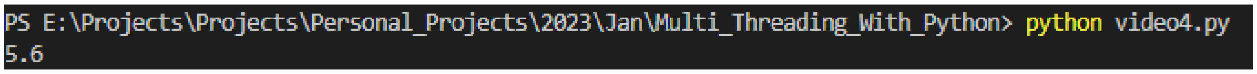


```
import multiprocessing
def calc_square(numbers, result):
  for idx, n in enumerate(numbers):
      result[idx] = n*n
if __name__ == "__main__":
    numbers = [2,3,5]
    result = multiprocessing.Array('i',3)
    p = multiprocessing.Process(target=calc_square,args=(numbers,result))
    p.start()
    p.join()
    print('outside process',str(result[:]))
```



```
PS E:\Projects\Projects\Personal Projects\2023\Jan\Multi Threading With Python> python video4.py
outside process [4, 9, 25]
```

```
import multiprocessing
def calc_square(numbers, result,v):
    v.value = 5.6
    for idx, n in enumerate(numbers):
      result[idx] = n*n
if __name__ == "__main__":
    numbers = [2,3,5]
    result = multiprocessing.Array('i',3)
    v = multiprocessing.Value('d',0.0)
    p = multiprocessing.Process(target=calc_square,args=(numbers,result,v))
    p.start()
    p.join()
    print(v.value)
```



```
PS E:\Projects\Projects\Personal_Projects\2023\Jan\Multi_Threading_With_Python> python video4.py
5.6
```