BITS Pilani
Hyderabad Campus

First Semester 2023-24
Data Structures and Algorithms Design (Merged-SEZG519/SSZG519)
Exercises (Elementary Data Structures)

1. Convert the following Infix expressions to Prefix and Postfix expressions.
   a. ((L+(M*N))/(O-P))
   b. ((L+M)*(N+P))
   c. (L+(M*N))
   d. (L*(M*(((N+L)+M)*N)))
   e. ((H*((((L+((M+N)*O))*F)*G)*P))+J)

2. The following algorithm is to implement the stack using two queues (i.e., Q1 and Q2) where pop and tos algorithms are computationally costly. Re-write push, pop, and tos algorithms where the computation complexity of push algorithm is high.

```
Algorithm push(o):
        if Q1.size( ) = N then
                indicate that a stack-full error has occurred
                return
        Q1.enqueue(o)
Algorithm pop( ):
        if Q1.isEmpty( ) then
                indicate that a stack-empty error has occurred
        for i = 1 to Q1.size( )-1 do
            Q2.enqueue(Q1.dequeue( ))
        e ← Q1.dequeue( )
        for i = 1 to Q2.size( ) do
            Q1.enqueue(Q2.dequeue( ))
        return e
Algorithm tos( ):
        if Q1.isEmpty( ) then
                indicate that a stack-empty error has occurred
        for i = 1 to Q1.size( )-1 do
            Q2.enqueue(Q1.dequeue( ))
        e ← Q1.dequeue( )
        Q2.enqueue(e)
        for i = 1 to Q2.size( ) do
            Q1.enqueue(Q2.dequeue( ))
        return e
```

3. The following algorithm is to implement the queue using two stacks (i.e., S1 and S2) where dequeue algorithm is computationally costly. Re-write enqueue and dequeue algorithms where the computation complexity of enqueue algorithm is high.

```
Algorithm enqueue(o):
        if S1.size( ) = N then
                indicate that a queue-full error has occurred
                return
        S1.push(o)
Algorithm dequeue( ):
        if S1.isEmpty( ) then
                indicate that a queue-empty error has occurred
                return NULL
        for i = 1 to S1.size( ) - 1 do
            S2.push(S1.pop( ))
        e ← S1.pop( )
        for i = 1 to S2.size( ) do
            S1.enqueue(S2.dequeue( ))
        return e
```