**Indian Institute of technology, Guwahati**

**Department of Computer Science and Engineering**

**Data Structure Lab: (CS210)**

**Assignment: 2**

**Date: 18th August, 2016. Total Marks: 30 (lab assignments) + 40 (Offline assignments)**

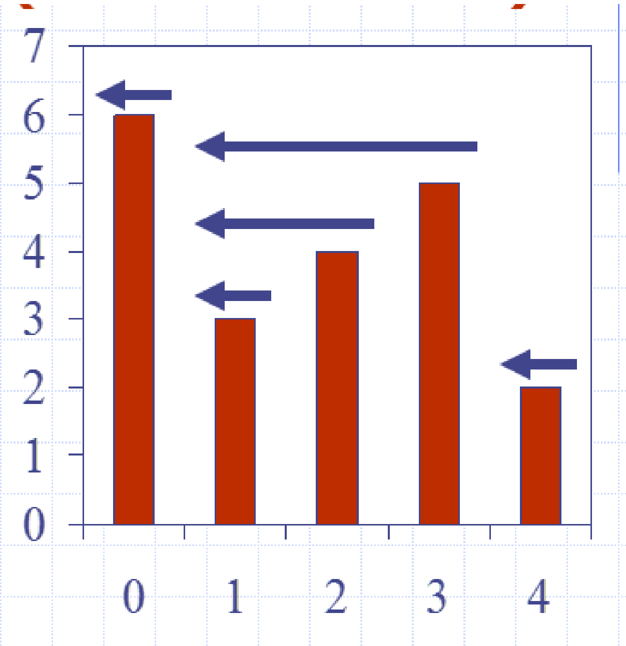
**Implement first three in Lab hour and submit rest by 24th August, 2016.**

**Lab Assignments:**

1. Define data structure for single link list. Write functions to (i) create a node (ii) Insert a new node after a given node in the list, (iii) Insert at the start of a list (iv) Insert a node at the end of a list, (v) delete a node after a given node in the list, (vi) delete the first node of the list and (vii) Delete the last node of the list. Create a list. Apply all the above operations randomly and print the final list. **(2x7 = 14)**
2. Define a stack. Implement push and pop operation using link list. Reuse the functions from problem 1. **(4x2=8)**
3. Define Queue. Implement Insert and Delete operations using link list. Reuse the functions from problem 1**.** **(4x2 = 8)**

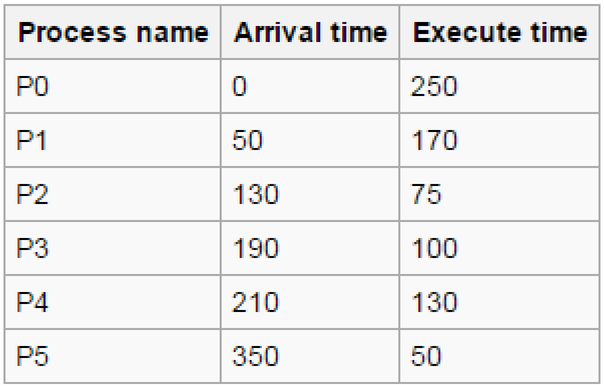
**Offline Assignments:**

4. Given an array X, the span S[i] of X[i] is the maximum number of consecutive elements X[j] immediately preceding X[i] and such that X[j] ≤ X[i]. Spans have applications to financial analysis - e.g., stock at 52-week high. A sample X and corresponding S is given below.
   - **a.** Write a O($n^2$) algorithm to find out array S for a given array X. **(6)**
   - b. Write a O(n) algorithm for the same using stack**.** **(12)**

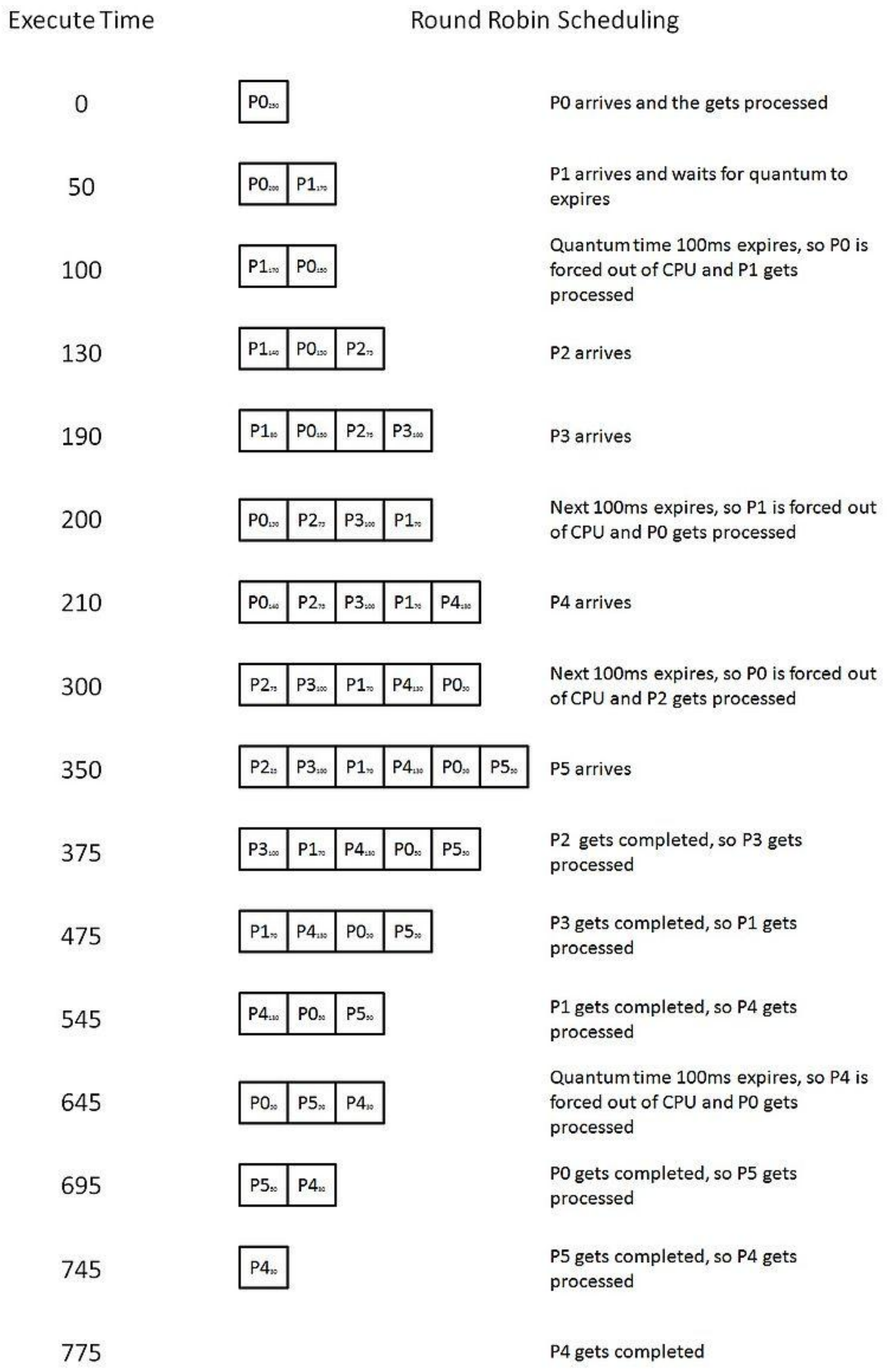


| | | | | | |
|---|---|---|---|---|---|
| *X* | 6 | 3 | 4 | 5 | 2 |
| *S* | 1 | 1 | 2 | 3 | 1 |

5. Round-robin Scheduling using Queue: Read detail about round robin scheduling from https://en.wikipedia.org/wiki/Round-robin_scheduling. Example is copied from the same link here. You have to implement Round-robin scheduling using queue. The input to your algorithm is the number of process, arrival time and execution time of each processes as given in below table and the time quantum for each process (e.g., 100 for the below example). The output would be the schedule of the processes. For example, you can print like: **(12)**
P0: <0-100>, <200-300>, <645-695>
P1: <100-200>, <475-545>,
so on…



| Process name | Arrival time | Execute time |
|---|---|---|
| P0 | 0 | 250 |
| P1 | 50 | 170 |
| P2 | 130 | 75 |
| P3 | 190 | 100 |
| P4 | 210 | 130 |
| P5 | 350 | 50 |



| Execute Time | Round Robin Scheduling | |
|---|---|---|
| 0 | $P0_{250}$ | P0 arrives and the gets processed |
| 50 | $P0_{200}$ $P1_{170}$ | P1 arrives and waits for quantum to expires |
| 100 | $P1_{170}$ $P0_{150}$ | Quantum time 100ms expires, so P0 is forced out of CPU and P1 gets processed |
| 130 | $P1_{140}$ $P0_{150}$ $P2_{75}$ | P2 arrives |
| 190 | $P1_{80}$ $P0_{150}$ $P2_{75}$ $P3_{100}$ | P3 arrives |
| 200 | $P0_{150}$ $P2_{75}$ $P3_{100}$ $P1_{70}$ | Next 100ms expires, so P1 is forced out of CPU and P0 gets processed |
| 210 | $P0_{140}$ $P2_{75}$ $P3_{100}$ $P1_{70}$ $P4_{130}$ | P4 arrives |
| 300 | $P2_{75}$ $P3_{100}$ $P1_{70}$ $P4_{130}$ $P0_{50}$ | Next 100ms expires, so P0 is forced out of CPU and P2 gets processed |
| 350 | $P2_{25}$ $P3_{100}$ $P1_{70}$ $P4_{130}$ $P0_{50}$ $P5_{50}$ | P5 arrives |
| 375 | $P3_{100}$ $P1_{70}$ $P4_{130}$ $P0_{50}$ $P5_{50}$ | P2 gets completed, so P3 gets processed |
| 475 | $P1_{70}$ $P4_{130}$ $P0_{50}$ $P5_{50}$ | P3 gets completed, so P1 gets processed |
| 545 | $P4_{130}$ $P0_{50}$ $P5_{50}$ | P1 gets completed, so P4 gets processed |
| 645 | $P0_{50}$ $P5_{50}$ $P4_{30}$ | Quantum time 100ms expires, so P4 is forced out of CPU and P0 gets processed |
| 695 | $P5_{50}$ $P4_{30}$ | P0 gets completed, so P5 gets processed |
| 745 | $P4_{30}$ | P5 gets completed, so P4 gets processed |
| 775 | | P4 gets completed |

6. Write an O(n) algorithm to determine if there is a cycle in a single link list. The function should return YES/NO for a given input link list. **(10)**