WORK SHEET 3

The following data structures are used in the first three questions.

```c
node{
    int i;          // variable index
    int low;        // low subtree
    int high;       // high subtree
    int count;      // reference counter
    int forward;    // next node on this level
    int backward;   // previous node on this level
    int next_free;  // next node on the free list
}
```

```c
int perm[n];       // permutation of variable indices
int level[n];      // node number of the first node on each level
int free_list;     // frist free node
```

The following procedures are used to manage the list which links the nodes for each level.

```c
Add_to_level(u){
    T[u].forward = level[T[u].i];
    if(level[T[u].i] != 0)
        T[level[T[u].i]].backward = u;
    level[T[u].i] = u;
    T[u].backward = 0;
}
```

```c
Remove_from_level(u){
    if(T[u].forward != 0)
        T[T[u].forward].backward = T[u].backward;
    if(T[u].backward != 0)
        T[T[u].backward].forward = T[u].forward;
    else
        level[T[u].i] = T[u].forward;
}
```

NOTE: Mk must be modified such that when MK allocates a new node, it is added to the level
(by a call to Add_to_level(u)). Additionally, every call to MK will automatically call Ref(u).

**Problem 1.** Write the pseudo-code for the procedure Ref(u), that will increase the reference count
for the node u.
Ref(u) {
    T[u].count++;
}

Problem 2. Write the pseudo-code for the procedure RecursiveDeref(u), that will recursively decrease the reference count for the node u and its descendents. When the reference count of a node reaches zero, it must be added to the freelist.

RecursiveDeref(u) {
    T[u].count--;
    if (T[u].count == 0) {
        Add_to_freelist(u);
        Remove_from_level(u);
        RecursiveDeref(T[u].low);
        RecursiveDeref(T[u].high);
    }
}

Problem 3. Given the procedure ExchangeOrder(i), that will interchange the order of the variables i and i+1 (i is an index into the array perm), write the pseudo-code for ExchangeOneNode(u,11,12), that will move node u from level 11 to 12.

ExchangeOrder(i) {
    int u = level[perm[i]];
    while(u != 0) {
        next = t[u].forward;
        ExchangeOneNode(u,perm[i],perm[i+1]);
        u = next;
    }
}

ExchangeOneNode(u,11,12) {
    low_child = T[u].low;
    high_child = T[u].high;
    if(((T[low_child].i == 12) || (T[low_child].i == 12)) {  
        // at least one child must be at the next level
        if(T[low_child].i == 12) {
            ll = T[low_child].low;
            lh = T[low_child].high;
        } else {
            ll = lh = low_child;
        }
    } else {
        ll = lh = high_child;
    }
    Remove_from_level(u);
    T[u].i = 12;
}
Problem 4. Given the function \texttt{add4} with 4 inputs and 3 outputs, where the inputs represent two 2-bit integers and the output represents their sum. How many garbage outputs are needed to make the function reversible? How many reversible functions (with the calculated number of variables) contain \texttt{add4}? Assume that the 3 output positions are fixed.

Problem 5. Find a Toffoli realization of the reversible function \( F(a, b, c) = [3, 2, 1, 0, 4, 5, 6, 7] \). Use the second algorithm described in class.

Problem 6. Find the function realized by the following circuit.

![Circuit Diagram](attachment:image.png)

Problem 7. Prove that there is no sequence of 3 Toffoli gates that produces the identity function.