**Question 1 – (25 points) – IntegerSet ADT**

class IntegerSet<type>

Constructor, etc…

// a) Returns the set of all elements of set X with element “a” added if it is not present.

IntegerSet insert(a,X){

Create a new IntegerSet called Temp;

While (X is not empty){

If compare(X.first < or = a){

Temp.add(X.first);

X = X.rest;

}

Else if compare(X.first > a){

Temp.add(X.first);

X = X.rest;

}

}

Return Temp;

}

// b) Returns the set of all elements of set X with element “a” removed if it is present.

IntegerSet delete(a,X){

Create a new IntegerSet called Temp;

While (X is not empty){

If compare(X.first is = a){

X = X.rest;

}

Else if compare(X.first is not = a){

Temp.add(X.first);

X = X.rest;

}

}

Return Temp;

}

// c) Returns whether set X contains element “a”. (True/False)

Boolean member(a,X){

Create a new IntegerSet called Temp;

Temp = X;

While (Temp is not empty){

If compare(Temp.first is = a){

Return true;

}

Else

Temp = Temp.rest;

}

Return false;

}

// d) Returns the set corresponding to the union of sets X and Y.

IntegerSet union(X,Y){

Create a new IntegerSet called Temp;

While(X is not empty AND Y is not empty){

If compare(X.first = Y.first){

Temp.add(X.first);

X = X.rest;

Y = Y.rest;

}

Else if compare(X.first < Y.first){

Temp.add(X.first);

X = X.next;

}

Else if compare(X.first > Y.first){

Temp.add(Y.first);

Y = Y.empty;

}

// At this point either X or Y is empty, so only need to empty the other one

While(X is not empty){

Temp.add(X.next);

X = X.rest;

}

While(Y is not empty){

Temp.add(Y.next);

Y = Y.rest;

}

Return Temp;

}

// e) Returns the set corresponding to the intersection of sets X and Y.

IntegerSet intersection(X,Y){

Create a new IntegerSet called Temp;

While(X is not empty AND Y is not empty){

If compare(X.first = Y.first)

Temp = X.next;

X = X.rest;

Y = Y.rest;

}

Else if compare(X.first < Y.first){

X = X.rest;

}

Else if compare(Y.first < X.first){

Y = Y.rest;

}

}

Return Temp;

}