PYTHON ADVANCED PROGRAMMING

Chitra G M and P Rama Devi
Department of Computer Science Engineering

PYTHON ADVANCED PROGRAMMING

Inter Process Communication

Chitra G M and P Rama Devi
Department of Computer Science and Engineering

PYTHON ADVANCED PROGRAMMING

IPC

A process can be of two types:

Independent process.
Co-operating process.

An independent process is not affected by the execution of other processes while a co-operating process can be affected by other executing processes

Though one can think that those processes, which are running independently, will execute very efficiently, in reality, there are many situations when co-operative nature can be utilised for increasing computational speed, convenience and modularity.

Inter process communication (IPC) is a mechanism which allows processes to communicate with each other and synchronize their actions. The communication between these processes can be seen as a method of co-operation between them.

Processes can communicate with each other through both:

- Shared Memory
- Message passing

PYTHON ADVANCED PROGRAMMING

Consider the program

```
import multiprocessing
import time
result = []
def square_list(mylist):
    global result
    # append squares of mylist to global list result
    for num in mylist:
        result.append(num * num)
    # print global list result
    print("Result(in process p1)",result)
    time.sleep(1)
```

```
if __name__ == "__main__":
    # input list
    mylist = [1,2,3,4]
    # creating new process
    p1 = multiprocessing.Process(target=square_list,
args=(mylist,))
    # starting process
    p1.start()
    # wait until process is finished
    p1.join()
    # print global result list
    print("Result(in main program)",result)
```

When the above program is executed the output is available only in the childs address space.

So modify the above program such that the output is available in both child and parents address space

```
import multiprocessing
def square_list(mylist, result, square_sum):
    # append squares of mylist to result array
    for idx, num in enumerate(mylist):
        result[idx] = num * num
    # square_sum value
    square_sum.value = sum(result)

    # print result Array
    print("Result(in process p1):",result[:])
    # print square_sum Value
    print("Sum of squares(in process
p1):",square_sum.value)
```

PYTHON ADVANCED PROGRAMMING

```
if __name__ == "__main__":
    # input list
    mylist = [1,2,3,4]
    # creating Array of int data type with space for 4 integers
    result = multiprocessing.Array('i', 4)
    # creating Value of int data type
    square_sum = multiprocessing.Value('i')
  #  print(square_sum)
    # creating new process
    p1 = multiprocessing.Process(target=square_list, args=(mylist, result,
square_sum))
    # starting process
    p1.start()
    # wait until process is finished
    p1.join()
    # print result array
    print("Result(in main program):",result[:])
    # print square_sum Value
    print("Sum of squares(in main program):",square_sum.value)
```

THANK YOU

Chitra G M and P Rama Devi
Department of CSE

pramadevi@pes.edu
chitragm@pes.edu