### Chapter 7

**15.**

A sequential search member function of SortedType has the following prototype:

void SortedType::Search(int value, bool& found);

A. Write the function definition as a recursive search, assuming a linked list implementation.

B. Write the function definition as a recursive search, assuming an array-based implementation.

Answer:

A.

void SortedType::Search(ItemType value, bool& found)

{

found = false;

RecursiveSearch(value, found, listData);

}

void SortedType::RecursiveSearch(ItemType value, bool& found, NodeType\* location)

{

if(location == NULL)

return;

else {

switch(value.ComparedTo(location->info)) {

case GREATER:

RecursiveSearch(value, found, location->next);

break;

case EQUAL:

found = true;

break;

}

}

}

B.

void SortedType::Search(ItemType value, bool& found)

{

RecursiveSearch(value, found, 0);

}

void SortedType::RecursiveSearch(ItemType value, bool& found, int location)

{

found = false;

if(location == length)

return;

else {

switch(value.ComparedTo(info[location])) {

case GREATER:

RecursiveSearch(value, found, location + 1);

std::cout << "Greater" << std::endl;

break;

case EQUAL:

found = true;

break;

}

}

}