**Karadeniz Technical University**
**Department of Computer Engineering**
**Lecturer Ömer ÇAKIR**

# BIL 205 Data Structures

Midterm Exam II, 12.12.2011
Time : 90 Minutes

Rules to be Observed During the Exam

**1.** Cell phones are **not allowed** to be used as a calculator or a watch. They must be **switched off** and **placed in pocket**.
**2.** Questions might be asked in the first **20 minutes**. Each student is free to ask only **one question**.
**3.** Sign this paper after writing your number and name. You will take **this paper**.

| NUMBER : | NAME : | SIGNATURE : |
|---|---|---|

```
void addBelowRootModified(Node* p, int elt)
{
    while(p->left != NULL && p->right != NULL)
    {
       if( p->elt > elt)
           p = p->left;
       else
           p = p->right;
    }

    Node* newNode = new Node;
    newNode->elt = elt;
    newNode->par = p;

    if (newNode->elt < p->elt)
       p->left = newNode;
    else
       p->right = newNode;

    n += 1;
}

void main()
{
   LinkedBinaryTree nuTree;

   nuTree.addRoot();
   nuTree._root->elt = 8;

   nuTree.addBelowRootModified (nuTree._root, 7);
   nuTree.addBelowRootModified (nuTree._root, 6);
   nuTree.addBelowRootModified (nuTree._root, 5);
   nuTree.addBelowRootModified (nuTree._root, 4);
   nuTree.addBelowRootModified (nuTree._root, 3);
   nuTree.addBelowRootModified (nuTree._root, 2);
   nuTree.addBelowRootModified (nuTree._root, 1);
   nuTree.addBelowRootModified (nuTree._root, 9);
   nuTree.addBelowRootModified (nuTree._root, 10);
   nuTree.addBelowRootModified (nuTree._root, 11);
   nuTree.addBelowRootModified (nuTree._root, 12);
   nuTree.addBelowRootModified (nuTree._root, 13);
   nuTree.addBelowRootModified (nuTree._root, 14);
   nuTree.addBelowRootModified (nuTree._root, 15);

   cout<< "Inorder Traversal : ";
   nuTree.inorder(nuTree._root);
}
```

**1.**

**a)** Write the output of the program above? **(15P)**

**b)** `addBelowRootModified()` function works correctly when the tree elements are in the order shown below but works erroneously in the order above. Fix the error in the function above. Write down the whole function, not only the corrected part. **(15P)**

**8 4 12 2 6 10 14 1 3 5 7 9 11 13 15**

**8 7 6 5 4 3 2 1 9 10 11 12 13 14 15**

**2.**

**a)** Insert the elements above into a binary tree.

**b)** Write elements using **inorder** rule.

**c)** Write elements using **preorder** rule.

**d)** Write elements using **postorder** rule.

**Note →** When inserting elements into a binary tree, insert 8 (root) first, insert 7 second ... and insert 15 last. **(40P)**

```
void HeapPriorityQueue::insert(const int& e)
{
        T.addLast(e);
        Position v = T.last();
        while (!T.isRoot(v)) {
                Position u = T.parent(v);
                if (!isLess(*v, *u)) break;
                T.swap(v, u);
                v = u;
        }
}

void main()
{
        HeapPriorityQueue Heap;

        Heap.insert(8);
        Heap.insert(7);
        Heap.insert(6);
        Heap.insert(5);
        Heap.insert(4);
        Heap.insert(3);
        Heap.insert(2);
        Heap.insert(1);
        Heap.insert(9);
        Heap.insert(10);
        Heap.insert(11);
        Heap.insert(12);
        Heap.insert(13);
        Heap.insert(14);
        Heap.insert(15);

        cout << "Heap Elements after Insertions :";
        Heap.print();

        Heap.removeMin();

        cout << "Heap Elements after removeMin  :";
        Heap.print();
}
```

**3.** Write the output of the program above? **(30P)**