

# Programming Assignment 1: Pseudocode

## CS5280

Darpan Gaur  
CO21BTECH11004

---

### FOCC-CTA

#### variables

`item`  $\rightarrow$  vector  
stores the value of data items

`local_items`  $\rightarrow$  vector  
stores the value of data items local for transaction

`read_set`  $\rightarrow$  map(`trans_id`, vector)  
stores the set of data items read by each transaction

`write_set`  $\rightarrow$  map(`trans_id`, vectore)  
stores the set of data items written by each transaction

`read_list`  $\rightarrow$  map(`data_item`, vector)  
stores the list of transactions that have read each data item

`is_aborted`  $\rightarrow$  vector  
true if transaction is aborted, false otherwise

#### begin\_trans

```
begin_trans()  
{  
    // returns the id for the transaction  
    lock(id_lock);  
    trans_id = id + 1;  
    id++;  
}
```

```

        initialize: read_set[trans\_id]
        initialize: write_set[trans\_id]
        set is_aborted[trans\_id] = false
        unlock(id_lock);
        return trans_id;
    }

```

## **read(i, x, l)**

```

read(i, x, l)
{
    // i is the transaction id
    // x is the variable to be read
    // store value of x in l
    lock(item_lock);
    if (is_aborted[i]==true){
        free_trans(i);
        unlock(item_lock);
        return -1;
    }
    l -> local_items[x]
    read_set[i].push(x);
    read_list[x].insert(i);
    unlock(item_lock);
    return 0;
}

```

## **write(i, x, l)**

```

write(i, x, l)
{
    // i is the transaction id
    // x is the variable to be written
    // l is the value to be written
    lock(item_lock);
    if (is_aborted[i]==true){
        free_trans(i);
        unlock(item_lock);
        return -1;
    }
    update local variable l
}

```

```

    local_items[i] -> l
    write_set[i].push(x);
    unlock(item_lock);
    return 0;
}

```

## **try\_commit(i)**

```

try_commit(i)
{
    // i is the transaction id
    lock(item_lock);
    if (is_aborted[i]==true) {
        free_trans(i);
        unlock(item_lock);
        return a;
    }
    for d_id in write_set[i] {
        for t_id in read_list[d_id] {
            is_aborted[t_id] = true;
        }
    }

    update items vector from local_items vector
    that are in write_set[i]

    free_trans(i);
    unlock(item_lock);
    return c;
}

```

## **free\_trans(i)**

```

free_trans(i) {
    delete local_items
    delete read_set
    delete write set
    remove i from read_list
}

```